SMART TRANSPORTATION SYSTEM

Master in Innovation and Research in Informatics
Universitat Politècnica de Catalunya

Mohanad El Daflawy

Advisor: Xavier Franch

4/17/2015
# Table of Contents

**Table of Contents**

List of Figures ............................................................................................................. 5

List of Tables ............................................................................................................... 7

1. Vision ....................................................................................................................... 9

2. Introduction ............................................................................................................. 10

3. Requirements ........................................................................................................ 12
   1-star modelling: ..................................................................................................... 12
   2.1 Functional Design ............................................................................................... 13
      2.1.1 Management and Visualization .................................................................. 13
      2.1.2 Work Order Management ......................................................................... 96
      2.1.3 Incidence Management .............................................................................. 114
   2.2 Non Functional ................................................................................................ 121
      2.2.1 Hardware Requirements ........................................................................... 121
      2.2.2 Software Requirements .......................................................................... 121
      2.2.3 Performance Requirements ..................................................................... 121
      2.2.4 Security Requirements ............................................................................ 122
      2.2.5 Interface Requirements .......................................................................... 122
      2.2.6 Availability Requirements ....................................................................... 122

4. Time plan ................................................................................................................ 123

5. Business logic ........................................................................................................ 124
   5.1 New client ........................................................................................................... 124
   5.2 New order ......................................................................................................... 125
   5.3 Work orders generation .................................................................................... 125
      3.3.1 For each distribution center ...................................................................... 126
      3.3.2 Get the orders that are not assigned ....................................................... 127
      3.3.3 Get vehicles that are available .................................................................. 127
3.4 Work order real-time tracking

3.4.1 Real-time monitoring

3.4.1.1 From element to system

3.4.1.1.1 Receive JSON Object

3.4.1.1.2 Process JSON Object

3.4.1.1.3 Save in cache memory

3.4.1.1.4 Push data in a database access queue

3.4.1.2 From system to user

4. Incidences Management

4.3 DISTRIBUTION CENTER INCIDENCES

4.4 INTERMODAL PLATFORM INCIDENCES

4.5 VEHICLE/ROUTE INCIDENCES

4.6 WAGON/ROUTE INCIDENCES

4.7 INCIDENCES AT DESTINATION

5. Architecture Overview

5.1 Architecture objectives

5.1.1 Scalability

5.1.2 Costs

5.1.3 Customizable

5.2 Application components

5.2.1 Apache Web Server
5.2.2 Tomcat Server ........................................................................................................... 144
5.2.3 GeoServer .................................................................................................................. 144
5.2.4 MemCached: ............................................................................................................... 144
5.2.5 PostgreSQL + PostGIS database: .............................................................................. 144
5.3 Architecture Diagram .................................................................................................. 145
5.4 Architecture Patterns .................................................................................................. 145
5.4.1 Single server ............................................................................................................... 145
5.4.2 Database replication ................................................................................................... 146
5.4.3 Server replication ...................................................................................................... 147
5.4.4 Database sharding ..................................................................................................... 147
5.5 Design Patterns ............................................................................................................ 148
5.5.1 Layered architecture style .......................................................................................... 149
5.5.2 Dependency injection pattern .................................................................................... 149
5.5.3 Model-View-Controller pattern ................................................................................ 150
5.5.4 Service oriented architecture pattern ...................................................................... 150
5.5.5 Data access object pattern ....................................................................................... 151
5.6 Database access .......................................................................................................... 151
5.6.1 Database connection ............................................................................................... 151
5.6.2 Memcached ............................................................................................................. 151
5.7 Technologies ................................................................................................................ 152
5.7.1 Java .......................................................................................................................... 152
5.7.2 Spring boot ............................................................................................................... 152
5.7.3 Thymeleaf ............................................................................................................... 152
5.7.4 Twitter bootstrap ...................................................................................................... 153
5.7.5 JavaScript ................................................................................................................ 153
5.7.6 Leaflet.js .................................................................................................................. 153
5.7.7 Hibernate ................................................................................................................ 153
5.7.8 PostgreSQL + PostGIS ............................................................................................ 153
6. Smart transportation system Data Model ...................................................................... 154
LIST OF FIGURES

Figure 1............................................................................................................ 13
Figure 2............................................................................................................ 15
Figure 3............................................................................................................ 19
Figure 4............................................................................................................ 20
Figure 5............................................................................................................ 30
Figure 6............................................................................................................ 31
Figure 7............................................................................................................ 39
Figure 8............................................................................................................ 40
Figure 9............................................................................................................ 48
Figure 10......................................................................................................... 49
Figure 11......................................................................................................... 58
Figure 12......................................................................................................... 65
Figure 13......................................................................................................... 71
Figure 14......................................................................................................... 72
Figure 15......................................................................................................... 80
Figure 16......................................................................................................... 86
Figure 17......................................................................................................... 87
Figure 18......................................................................................................... 96
Figure 19......................................................................................................... 102
Figure 20........................................................................................................ 110
Figure 21........................................................................................................ 114
Figure 22........................................................................................................ 115
Figure 23........................................................................................................ 115
Figure 24........................................................................................................ 124
Figure 25........................................................................................................ 125
Figure 26........................................................................................................ 125
Figure 27........................................................................................................ 126
Figure 28........................................................................................................ 128
Figure 29........................................................................................................ 128
Figure 30........................................................................................................ 130
Figure 31........................................................................................................ 131
Figure 32........................................................................................................ 133
# LIST OF TABLES

Table 1 .................................................................................................................. 12  
Table 2 .................................................................................................................. 123  
Table 3 .................................................................................................................. 138  
Table 4 .................................................................................................................. 156  
Table 5 .................................................................................................................. 156  
Table 6 .................................................................................................................. 157  
Table 7 .................................................................................................................. 159  
Table 8 .................................................................................................................. 159  
Table 9 .................................................................................................................. 160  
Table 10 ............................................................................................................... 161  
Table 11 ............................................................................................................... 163  
Table 12 ............................................................................................................... 163  
Table 13 ............................................................................................................... 165  
Table 14 ............................................................................................................... 165  
Table 15 ............................................................................................................... 166  
Table 16 ............................................................................................................... 166  
Table 17 ............................................................................................................... 168  
Table 18 ............................................................................................................... 168  
Table 19 ............................................................................................................... 168
I would like to dedicate this project to my mother.
She encouraged me to study this master and without her support I wouldn’t have been able to write this thesis.
1. VISION

Smart transportation system is a GIS routing web-based application for industrial and logistics purposes. The main idea behind the smart transportation application is that the system will be having different elements such as vehicles, wagons, boxes to carry the goods, batteries where the vehicles will run on, operation centres to park and recharge the batteries, and distribution centre where the orders will start from for delivery. However, the app will be controlling all the elements in order to form a logistic company, for example. Basically, the vehicles will be operating on batteries. Users of the app will be divided into groups and each group will have different access level. A typical case will be a new purchase order entered into the system, the system will process the details of that purchase order, assign the boxes and suggest an order plan with the route that should be followed giving exact directions and time. The user will be able to locate elements on the map and make some modifications to a plan.

Multimodal routing systems in general allow the following:

- From point to point with some points in between.
- Takes into account the road network limitations such as construction and traffic.
- Planning is done according to a specific type of transportation either car, bus, train, walking or a bicycle.

However, the planning system that we will provide will take into account more restrictions to calculate the route such as:

- The charge level of the batteries.
- Simulation of discharging the batteries in proportion to the route (speed and the amount of goods to be delivered).
- Planning a stop through the operation centres in order to recharge the batteries to allow vehicles to reach the destination.
- Will allow for route calculation using different types of transportation together such as the vehicle and a wagon together to reach destination keeping into account the train schedule and the state of charge of the batteries.
2. INTRODUCTION

Nowadays, there are lot of logistic companies. Logistic companies in general have a lot of offices, vehicles, warehouses, orders with goods to be delivered, employees and customers. In order to achieve its goals it must have a system that organizes all the elements, produce efficient plan for the orders to be delivered and keep track of them.

Smart transportation system is an intelligent control platform for fleet monitoring and intermodal logistics optimization. Smart transportation system is based on electric technology. This system consists of the following elements:

- Vehicles
  Vehicles have different models; depending on the model the vehicle would be able to carry one box or more. Vehicles by default are not carrying any boxes. Vehicles in smart transportation system run on batteries. In each vehicle there will be a tablet that will be responsible of communicating with the software.

- Wagons
  Wagons have different models; depending on the model the wagon would be able to carry one box or more. Wagons by default are not carrying any boxes. In general wagons have a route that they operate on depending on a specific schedule. In each wagon there will be a tablet that will be responsible of communicating with the software.

- Boxes
  In smart transportation system boxes are treated as an entity by its own. Boxes will be carrying the goods that need to be delivered from one point to another. These boxes will be assigned to either a vehicle or a wagon in order to reach the destination.

- Batteries
  Batteries in smart transportation system are the fuel that will be given to vehicles in order to do the assigned route.

- Operation centers
  Operation centers is the place where all the vehicles will be parking and boxes will be kept in. In addition it is the place where all the batteries will be charged.

- Distribution centers
  Distribution centers is the place where all the goods will be located and is the place where an order will start.

- Train stations
  Train stations in smart transportation system represent the stops that the wagons will go through from one place to another.

- Intermodal platform
Intermodal platform is a system that will be located in an operation center and train stations. Intermodal platform is a platform that consists of different lanes and each lane is responsible for transferring a box from a vehicle to a wagon and vice versa.

After getting to know all the elements we need to know how they are related to each other and how they related to the software. The software will be running from the control platform. There are two types of users a normal operator and an administrator who has all the privileges of the system. An administrator will be in charge of putting all the elements that the company has in the system i.e. vehicles, wagons, On the other hand the operator will be receiving orders from clients that has to input into the system, each order has an origin that is one of the distribution centers that are in the system and a destination. The operator will be doing his job by adding orders to the system. The system will be adjusted a schedule to run a process in order to generate work orders. Work orders are basically the template that has to be followed in order for the goods to reach the final consumer. The software will be valuating all the resources that it has and do some calculation and offer an efficient work orders to be followed. Work order will be including expected times at different points in addition to a route on the map that should be followed. On the other hand once elements are assigned the work orders and start doing assigned tasks the software will allow the operator to be monitoring the vehicles and the wagons and get the corresponding information about the attached elements and orders.

Smart transportation system has three main goals:

- Connectivity: all elements are connected and tracked
- Intelligence: Work plan optimization
- Reliability: elements monitoring

The target audience for such software will be logistic companies that are looking for innovation and a smart system.

Smart transportation system will allow end users to have more credibility with their direct customers and will be reducing costs through an optimized planning.
3. REQUIREMENTS

In this chapter we will be explaining the system requirements. System requirements will be divided into subsections including I-star model, functional description, non-functional description, design, detail functionality and use cases.

I-STAR MODELLING:

Strategic Rationale model of the I-star modelling has been applied in the requirements section. This is a graphical presentation with several types of nodes and links that work together to provide a representational structure for expressing the rationales behind dependencies.

First we need to identify the stakeholders that will be represented as actors, the following table describes the smart transportation system actors:

<table>
<thead>
<tr>
<th>Actor</th>
<th>Description</th>
<th>Main Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software system</td>
<td>Is the software application itself</td>
<td>To support the functionalities of the smart transportation system</td>
</tr>
<tr>
<td>Administrator</td>
<td>The user with administrator privileges</td>
<td>To support the software with data</td>
</tr>
<tr>
<td>Operator</td>
<td>The user with operator privileges</td>
<td>To operate the software</td>
</tr>
<tr>
<td>Driver</td>
<td>The vehicle driver</td>
<td>To keep the control platform updated with his actions</td>
</tr>
<tr>
<td>Tablet</td>
<td>The tablet that will be placed inside the vehicles and the wagons</td>
<td>To keep the control platform updated with the exact situation inside the vehicle or the wagon</td>
</tr>
<tr>
<td>External services</td>
<td>The services that will support the software in doing its functionalities from outside the system</td>
<td>To support the software in achieving its goals</td>
</tr>
</tbody>
</table>

Table 1

I-star model annotations are shown in the figure below:
I-star model starts by the actors, then for each actor there must be a main goal. In order for the actor to reach his goal he must undergo some tasks. However, some tasks are formed by doing other subtasks in addition to some tasks when accomplished they achieve other goals. On the other hand some tasks depend on other task to release a resource in order for them to do their tasks.

- Functional description

Functional description will be the main tasks that the software must follow in order to achieve the main goal.

- Non-functional description

Non-functional requirements that are related to a functional requirement should be presented in the model through soft goals that leads to accomplishing the main goal.

## 2.1 FUNCTIONAL DESIGN

### 2.1.1 MANAGEMENT AND VISUALIZATION

This requirement corresponds to the management of all components in the logistics process of the smart transportation system and the display at terminals for control and management.

The requirement include the following sub-requirements:

- Real-Time Monitoring
- Vehicle Management
- Battery Management
- Boxes Management
- Wagons Management
• Operation Centers Management
• Distribution Centers Management
• Intermodal Platform Management
• Train Stations Management
• Train Routes Management

Each of these sub-requirements will be explained in details in the following subsections.

2.1.1.1 REAL-TIME MONITORING

I-STAR MODEL
**FUNCTIONALITY DESCRIPTION**

The system must be able to monitor on a map, in real time, the position of any component Smart transportation system logistics.

This requirement has to comply:

1) Representation of the cartography base on the map.

2) Representation of real-time location of electric vehicles (LCV) in circulation.
   a) Collect data
   b) Process data
c) Save data

3) Representation of real-time location of rail wagons in circulation.
   a) Collect data
   b) Process data
   c) Save data

4) Graphical visualization of the entire elements even if not in circulation (Distribution centers, Operation centers, etc).
   a) DB data access
   b) Process data

5) View of the routes assigned to each work plan on the map.
   a) DB data access
   b) Process data

Optionally and as improvement it can meet:

- Graphical representation for those special vehicles or wagons that have an incidence or alarm.
- Filtering of the vehicles shown on the map.

NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an operator or admin in order to view the real-time monitoring.

Security:

6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:

7) User role is verified
Reliability:

8) To receive wagons and vehicles data in correct format every x seconds

Usability:

9) Representation of a map with all elements in circulation by default

DETAILS FUNCTIONALITY

In order to access the real time monitoring functionality, the user will be sent a login form that he or she has to fill. Accordingly the system will check the identity of the user, if the user is verified and has a role of either an admin or operator he or she will be given access to the application. Otherwise access will be denied.

The web tool will be integrated into your dashboard area where the map appears and mapping elements based on its real-time location. Those are:

- Electrical Vehicles (LCV)
- Rail Wagons
- Operation Centers
- Distribution Centers
- Train stations
- Working Routes

In order to locate vehicles and wagons on the map, they will be continuously sending a special formatted JSON object with all the details through the tablet this will be including their GPS coordinates to the control center. These coordinates will be interpreted by the system, which graphically displays the position of the elements on the map.

The base cartography to be used will be obtained via the website of any of the existing free providers, whether OpenStreetMaps (OSM), Google Maps or other. The choice and form of layers will be implemented in Technical Design phase.

USE CASES

Graphic representation in real-time:

1. The user will go to the web application URL.
2. Then click on “Sign in” on the main screen. The following screen is displayed, the user must introduce his username and password and click on “Sign in” in order to proceed to the home page of the application.

3. By default the same area of the map will always appear with the representation of elements in circulation at that specific time.
4. The system will be automatically updating the position of the vehicles and wagons when new coordinates data are received.

2.1.1.2 VEHICLE MANAGEMENT

I-STAR MODEL

*Vehicles’ model:*
**FUNCTIONALITY DESCRIPTION**

The system must be able to have an enterprise managing system for vehicles. However, in order to be able to have a managing system for vehicles there is a sub requirement that the user should be able to add, modify and delete vehicles models, which will be the base for creating a new vehicle.

**The vehicle model requirement has to comply:**

1) To manage vehicle models
   a) Representation of a list with all the vehicle models
      i) DB data access
      ii) Provide a list with existing data
   b) To add a new vehicle model
i) Provide a form  
ii) Process data  
iii) Save data

c) To modify an existing model
   i) DB data access  
   ii) Provide a form with existing data  
   iii) Process data  
   iv) Save data

d) To delete an existing model
   i) DB data access

The vehicle requirement has to comply:

2) Representation of a list with all the vehicles
   a) DB data access  
   b) Provide a list with existing data

3) To add a new vehicle
   a) DB data access to get existing models  
   b) Provide a form with the models  
   c) Process data  
   d) Save data

4) To modify a vehicle
   a) DB data access  
   b) Provide a form with existing data  
   c) Process data  
   d) Save data

5) To delete an existing vehicle
   a) DB data access

6) To access historical data of the vehicle
a) DB data access
b) Provide a table with existing data

NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an admin in order to be able to manage vehicles.

Security:
7) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
8) User role is verified

Reliability:
9) The administrator has to provide the system with vehicles and models real data
   a) New model specifications
   b) Model updated specifications
   c) Model to be deleted
   d) New vehicle specifications
   e) Vehicle updated specifications
   f) Vehicle to be deleted

Usability:
10) User friendly forms for users to input the information
In order to Access the vehicle management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the vehicles.

The tool will be showing by default the list of vehicles that belong to the company. By then the user will be able to access the vehicle models where he will be able to create a new model, update an existing one or delete an existing model. However, the vehicle model is the basis that will serve in creating a new vehicle. By then the user will be able to create a new vehicle, update an existing one or delete a vehicle.

There will be a history table where there is a full history of each truck in the company. The table will be showing the date and time when the truck was working and the times that it was not. In addition to the location attached to that specific time and the state of the storage i.e. if the truck was storing goods or was empty.

---

**USE CASES**

**Access to vehicles:**

1. The user will go to the web application URL and login.

![Login Form]

2. By default the same area of the map will always appear with the representation of vehicles in circulation at the time.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.
4. By clicking on “vehicles”, a new page with the list of vehicles will appear.

The list of Vehicles includes the following information:
- ID number.
- Plate number.
- Vehicle Model.

**Add a new Vehicle:**
1. Starting from the list of vehicles and clicking on "New vehicle" button, a new page will open.
   - The user should fill the plate number and choose a model. Then click on “Submit” to add the vehicle to the list.
   - A new page with a message confirming that the information was successfully added will appear.
   - Then by clicking on “Back to list of vehicles” and the newly added vehicle will be shown among the list of vehicles with an ID number automatically assigned.

**Edit a vehicle:**
1. Starting from the list of vehicles and clicking on "Edit" button next to a vehicle, a new page will open.
2. The user will be able to change the plate number and the vehicle model. Then click on “Submit” to confirm the changes.
3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of vehicles” the user will be directed to the list of vehicles.

Delete a vehicle:

1. Starting from the list of vehicles and clicking on “Delete” button next to a vehicle.
2. A message will be shown confirming that the vehicle was successfully deleted, and then by clicking on “Back to list of vehicles” the user will be directed to the list of vehicles.

History of vehicles:

1. Starting from the list of vehicles and clicking on “History” button next to a vehicle, a new page will open.
2. The user will be able to see a report with:
   a. Location points of the vehicles along the route
   b. The id of the vehicle
   c. Time
   d. Status
   e. Location
   f. Comments
3. By Clicking on “Back to vehicles”, the user will be directed to the list of vehicles.
**Manage vehicle models:**

1. Starting from the list of vehicles and clicking on "Manage vehicle models", a new page will open.
2. The new page displayed will include all the information regarding the different vehicle models available. The information includes:
   a. ID number.
   b. Vehicle model.
   c. Vehicle year.
   d. Vehicle power.
   e. Maximum speed.
   f. Weight.
   g. Type of battery.
   h. Maximum number of batteries that can be attached.
   i. Type of box.
   j. Maximum number of boxes.
   k. Maximum weight of boxes.
Add a new vehicle model:

1. Starting from the list of vehicle models and clicking on “New vehicle model”, a new page will open.
2. The user should introduce all the information related to the model, year, power, maximum speed, weight, type of battery, maximum number of batteries, type of box, boxes to attach, maximum weight of boxes and click on “Submit” to add the new vehicle model to the list.

3. A message will be shown confirming that the information has been successfully added.
4. Then by clicking on “Back to list of vehicle models” and the newly added vehicle model will be shown among the list of vehicle models with an ID number automatically assigned.

Edit a vehicle model:

1. Starting from the list of vehicle models and clicking on “Edit” next to a model, a new page will open.
2. The user will be able to change the specifications. Then click on “Submit” to confirm the changes.
4. Once the changes are saved, a message is shown confirming that the information has been successfully saved, and then by clicking on “Back to list of vehicle models” the user will be directed to the list of vehicle models.

<table>
<thead>
<tr>
<th>Edit Vehicle Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
</tr>
<tr>
<td>Year</td>
</tr>
<tr>
<td>Power</td>
</tr>
<tr>
<td>Max Speed</td>
</tr>
<tr>
<td>Weight</td>
</tr>
<tr>
<td>Type of battery</td>
</tr>
<tr>
<td>Max number of batteries</td>
</tr>
<tr>
<td>Type of box</td>
</tr>
<tr>
<td>Boxes to attach</td>
</tr>
<tr>
<td>Max weight of boxes</td>
</tr>
</tbody>
</table>

**Delete a vehicle model:**

1. Starting from the list of vehicle model and clicking on “Delete” button next to a vehicle model.
2. A message will be shown confirming that the vehicle model was successfully deleted, and then by clicking on “Back to list of vehicle models” the user will be directed to the list of vehicle models.
Battery Models:

Admin

To manage the company's data and information

Log in

To manage battery models

To input new information

To make sure the models data present is correct

List of models

To provide the models with the new student's information

Find to insert new modal information

To insert the updated information

Figure 5
**Batteries:**

The system must be able to have an enterprise managing system for batteries. However, for being able to have a managing system for batteries there is a sub requirement that the user should be able to add, modify and delete battery models, which will be the base for creating a new battery.

**The battery model requirement has to comply:**

1) To manage battery models
   a) Representation of a list with all the battery models
      i) DB data access
      ii) Provide a list with existing data
   b) To add a new battery model
      i) Provide a form
      ii) Process data
iii) Save data

c) To modify an existing model
   i) DB data access
   ii) Provide a form with existing data
   iii) Process data
   iv) Save data

d) To delete an existing model
   i) DB data access

The battery requirement has to comply:

2) Representation of a list with all the batteries
   a) DB data access
   b) Provide a list with existing data

3) To add a new battery
   a) DB data access to get existing models
   b) Provide a form with the models
   c) Process data
   d) Save data

4) To modify a battery
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

5) To delete an existing battery
   a) DB data access

NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an admin in order to be able to manage batteries.
Security:
6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
7) User role is verified

Reliability:
8) The administrator has to provide the system with batteries and models real data
   a) New model specifications
   b) Model updated specifications
   c) Model to be deleted
   d) New battery specifications
   e) Battery updated specifications
   f) Battery to be deleted

Usability:
9) User friendly forms for users to input the information

DETAILS FUNCTIONALITY

In order to Access the battery management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the batteries.

The tool will be showing by default the list of batteries that belong to the company. By then the user will be able to access the battery models where he will be able to create a new model, update an existing one or delete an existing model. However, the battery model is the basis
that will serve in creating a new battery. By then the user will be able to create a new battery, update an existing one or delete a battery.

USE CASES

Access to batteries:

1. The user will go to the web application URL and login.

2. By default the same area of the map will always appear with the representation of elements in circulation at the time.

3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.
4. By clicking on “batteries”, a new page with the list of batteries will appear.

The list of Batteries includes:

- ID number.
- Battery model.
- A serial number.

**Add a new Battery:**

1. Starting from the list of batteries and clicking on “New battery” button, a new page will open.
2. The user should choose a model and fill the serial number. Then click on “Submit” to add the battery to the list.
3. A new page with a message confirming that the information was successfully added will appear.
4. Then by clicking on “Back to list of batteries” and the newly added battery will be shown among the list of batteries with an ID number automatically assigned.

**Edit a battery:**

1. Starting from the list of batteries and clicking on “Edit” button next to a battery, a new page will open.
2. The user will be able to change the battery model and the serial number. Then click on “Submit” to confirm the changes.
3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of batteries” the user will be directed to the list of batteries.

**Delete a battery:**

1. Starting from the list of batteries and clicking on “Delete” button next to a battery.
2. A message will be shown confirming that the battery was successfully deleted, and then by clicking on “Back to list of batteries” the user will be directed to the list of batteries.
**Manage battery models:**

1. Starting from the list of batteries and clicking on “Manage battery models”, a new page will open.
2. The new page displayed will include all the information regarding the different battery models available. The information includes:
   
a. ID number.
   b. Type.
   c. Year of the battery for management purposes.
   d. Capacity.
   e. Voltage.
   f. Maximum charge power.
   g. Maximum discharge power.
   h. Dimensions.
   i. Weight.

![List of Battery Models](image)

**Add a new battery model:**

1. Starting from the list of battery models and clicking on “New battery model”, a new page will open.
2. The user should introduce all the information related to the model, type, year, capacity, voltage, maximum charge power, maximum discharge power, dimensions and weight.
3. A message will be shown confirming that the information was successfully added.

4. Then by clicking on “Back to list of battery models” and the newly added battery model will be shown among the list of battery models with an ID number automatically assigned.

**Edit a battery model:**

1. Starting from the list of battery models and clicking on “Edit” next to a model, a new page will open.
2. The user will be able to change the specifications. Then click on “Submit” to confirm the changes.

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of battery models” the user will be directed to the list of battery models.
Delete a battery model:

1. Starting from the list of battery model and clicking on "Delete" button next to a battery model.
2. A message will be shown confirming that the battery model was successfully deleted, and then by clicking on "Back to list of battery models" the user will be directed to the list of battery models.

---

2.1.1.4 BOXES MANAGEMENT

I-STAR MODEL

Box models

Figure 7
The system must be able to have an enterprise managing system for boxes. However, for being able to have a managing system for boxes there is a sub requirement that the user should be able to add, modify and delete box models, which will be the base for creating a new box.

**The box model requirement has to comply:**

1) To manage box models
   a) Representation of a list with all the box models
      i) DB data access
      ii) Provide a list with existing data
   b) To add a new box model
      i) Provide a form
ii) Process data
iii) Save data
c) To modify an existing model
   i) DB data access
   ii) Provide a form with existing data
   iii) Process data
   iv) Save data
d) To delete an existing model
   i) DB data access

The box requirement has to comply:

2) Representation of a list with all the boxes
   a) DB data access
   b) Provide a list with existing data
3) To add a new box
   a) DB data access to get existing models
   b) Provide a form with the models
   c) Process data
   d) Save data
4) To modify a box
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data
5) To delete an existing box
   a) DB data access
The system must be securely protected through login and users must have the role of an admin in order to be able to manage boxes.

Security:
6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
7) User role is verified

Reliability:
8) The administrator has to provide the system with boxes and models real data
   a) New model specifications
   b) Model updated specifications
   c) Model to be deleted
   d) New box specifications
   e) Box updated specifications
   f) Box to be deleted

Usability:
9) User-friendly forms for users to input the information.

In order to Access the box management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified
and have a role of an admin he will be given an access to a button with the admin tools where he can access the boxes.

The tool will be showing by default the list of boxes that belong to the company. By then the user will be able to access the box models where he will be able to create a new model, update an existing one or delete an existing model. However, the box model is the basis that will serve in creating a new box. By then the user will be able to create a new box, update an existing one or delete a box.

---

**USE CASES**

*Access to boxes:*

1. The user will go to the web application URL and login.

   ![Please sign in](image)

2. By default the same area of the map will always appear with the representation of elements in circulation at the time.

3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.
4. By clicking on “Boxes”, a new page with the list of boxes will appear.

The list of Boxes includes this information:

- An ID number.
- The model.
- A serial number.

**Add a new Box:**

1. Starting from the list of boxes and clicking on “New box” button, a new page will open.
2. The user should choose a model and fill the serial number. Then click on “Submit” to add the box to the list.
3. A new page with a message confirming that the information was successfully added will appear.
4. Then by clicking on “Back to list of boxes” and the newly added box will be shown among the list of boxes with an ID number automatically assigned.

**Edit a box:**

1. Starting from the list of boxes and clicking on “Edit” button next to a box, a new page will open.
2. The user will be able to change the box model and the serial number. Then click on “Save” to confirm the changes.

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of boxes” the user will be directed to the list of boxes.
**Delete a box:**

1. Starting from the list of boxes and clicking on "Delete" button next to a box.
2. A message will be shown confirming that the box was successfully deleted, and then by clicking on "Back to list of boxes" the user will be directed to the list of boxes.

![Box with id=550 deleted successfully](image)

**Manage box models:**

1. Starting from the list of boxes and clicking on “Manage box models”, a new page will open.
2. The new page displayed will include all the information regarding the different box models available. The information includes:
   a. ID number.
   b. Box model.
   c. Box dimensions: including internal and external.
   d. Capacity.
   e. Weight.
   f. Maximum load.

![List of Box details](image)

**Add a new box model:**

1. Starting from the list of box models and clicking on “New box detail”, a new page will open.
2. The user should introduce all the information related to the model, external and internal dimensions, weight and maximum load, and the capacity will be automatically calculated.
3. A message will be shown confirming that the information was successfully added
4. Then by clicking on “Back to list of box details” and the newly added box model will be shown among the list of box models with an ID number automatically assigned.

**Edit a box model:**

1. Starting from the list of box models and clicking on “Edit” next to a model, a new page will open.
2. The user will be able to change the specifications. Then click on “Submit” to confirm the changes.
3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of battery details” the user will be directed to the list of box models.

```
Changes to box Detail with id=100 saved successfully
Back to list of box details
```

Delete a box model:

1. Starting from the list of box model and clicking on “Delete” button next to a box model.
2. A message will be shown confirming that the box model was successfully deleted, and then by clicking on “Back to list of box detail” the user will be directed to the list of box models.

```
BoxDetail with id=651 deleted successfully
Back to list of box detail
```

2.1.1.5 WAGONS MANAGEMENT

I-STAR MODEL

Wagon models:

![Diagram of I-Star Model with Wagon models](image)

Figure 9
**Wagons:**

**FUNCTIONALITY DESCRIPTION**

The system must be able to have an enterprise managing system for wagons. However, for being able to have a managing system for wagons there is a sub requirement that the user should be able to add, modify and delete wagon models, which will be the base for creating a new wagon.

The wagon model requirement has to comply:

1) **To manage wagon models**
   a) Representation of a list with all the wagon models
      i) DB data access
      ii) Provide a list with existing data
   b) **To add a new wagon model**
      i) Provide a form
      ii) Process data
iii) Save data

c) To modify an existing model
   i) DB data access
   ii) Provide a form with existing data
   iii) Process data
   iv) Save data

d) To delete an existing model
   i) DB data access

The wagon requirement has to comply:

2) Representation of a list with all the wagon
   a) DB data access
   b) Provide a list with existing data

3) To add a new wagon
   a) DB data access to get existing models
   b) Provide a form with the models
   c) Process data
   d) Save data

4) To modify a wagon
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

5) To delete an existing wagon
   a) DB data access

6) To access historical data of the wagon
   a) DB data access
   b) Provide a table with existing data
NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an admin in order to be able to manage wagons.

Security:
7) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
8) User role is verified

Reliability:
9) The administrator has to provide the system with wagons and models real data
   a) New model specifications
   b) Model updated specifications
   c) Model to be deleted
   d) New wagon specifications
   e) Wagon updated specifications
   f) Wagon to be deleted

Usability:
10) User-friendly forms for users to input the information.

DETAILS FUNCTIONALITY

In order to Access the wagon management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is
verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the wagons.

The tool will be showing by default the list of wagons that belong to the company. By then the user will be able to access the wagon models where he will be able to create a new model, update an existing one or delete an existing model. However, the wagon model is the basis that will serve in creating a new wagon. By then the user will be able to create a new wagon, update an existing one or delete a wagon.

There will be a history table where there is a full history of each wagon in the company. The table will be showing the date and time when the wagon was working and the times that it was not. In addition to the location attached to that specific time and the state of the storage i.e. if the wagon was storing goods or was empty.

---

**USE CASES**

**Access to wagons:**

1. The user will go to the web application URL and login.

   ![Login](image)

   - Please sign in
     - username
     - password
     - Sign in

2. By default the same area of the map will always appear with the representation of wagons in circulation at the time.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.

4. By clicking on “Wagons”, a new page with the list of wagons will appear.

The list of Wagons includes the following information:

- An ID number.
• Plate number.
• Wagon model.

**Add a new Wagon:**

1. Starting from the list of wagons and clicking on "New wagon" button, a new page will open.

![New Wagon](image)

2. The user should fill the plate number and choose a model. Then click on “Submit” to add the wagon to the list.
3. A new page with a message confirming that the information was successfully added will appear.
4. Then by clicking on “Back to list of wagons” and the newly added wagon will be shown among the list of wagons with an ID number automatically assigned.

**Edit a wagon:**

1. Starting from the list of wagons and clicking on "Edit" button next to a wagon, a new page will open.
2. The user will be able to change the plate number and the wagon model. Then click on “Submit” to confirm the changes.

![Edit Wagon](image)

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of wagons” the user will be directed to the list of wagons.
Delete a wagon:

1. Starting from the list of wagons and clicking on "Delete" button next to a wagon.
2. A message will be shown confirming that the wagon was successfully deleted, and then by clicking on “Back to list of wagons” the user will be directed to the list of wagons.

History of wagons:

1. Starting from the list of wagons and clicking on “History” button next to a wagon, a new page will open with a report of some location points of the wagons along the route. The structure and the information shown are similar to those explained in the section of “History of vehicles”.
2. The user will be able to see a report with:
   a. Location points of the vehicles along the route
   b. The Id of the vehicle
   c. Time
   d. Status
   e. Location
   f. Comments

Manage wagon models:

1. Starting from the list of wagons and clicking on "Manage wagon models", a new page will open.
2. The new page displayed will include all the information regarding the different wagon models available. The information includes:
   a. ID number.
   b. Wagon model.
   c. Wagon year.
   d. Wagon power.
   e. Maximum speed.
Add a new wagon model:

1. Starting from the list of wagon models and clicking on “New wagon model”, a new page will open.
2. The user should introduce all the information related to the model, year, power, maximum speed, type of battery, maximum number of batteries, type of box, boxes to attach, maximum weight of boxes, and maximum regeneration power.

A message will be shown confirming that the information was successfully added

Then by clicking on “Back to list of wagon models” and the newly added wagon model will be shown among the list of wagon models with an ID number automatically assigned.

Edit a wagon model:

1. Starting from the list of wagon models and clicking on “Edit” next to a model, a new page will open.
2. The user will be able to change the specifications. Then click on “Submit” to confirm the changes.

<table>
<thead>
<tr>
<th>Model</th>
<th>Model 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>2016</td>
</tr>
<tr>
<td>Power</td>
<td>160</td>
</tr>
<tr>
<td>Max Speed</td>
<td>180</td>
</tr>
<tr>
<td>Type of battery</td>
<td>Type of battery</td>
</tr>
<tr>
<td>Max number of batteries</td>
<td>4</td>
</tr>
<tr>
<td>Type of box</td>
<td>Type of box</td>
</tr>
<tr>
<td>Boxes to attach</td>
<td>1</td>
</tr>
<tr>
<td>Max weight of boxes</td>
<td>Max weight of boxes</td>
</tr>
<tr>
<td>Max regeneration power</td>
<td>Max regeneration power</td>
</tr>
</tbody>
</table>

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of wagon models” the user will be directed to the list of wagon models.

<table>
<thead>
<tr>
<th>Changes to wagon Model with id=100 saved successfully</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back to list of wagon models</td>
</tr>
</tbody>
</table>

**Delete a wagon model:**

1. Starting from the list of wagon model and clicking on "Delete" button next to a wagon model.
2. A message will be shown confirming that the wagon model was successfully deleted, and then by clicking on “Back to list of wagon models” the user will be directed to the list of wagon models.

<table>
<thead>
<tr>
<th>Wagon Model with id=100 deleted successfully</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back to list of wagon models</td>
</tr>
</tbody>
</table>
2.1.1.6 OPERATION CENTERS MANAGEMENT

I-STAR MODEL

**FUNCTIONALITY DESCRIPTION**

The system must be able to have an enterprise managing system for operation centers.

The operation center requirement has to comply:

1) Representation of a list with all the operation centers
   a) DB data access
   b) Provide a list with existing data

2) Representation of all operation centers on the map
a) DB data access
b) Process coordinates on the map

3) To add a new operation center
   a) Provide a form with a map
   b) Process data
   c) Save data

4) To modify an operation center
   a) DB data access
   b) Provide a form with existing data
   c) Process data
d) Save data

5) To delete an existing operation center
   a) DB data access

NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an admin in order to be able to manage operation center or have a role of an operator and be able to have a map view with operation centers on.

Security:

6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:

7) User role is verified

Reliability:
8) The administrator has to provide the system with operation centers real data
   a) New operation center specifications
   b) Operation center updated specifications
   c) Operation center to be deleted

Usability:
9) User-friendly forms for users to input the information
10) Map presentation of the operation centers

DETAILS FUNCTIONALITY

The tool will be having a managing system for the operation centers. Operation centers are
the places where you can park the vehicles and boxes in addition to charging the batteries.

In order to Access the operation centers management functionality, the user will be sent a
login form that he has to fill. Accordingly the system will check the identity of the user, if the
user is verified and have a role of an admin he will be given an access to a button with the
admin tools where he can access the operation centers. Otherwise, if the use has an operator
role he will have access to the map where the operation centers will be on the map.

USE CASES

Access to operation centers:

1. The user will go to the web application URL and login.

2. By default the same area of the map will always appear with the representation of the
   icons of operation centers.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.

4. By clicking on “Operation centers”, a new page with the list of operation centers will appear.
The list of operation centers includes the following information:

- ID number.
- Name of the center.
- A registration number.
- The address and the city where it is located.
- Recharge centers attached, where the batteries are loaded and distributed to the vehicles.
- Battery capacity and racks.
- Parking slots.

**Add a new operation center:**

1. Starting from the list of operation centers and clicking on “New operation center” button, a new page will open.
2. The user should fill the following information:
   a. Name of the operation center.
   b. Registration number.
   c. City.
   d. Address: this field is automatically filled using the magnifying glass icon on the map, click on it and a new field is opened to enter the address, and then select the correct address from the dropdown menu.
   e. Recharge centers attached.
   f. Battery capacity.
   g. Battery racks.
   h. Parking slots.
3. Then click on “Submit” to add to the operation centers list.
4. A new page with a message confirming that the information was successfully added will appear.
5. Then by clicking on “Back to list of operation centers” and the newly added operation center will be shown among the list of operation centers with an ID number automatically assigned.

**Edit an operation center:**

1. Starting from the list of operation centers and clicking on “Edit” button next to an operation center, a new page will open.
2. The user will be able to change the data. Then click on “Submit” to confirm the changes.
3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of operation centers” the user will be directed to the list of operation centers.

**Changes to Operation Center with id=2650 saved successfully**

**Back to list of operation centers**

**Delete an operation center:**

1. Starting from the list of operation centers and clicking on “Delete” button next to an operation center.
2. A message will be shown confirming that the operation center was successfully deleted, and then by clicking on “Back to list of operation centers” the user will be directed to the list of operation centers.

**Operation Center with id=2650 deleted successfully**

**Back to list of operation centers**

---

2.1.1.7 DISTRIBUTION CENTERS MANAGEMENT

---

I-STAR MODEL
The system must be able to have an enterprise managing system for distribution centers.

The distribution center requirement has to comply:

1) Representation of a list with all the distribution centers
   a) DB data access
   b) Provide a list with existing data

2) Representation of all distribution centers on the map
   a) DB data access
   b) Process coordinates on the map

3) To add a new distribution center
a) Provide a form with a map
b) Process data
c) Save data

4) To modify a distribution center
   a) DB data access
   b) Provide a form with existing data
c) Process data
d) Save data

5) To delete an existing distribution center
   a) DB data access

---

**NON FUNCTIONAL DESCRIPTION**

The system must be securely protected through login and users must have the role of an admin in order to be able to manage distribution centers or have a role of an operator and be able to have a map view with distribution centers on.

**Security:**

6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

**Privacy:**

7) User role is verified

**Reliability:**

8) The administrator has to provide the system with distribution centers real data
   a) New distribution center specifications
   b) Distribution center updated specifications
c) Distribution center to be deleted

Usability:

9) User-friendly forms for users to input the information
10) Map presentation of the distribution centers

DETAILS FUNCTIONALITY

The tool will be having a managing system for the distribution centers. Distribution centers are the places where vehicles pick the goods for distribution.

In order to Access the distribution centers management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the distribution centers. Otherwise, if the user has an operator role he will have access to the map where the distribution centers will be on the map.

USE CASES

Access to distribution centers:

1. The user will go to the web application URL and login.

2. By default the same area of the map will always appear with the representation of the icons of distribution centers.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.

4. By clicking on “Distribution centers”, a new page with the list of distribution centers will appear.
The list of distribution centers includes the following information:

- An ID number.
- Name
- Address.

Add a new distribution center:

1. Starting from the list of distribution centers and clicking on “New distribution center” button, a new page will open.

2. The user has to introduce the name of the distribution center and indicate the address using the map, to select the address use the magnifying glass icon on the map, click on it and a new field is open to enter the address, and then select the correct address from the dropdown menu.
3. Then click on “Submit” to add to the distribution centers list.
4. A new page with a message confirming that the information was successfully added will appear.
5. Then by clicking on “Back to list of distribution centers” and the newly added operation center will be shown among the list of distribution centers with an ID number automatically assigned.
**Edit a distribution center:**

1. Starting from the list of distribution centers and clicking on “Edit” button next to a distribution center, a new page will open.
2. The user will be able to change the data. Then click on “Submit” to confirm the changes.

![Edit distribution center](image)

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of distribution centers” the user will be directed to the list of distribution centers.

![Changes to distribution center](image)

**Delete a distribution center:**

1. Starting from the list of distribution centers and clicking on “Delete” button next to a distribution center.
2. A message will be shown confirming that the distribution center was successfully deleted, and then by clicking on “Back to list of distribution centers” the user will be directed to the list of distribution centers.

![Distribution Center](image)
Intermodal platform models:

![Diagram of intermodal platform models](image)

Figure 13
**Intermodal platforms:**

The system must be able to have an enterprise managing system for intermodal platforms. However, for being able to have a managing system for intermodal platforms there is a sub requirement that the user should be able to add, modify and delete intermodal platform models, which will be the base for creating a new intermodal platforms.

The intermodal platforms model requirement has to comply:

1) To manage intermodal platform models
   a) Representation of a list with all the intermodal platform models
      i) DB data access
      ii) Provide a list with existing data
b) To add a new intermodal platform model
   i) Provide a form
   ii) Process data
   iii) Save data

c) To modify an existing model
   i) DB data access
   ii) Provide a form with existing data
   iii) Process data
   iv) Save data

d) To delete an existing model
   i) DB data access

The intermodal platforms requirement has to comply:

2) Representation of a list with all the intermodal platforms
   a) DB data access
   b) Provide a list with existing data

3) To add a new intermodal platform
   a) DB data access to get existing models and train stations
   b) Provide a form with the models
   c) Process data
   d) Save data

4) To modify an intermodal platform
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

5) To delete an existing intermodal platform
   a) DB data access
The system must be securely protected through login and users must have the role of an admin in order to be able to manage intermodal platforms.

Security:
6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
7) User role is verified

Reliability:
8) The administrator has to provide the system with intermodal platforms and models real data
   a) New model specifications
   b) Model updated specifications
   c) Model to be deleted
   d) New intermodal platform specifications
   e) Intermodal platform updated specifications
   f) Intermodal platform to be deleted

Usability:
9) User-friendly forms for users to input the information.

In order to Access the intermodal platform management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the
user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the intermodal platforms.

The tool will be showing by default the list of intermodal platforms that belong to the company. By then the user will be able to access the intermodal platform models where he will be able to create a new model, update an existing one or delete an existing model. However, the intermodal platform model is the basis that will serve in creating a new intermodal platform. By then the user will be able to create a new intermodal platform, update an existing one or delete an intermodal platform.

USE CASES

Access to intermodal platforms:

1. The user will go to the web application URL and login.

   ![Sign in form](image)

2. By default the same area of the map will always appear with the representation of elements in circulation at the time.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.

4. By clicking on “Intermodal platforms”, a new page with the list of intermodal platforms will appear.

The list of Intermodal platforms includes this information:

- ID number
- Intermodal platform model.
- Train station where the platform is located.
- Year
Add a new intermodal platform:

1. Starting from the list of intermodal platforms and clicking on “New intermodal platform” button, a new page will open.

![New Intermodal Platform](image)

2. The user should choose a model and choose the train station where the intermodal platform will be allocated and fill in the year. Then click on “Submit” to add the intermodal platform to the list.
3. A new page with a message confirming that the information was successfully added will appear.
4. Then by clicking on “Back to list of intermodal platforms” and the newly added intermodal platform will be shown among the list of intermodal platforms with an ID number automatically assigned.

Edit an intermodal platform:

1. Starting from the list of intermodal platforms and clicking on “Edit” button next to an intermodal platform, a new page will open.
2. The user will be able to change the model, train station or update the year. Then click on “Submit” to confirm the changes.

![Edit Intermodal platform](image)

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of intermodal platforms” the user will be directed to the list of intermodal platforms.
Delete an intermodal platform:

1. Starting from the list of intermodal platforms and clicking on “Delete” button next to an intermodal platform.
2. A message will be shown confirming that the intermodal platform was successfully deleted, and then by clicking on “Back to list of list of intermodal platform” the user will be directed to the list of intermodal platforms.

Manage intermodal platform models:

1. Starting from the list of intermodal platforms and clicking on “Manage intermodal platform models”, a new page will open.
2. The new page displayed will include all the information regarding the different intermodal platform models available. The information includes:
   a. ID number.
   b. Maximum number of batteries.
   c. Number of lines.
   d. Number of boxes per line.

Add a new intermodal platform model:

1. Starting from the list of intermodal platform models and clicking on “New intermodal platform model”, a new page will open.
2. The user should introduce all the information related to the model, maximum number of batteries, number of lines and number of boxes per line.
3. A message will be shown confirming that the information was successfully added.
4. Then by clicking on “Back to list of intermodal platform models” and the newly added intermodal platform model will be shown among the list of intermodal platform models with an ID number automatically assigned.

**Edit an intermodal platform model:**

1. Starting from the list of intermodal platform models and clicking on “Edit” next to a model, a new page will open.
2. The user will be able to change the model, maximum number of batteries, number of lines and number of boxes per line. Then click on “Submit” to confirm the changes. Then click on “Submit” to confirm the changes.

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of intermodal platform models” the user will be directed to the list of intermodal platform models.

**Delete an intermodal platform model:**

1. Starting from the list of intermodal platform model and clicking on “Delete” button next to an intermodal platform model.
2. A message will be shown confirming that the intermodal platform model was successfully deleted, and then by clicking on “Back to list of intermodal platform models” the user will be directed to the list of intermodal platform models.

![IntermodalPlatform model with id=150 deleted successfully](image)

Back to list of intermodal platform models

## 2.1.1.9 TRAIN STATIONS MANAGEMENT

### I-STAR MODEL

![Diagram of I-STAR Model](image)

**Figure 15**

**FUNCTIONALITY DESCRIPTION**

The system must be able to have an enterprise managing system for train stations.

The train station requirement has to comply:
1) Representation of a list with all the train stations
   a) DB data access
   b) Provide a list with existing data

2) Representation of all train stations on the map
   a) DB data access
   b) Process coordinates on the map

3) To add a new train station
   a) Provide a form with a map
   b) Process data
   c) Save data

4) To modify a train station
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

5) To delete an existing train station
   a) DB data access

-----------------------------------------------------------------------------
NON FUNCTIONAL DESCRIPTION
-----------------------------------------------------------------------------

The system must be securely protected through login and users must have the role of an
admin in order to be able to manage train stations or have a role of an operator and be able to
have a map view with train stations on.

Security:

6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access
Privacy:
7) User role is verified

Reliability:
8) The administrator has to provide the system with train stations real data
   a) New train station specifications
   b) Train station updated specifications
   c) Train station to be deleted

Usability:
9) User-friendly forms for users to input the information
10) Map presentation of the train stations

DETAILS FUNCTIONALITY

The tool will be having a managing system for the train stations. Train stations are the places where wagons will stop.

In order to Access the train stations management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the train stations. Otherwise, if the user has an operator role he will have access to the map where the train station’s icons will be on the map.

USE CASES

Access to train stations:
1. The user will go to the web application URL and login.
2. By default the same area of the map will always appear with the representation of the icons of train stations.

3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.
4. By clicking on “Train stations”, a new page with the list of train stations will appear.

<table>
<thead>
<tr>
<th>List of Train stations</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>500</td>
</tr>
<tr>
<td>501</td>
</tr>
<tr>
<td>502</td>
</tr>
</tbody>
</table>

The list of Train Stations displays the following information:

a. ID number.
b. Name of the train station.
c. Address of the train station.

**Add a new train station:**

1. Starting from the list of train stations and clicking on "New train station" button, a new page will open.

<table>
<thead>
<tr>
<th>New Train Station</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Train station Name</td>
</tr>
</tbody>
</table>

2. The user has to introduce the following:
a. Name of the train station.
b. Address: this field is automatically filled using the magnifying glass icon on the map, click on it and a new field is opened to enter the address, and then select the correct address from the dropdown menu.

3. Then click on “Submit” to add to the train stations list.
4. A new page with a message confirming that the information was successfully added will appear.
5. Then by clicking on “Back to list of train stations” and the newly added train station will be shown among the list of train stations with an ID number automatically assigned.

**Edit a train station:**

1. Starting from the list of train stations and clicking on “Edit” button next to a train station, a new page will open.
2. The user will be able to change the train station name and address. Then click on “Submit” to confirm the changes.

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of train stations” the user will be directed to the list of train stations.
**Delete a train station:**

1. Starting from the list of train stations and clicking on “Delete” button next to a train station.
2. A message will be shown confirming that the train station was successfully deleted, and then by clicking on "Back to list of train stations" the user will be directed to the list of train stations.

---

**Train routes:**

**Figure 16**
**Train route schedules:**

The system must be able to have an enterprise managing system for train routes. However, for each train route there will be a sub requirement, which is the schedule that belongs to each route.

The train route requirement has to comply:

1. Representation of a list with all the train routes
   a) DB data access
   b) Provide a list with existing data
2. Representation of all train routes on the map
   a) DB data access
   b) Process data to be shown on the map
3) To add a new train route
   a) Provide a form
   b) Process data
   c) Save data

4) To modify an train route
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

5) To delete an existing train route
   a) DB data access

The train route schedule requirement has to comply:

6) To manage train routes
   a) Representation of a list with all the train route schedules
      i) DB data access
      ii) Provide a list with existing data
   b) To add a new train route schedule
      i) DB data access to get existing train routes and train stations
      ii) Provide a form with the train routes and train stations
      iii) Process data
      iv) Save data
   c) To modify a train route schedule
      i) DB data access
      ii) Provide a form with existing data
      iii) Process data
      iv) Save data
   d) To delete an existing train route schedule
The system must be securely protected through login and users must have the role of an admin in order to be able to manage train routes.

Security:

7) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:

8) User role is verified

Reliability:

9) The administrator has to provide the system with train routes and schedules real data
   a) New train route specifications
   b) Train route updated specifications
   c) Train route to be deleted
   d) New train route schedule specifications
   e) Train route schedule updated specifications
   f) Train route schedule to be deleted

Usability:

10) User-friendly forms for users to input the information

11) Map presentation of the train routes
In order to Access the train route management functionality, the user will be sent a login form that he has to fill. Accordingly the system will check the identity of the user, if the user is verified and have a role of an admin he will be given an access to a button with the admin tools where he can access the train routes, otherwise if he has the role of operator he will be having access to the main map where the routes will be shown on.

In order to be able to see the routes on the map, there will be a sub requirement, which is the train route schedule. Train route schedule will be basically a schedule for each route including the stop stations with their corresponding times.

**USE CASES**

**Access to train routes:**

1. The user will go to the web application URL and login.

2. By default the same area of the map will always appear with the representation of the train routes.
3. By clicking on the “Admin tools” from the navigation bar, a new page will be opened with different icons.

4. By clicking on “Train routes”, a new page with the list of train routes will appear.
The list of train routes includes the following information:

a. An ID code.
b. A Route code
c. The company.
d. The direction of the train: for example if it is from point A to B it will be shown as 1, while if it is from B to A it will be shown as -1.

**Add a new train route:**

1. Starting from the list of train routes and clicking on “New train route” button, a new page will open.

2. The user should set the Route Code, the company and select from a dropdown menu the direction of the train (Ascending or descending). Then click on “Submit” to add the new train route to the list.

3. A new page with a message confirming that the information was successfully added will appear.
4. Then by clicking on “Back to list of train routes” and the newly added train route will be shown among the list of train routes with an ID number automatically assigned.

**Edit a train route:**

1. Starting from the list of train routes and clicking on “Edit” button next to a train route, a new page will open.
2. The user will be able to change the route code, the company and also change the direction of the train. Then click on “Save” to confirm the changes.

![Edit train route](image)

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of train routes” the user will be directed to the list of train routes.

![Changes to Train Route](image)

**Delete a train route:**

1. Starting from the list of train routes and clicking on “Delete” button next to a train route.
2. A message will be shown confirming that the train route was successfully deleted, and then by clicking on “Back to list of list of train routes” the user will be directed to the list of train routes.

![Train Route deleted successfully](image)
**Manage train route schedules:**

1. Starting from the list of train routes and clicking on “Manage train route schedules” a new page will open.
2. The new page displayed will include all the information regarding the different train route schedules available. The information includes:
   a. ID number.
   b. Train Route.
   c. Stop number: refers to the order number of that station in regard to the route.
   d. Stop time: date and hour when the train will stop in that station.
   e. Stop station: it refers to train stations.
   f. Start and final station: are indicated with “true” or false” values if the mentioned station is the beginning or the end of the route.

---

### List of Train route schedules

<table>
<thead>
<tr>
<th>#</th>
<th>Train Route</th>
<th>Stop Number</th>
<th>Stop Time</th>
<th>Stop Station</th>
<th>Start Station</th>
<th>Final Station</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>2</td>
<td>2015-07-01 00:00:00:00</td>
<td>Valencia, Fuente San Luis</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>600</td>
<td>1</td>
<td>Madrid - Atocha</td>
<td>false</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[New Train route schedule]

---

**Add a new train route schedule:**

1. Starting from the list of train route schedule and clicking on “New train route schedule”, a new page will open.
2. The user should introduce all the information related to the train route, stop number, stop time, train stop station, start and final station.

---

### New Train Route Schedule

<table>
<thead>
<tr>
<th>Train Route</th>
<th>Stop Number</th>
<th>Stop Time</th>
<th>Train Stop Station</th>
<th>Start Station</th>
<th>Final Station</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>Madrid - Atocha</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

[Submit]
3. A message will be shown confirming that the information was successfully added.
4. Then by clicking on “Back to list of train route schedules” and the newly added train route schedule will be shown among the list of train route schedules with an ID number automatically assigned.

**Edit a train route schedule:**

1. Starting from the list of train route schedules and clicking on “Edit” next to a train route schedule, a new page will open.
2. The user will be able to change the data. Then click on “Save” to confirm the changes.

![Edit train route schedule](image)

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of train route schedules” the user will be directed to the list of train route schedules.

![Changes to Train Route Schedule](image)

**Delete a train route schedule:**

1. Starting from the list of train route schedules and clicking on “Delete” button next to a train route schedule.
2. A message will be shown confirming that the train route schedule was successfully deleted, and then by clicking on “Back to list of train route schedules” the user will be directed to the list of train route schedules.
2.1.2 WORK ORDER MANAGEMENT

This requirement corresponds to the management of Work Orders in the application.

The requirement include the following sub-requirements:

- Clients Management
- Purchase Orders Management
- Work order management

2.1.2.1 CLIENTS MANAGEMENT

I-STAR MODEL

![Diagram of I-STAR Model](image)

Figure 18
The system must be able to have an enterprise managing system for the clients.

The client’s requirement has to comply:

1) Representation of a list with all the clients
   a) DB data access
   b) Provide a list with existing data

2) To add a new client
   a) Provide a form
   b) Process data
   c) Save data

3) To modify client’s data
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

4) To delete an existing client
   a) DB data access

The system must be securely protected through login and users must have the role of an operator in order to be able to manage clients.

Security:

5) User is verified
   a) Prepare the login form
   b) Process login credentials
i) DB data access

Privacy:
6) User role is verified

Reliability:
7) The operator has to provide the system with client’s real data
   a) New client’s data
   b) Client’s updated data
   c) Client to be deleted

Usability:
8) User-friendly forms for users to input the information

DETAILS FUNCTIONALITY

The tool will be having a managing system for the clients. Purchase orders will be done in the system for a particular client.

The tool will allow the operator to be able to add to the system new clients in addition to modifying and removing existing ones.

In addition the tool will be able to show the purchase orders filtering them by customer name for example.

USE CASES

Access to clients:
1. The user will go to the web application URL and login.
2. By default the same area of the map will always appear with the representation of the elements in circulation.

3. By clicking on the customers’ dropdown box, the operator will be able to see the customer list or add the information of a new one.

4. By clicking on the “List of customers” the user will be redirected to the list of customer's page.
This list includes all the information related to the customer:

   a. Id number.
   b. Name
   c. Address
   d. Telephone number
   e. Email

Add a new client:

1. Starting from the list of clients and clicking on “New customer” button, a new page will open or an alternative would be from the home page and clicking on “New customer” from the dropdown box.

2. The user has to introduce the following:
   a. Customer name.
   b. Address.
   c. Telephone number.
   d. Email
3. Then click on “Submit” to add to the clients’ list.
4. A new page with a message confirming that the information was successfully added will appear.
5. Then by clicking on “Back to list of customers” and the newly added client will be shown among the list of customers with an ID number automatically assigned.

**Edit a client:**
1. Starting from the list of clients and clicking on “Edit” button next to a client, a new page will open.
2. The user will be able to change client’s data. Then click on “Submit” to confirm the changes.

![Edit Customer]

3. Once the changes are saved, a message is shown confirming that the information was successfully saved, and then by clicking on “Back to list of customers” the user will be directed to the list of clients.

![Changes to customer with id=350 saved successfully]

**Delete a client:**
1. Starting from the list of customers and clicking on “Delete” button next to a customer.
2. A message will be shown confirming that the customer was successfully deleted, and then by clicking on “Back to list of customers” the user will be directed to the list of customers.

![Customer with id=350 deleted successfully]
2.1.2.2 PURCHASE ORDERS MANAGEMENT

I-STAR MODEL

The system must be able to have an enterprise managing system for the purchase orders. In addition to showing the state of the purchase order i.e. if it has already a planned route or not and if it has a planned route the user would be able to the planned route.

The purchase orders requirement has to comply:

1) Representation of a list with all the purchase orders including the status
   a) DB data access
   b) Provide a list with existing data
2) To add a new purchase order
   a) Provide a form
   b) Process data
   c) Save data

3) To modify purchase order’s data
   a) DB data access
   b) Provide a form with existing data
   c) Process data
   d) Save data

4) To delete an existing purchase order if it is not planned
   a) DB data access

5) To show planned route
   a) DB data access
   b) Process data on map

NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an operator in order to be able to manage purchase orders.

Security:

6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:

7) User role is verified

Reliability:
8) The operator has to provide the system with purchase order's information

   a) New purchase order's data
   b) Purchase order's updated data
   c) Purchase order to be deleted

Usability:
9) User-friendly forms for users to input the information
10) Map with planned route

DETAILS FUNCTIONALITY

The tool will be having a managing system for the purchase orders. Purchase orders will be done in the system for a particular customer.

The tool will allow the user with role operator to be able to add to the system new purchase order in addition to modifying and removing existing ones if it haven't been assigned a work order plan.

In addition the tool will be able to show if the system has already planned a work order for each particular purchase order and if so the operator will be able to access the planned route.

The operator will be also able to track a purchase order in order to know if it is already delivered or not.

USE CASES

**Access to purchase orders:**

1. The user will go to the web application URL and login.
2. By default the same area of the map will always appear with the representation of the elements in circulation.

3. By clicking on the orders dropdown box, the operator will be able to see the list of purchase orders or add the information of a new one.

4. By clicking on the “List of purchase orders” the user will be redirected to the list of purchase orders page.
The list of orders includes all the orders with this information:

- a. An ID number.
- b. A serial number.
- c. Customer.
- d. Origin of the box.
- e. Address destination.
- f. Comments

**Add a new purchase order:**

1. Starting from the list of orders and clicking on “New order” button, a new page will open or an alternative would be from the home page and clicking on “New purchase order” from the dropdown box.

2. The user has to introduce the following:
   - a. Customer
   - b. Serial
   - c. Origin: list of distribution centres where the order can be collected.
   - d. Destination: this field is automatically filled using the magnifying glass icon on the map, click on it and a new field is opened to enter the address.
   - e. Comments
3. Then click on “Submit”, a new screen is displayed confirming the information (ID and serial number) of the order and the fields to enter the goods information such as length, width, height and weight.

4. The user has to enter the good details, click on “Submit” to save it and add another good if is necessary. When the good is added, a “Delete” button is activated in case that is needed.

5. By clicking on “Back to order details”, the application navigates to the order details.

6. Clicking on “New order” button to access to the “New order” screen and enter a new order as was already explained before.

7. Clicking on “Orders” button to access the “List of orders” and see all the information related to all the orders saved.

**Edit a purchase order:**

1. Starting from the list of orders and clicking on “Edit” button next to an order, a new page will open.

2. The user will be able to change the serial number, the origin of the service, the destination and comments. Then click on “Save” to confirm the changes.
3. Once the changes are saved after clicking on “Save” button, the “List of orders” screen is shown.

4. On this screen it is possible to go back to the list without editing the order, by clicking on “Back to list of purchase orders” button.

5. Also it is possible to edit the goods that belong to the current order by clicking on “Edit Goods” button, the procedure to do it was explained in “Add a new order”.

108
**Delete an order:**

1. Starting from the list of orders and clicking on “Delete” button next to an order.
2. A message will be shown confirming that the purchase order was successfully deleted, and then by clicking on “Back to list of orders” the user will be directed to the list of purchase orders.

```
Purchase Order with id=8950 deleted successfully
Back to list of orders
```

**Show the planned route:**

1. Starting from the list of orders and clicking on “Show Route” button next to an order.
The system must be able to process work orders by showing the best route, consumption of energy, boxes to ship and whether it is better to use LCV, wagons or both.
This requirement has to comply:

1) To generate work orders for distribution centers  
   a) To query DB for the list of distribution centers  
   b) To generate work orders for each distribution center  
      i. To query DB for the list of purchase orders that have been added to the system and have not been assigned a work order  
      ii. To query DB for the list of vehicles that are available at the time of processing  
      iii. To query DB for the list of Boxes that are available at the time of processing  
      iv. To querying DB for the train’s schedules  
      v. Check how many boxes can be used  
      vi. Sort the goods by destination  
      vii. Assign goods to boxes  
      viii. Assign boxes to vehicles  
      ix. Calculate truck route  
      x. Calculate Energy needed  
      xi. Calculate required batteries

NON FUNCTIONAL DESCRIPTION

Reliability:
2) To receive accurate routing information  
3) To get work order assigned tasks

Usability:
4) To run automatically on x basis

Availability:
5) High availability service
Response time:

6) Low response time

DETAIL FUNCTIONALITY

The tool will be having a work order simulation. This simulation would be done on regular basis automatically without the need of any user.

The tool will be calculating the work orders for each distribution center. However, work order simulation is the way the goods will be delivered from a particular distribution center until reached by the final destination that was mentioned in the purchase order creation for a particular client.

The software will start the simulation by gathering all the information about the distribution centers that belong to the system. By then the system will get the vehicles, boxes that are available for usage in addition to the purchase orders that have been created with an origin of departure that particular distribution center. Then the tool will start by checking the exact boxes that can be used corresponding to the availability of vehicles, and then sort the goods by destination then assigning them to the boxes. Boxes will be assigned to vehicles and for each vehicle there will be assigned route and energy calculation.

The tool will be calculating the following for each work order:

- Total distance of the route
- Total time of the route
- Total energy needed for the route
- Stop points needed
- Partial timing for each stop point needed
- Calculation of optimal routes for trucks and trains, optimizing the use of energy and stop points needed.
- Inclusion of parameters to calculate the optimal route: cut roads, etc....

As an improvement, in addition, it is proposed that the application is able to generate altitude profiles and graphical time / speed.

The system must not be limited to calculate a route between two points, but must try to search for it using the best energy-saving criteria.

This is basic to the generation of a graph of road network, road and rail and allowing for the different calculation algorithms. This graph, because it is based on a mapping that can change over time, must be able to update from time to time to adapt to the latest data base existing.
To improve the calculation of these optimal routes, adjusting some parameters, which can be modified by the user are proposed. These parameters allow refining the route to use and the associated costs that will result (time and energy).

Some parameters to configure:

- Average vehicle speed
- Average train speed
- Railway inclusion or not

When the system calculates the route and the associated work plan and display the information about the estimated work plan. At a minimum must show the following information:

- Total time estimated
- Vehicles involved
- Wagons involved
- Boxes involved
- Goods that will be placed in each Box
- Energy needed
- Stop Points
- Expected arrival times at points

-----------------------------------------------
USE CASES

See Business process modeling section
2.1.3 INCIDENCES MANAGEMENT

This requirement corresponds to the management of incidences in the application.

2.1.3.1 INCIDENCES MANAGEMENT

I-STAR MODEL

**Incidence categories:**

![Incidence categories diagram](Figure 21)
Incidence codes:

Figure 22

Incidences:

Figure 23
The system must be able to have an enterprise managing system for incidences. In addition to showing the state of an incidence i.e. if the incidence is still open or has already been resolved and closed.

Incidence management consists of three components. First, the administrator user has to provide different incidence categories. Second, for each category there should be a list of incidences codes to be provided to the system. Last, the operator will be able to open an incidence using the provided categories and the specifically provided codes then when it is solved to close the incidence.

The incidences categories requirement has to comply:

1) To manage incidences categories
   a) Representation of a list with all the incidences categories
      i) DB data access
      ii) Provide a list with existing data
   b) To add a new incidence category
      i) Provide a form
      ii) Process data
      iii) Save data
   c) To modify an existing incidence category
      i) DB data access
      ii) Provide a form with existing data
      iii) Process data
      iv) Save data
   d) To delete an existing incidence category
      i) DB data access

The incidences codes requirement has to comply:

2) To manage incidences codes
a) Representation of a list with all the incidences codes
   i) DB data access
   ii) Provide a list with existing data

b) To add a new incidence code
   i) DB data access to get existing categories
   ii) Provide a form with the categories to choose from
   iii) Process data
   iv) Save data

c) To modify an incidence code
   i) DB data access
   ii) Provide a form with existing data
   iii) Process data
   iv) Save data

d) To delete an existing incidence code
   i) DB data access

The incidences requirement has to comply:

3) Representation of a list with all the incidences possible solutions depending on the category and code
   a) DB data access
   b) Provide a list with existing data

4) To open a new incidence
   a) DB data access to get existing categories and codes
   b) Provide a form with the categories and codes
   c) Process data
   d) Save data

5) To modify an incidence
   a) DB data access
   b) Provide a form with existing data
NON FUNCTIONAL DESCRIPTION

The system must be securely protected through login and users must have the role of an admin in order to be able to manage incidence categories and codes and the role of operator to manage incidences.

Security:
6) User is verified
   a) Prepare the login form
   b) Process login credentials
      i) DB data access

Privacy:
7) User role is verified

Reliability:
8) The administrator has to provide the system with incidences categories
   a) New incidence category’s data
   b) Incidence category’s updated data
   c) Incidence category to be deleted

9) The administrator has to provide the system with incidences codes
   a) New incidence code’s data
   b) Incidence code’s updated data
   c) Incidence code to be deleted

10) The operator has to provide the system with incidences
   a) New incidence's information
   b) Incidence's updated information
c) Incidence code to be closed

Usability:

11) User-friendly forms for users to input the information

12) Map with planned route

DETAILS FUNCTIONALITY

The tool will be having a managing system for the incidences. First in order to have a managing system for incidences, there should be defined categories and codes for the incidences.

The tool will allow the user with role administrator to be able to add to the system new incidence category in addition to modifying and deleting existing ones. Incidence categories will serve as the basis for creating new incidence codes. Incidence codes will be codes defining different cases of that particular category. However, both an incidence category and an incidence code will serve as the basis for opening a new incident.

The operator will basically open a new incidence first by choosing the category i.e. incidence at destination, then while choosing the incidence code there will be possible choices i.e. client not found for example. By then the operator has to fill some other fields including the date, description and the affected elements.

USE CASES

Access to incidences:

1. The user will go to the web application URL and login.

2. By default the same area of the map will always appear with the representation of the elements in circulation.
3. By clicking on the “incidents“ dropdown box, the operator will be able to see the list of incidents or add the information of a new one.

![Incidents dropdown box](image)

4. By clicking on the “List of incidents” the user will be redirected to the list of incidents page.

**Access to possible solutions:**

1. Starting from the list of incidents and clicking on an incident, a new page will open with a possible recommendation for solving an incident.
This section will explain the main non-functional requirements of the application.

### 2.2.1 HARDWARE REQUIREMENTS

Hardware requirements are needed in order to run the application. However, the exact system requirements will depend mainly upon the number of users and amount of information to be carried through the software, so we will mention the minimum requirements. Moreover, we will split hardware requirements into two groups. First group is from the server-side and the second group is from the client-side as follows:

- **Server-side:**
  - Processor: Dual Core 2GHz
  - Memory: 4 GB RAM

- **Client-side:**
  - Processor: 1GHz
  - Memory: 512 MB RAM

### 2.2.2 SOFTWARE REQUIREMENTS

Software requirements are needed in order to run the application. However, the exact system requirements will depend mainly upon the number of users and amount of information to be carried through the software, so we will mention the minimum requirements. Moreover, we will split hardware requirements into two groups. First group is from the server-side and the second group is from the client-side as follows:

- **Server-side:**
  - JDK 1.6 or later
  - Gradle 1.11+ or Maven 3.0+
  - Geo Server
  - MemCached
  - PostgreSQL with extension PostGIS

- **Client-side:**
  - **WEB Browser:**
    - Internet Explorer 9, 10
    - Mozilla Firefox
    - Chrome
    - Safari

### 2.2.3 PERFORMANCE REQUIREMENTS

Performance requirements will basically present how the smart transportation will be performing on a daily basis. However, in this part according to the application we will be splitting the performance of the application into three different groups. The first will be web browsing the application, which will include management of the entities for example new, edit and delete of a vehicle in addition to adding users, clients and purchase orders to the system. The second group will be the real-time monitoring of elements and this will be
through a Websocket that can be adjusted. Finally, the last group will be the batch processing of work orders, as it will be approaching external services and different algorithms.

- **WEB Browsing:**
  - Response time: 1-2 seconds
  - Concurrent users: 8,192 – 10,000 users according to apache Tomcat 7.0 Manual

- **Real-Time Monitoring:**
  - Response time: 10 seconds
  - Concurrent users: 8,192 – 10,000 users according to apache Tomcat 7.0 Manual

- **Batch processing:**
  - Response time: 30-60 seconds
  - Frequency: 1 Time/Day
  - Time window: 30 – 60 seconds

### 2.2.4 SECURITY REQUIREMENTS

Security requirements will be depending on the group of users that the user will lie in; different groups will have different access levels. In smart transportation system we will be having three different groups. First group will be the admin group that will be having full access to all the features of the applications including the management of users and elements. Second group will be the operators group which will only be able to monitor the elements on real-time basis, view and add purchase orders and clients to the system. Finally, the drivers a user that has only access from a tablet to a view of a map with the itineraries and incidences of the orders that he has to deliver.

### 2.2.5 INTERFACE REQUIREMENTS

Interface requirements will include content presentation using Bootstrap, which will allow users to use the application through their pad and mobile phones. In addition the smart transportation web application includes a navigation bar in all HTML pages for easy navigation within the application.

### 2.2.6 AVAILABILITY REQUIREMENTS

Availability requirements will mainly depends on the hardware provided. However, in our case since we will use single server application, so we will assume the following:

- Availability: 99%
- Down-time: 87 hours (3.5 days) every year
4. **TIME PLAN**

The following table will be explaining a time plan for the smart transportation system application implementation.

<table>
<thead>
<tr>
<th>Task</th>
<th>Start Time</th>
<th>End Time</th>
<th>Duration</th>
<th>Reasons for deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicles management</td>
<td>8-1-14</td>
<td>8-8-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Wagons management</td>
<td>8-8-14</td>
<td>8-15-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Boxes management</td>
<td>8-15-14</td>
<td>8-22-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Batteries management</td>
<td>8-22-14</td>
<td>8-29-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Distribution centers management</td>
<td>8-29-14</td>
<td>9-6-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Operation centers management</td>
<td>9-6-14</td>
<td>9-13-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Intermodal platforms management</td>
<td>9-13-14</td>
<td>9-20-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Train stations management</td>
<td>9-20-14</td>
<td>9-27-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Train routes and train route schedule management</td>
<td>9-27-14</td>
<td>10-11-14</td>
<td>2 week</td>
<td></td>
</tr>
<tr>
<td>Clients Management</td>
<td>10-11-14</td>
<td>10-18-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Purchase Orders Management</td>
<td>10-18-14</td>
<td>10-25-14</td>
<td>1 week</td>
<td></td>
</tr>
<tr>
<td>Work order management</td>
<td>10-25-14</td>
<td>11-29-14</td>
<td>5 weeks</td>
<td></td>
</tr>
<tr>
<td>Real-Time Monitoring</td>
<td>11-29-14</td>
<td>01-14-15</td>
<td>4 weeks</td>
<td>Christmas Holidays</td>
</tr>
<tr>
<td>Incidences Management</td>
<td>01-14-15</td>
<td>01-28-15</td>
<td>2 weeks</td>
<td></td>
</tr>
</tbody>
</table>

Table 2
In this part we will be explaining the business logic workflows of smart transportation system. Business logic is basically the part of the application where the workflows are applied according to the business rules.

In smart transportation system there are different workflows that have been explained in the functional requirements use cases section. These workflows include:

- Vehicles management
- Wagons management
- Boxes management
- Batteries management
- Distribution centers management
- Operation centers management
- Intermodal platforms management
- Train stations management
- Train routes and train route schedule management

The above mentioned workflows will serve as the basis for running our core main key aspects that will be explained in details in the following subsections.

### 5.1. NEW CLIENT

The following diagram shows the workflow to introduce a new client into the system:

![Workflow Diagram](Figure 24)

In order to add a new customer to the system, the user must have a role of operator or administrator. First the user has to login in to the application, then go to the new customer’s page and fill in the customer’s information, then save the customers data to the database.
5.2. NEW ORDER

The following diagram shows the workflow to introduce a new order into the system:

In order for a client to do a new order there are two options:

1. If the client has an access to the web application, then the order can be done by logging in and introducing the purchase order's information including the distribution center origin and the destination where the order will be delivered.
2. If the client does not have an account, then he can do the order by calling the operator that have an access to the application and give all the information about the order including the distribution center origin and the destination where the order will be delivered.

5.3. WORK ORDERS GENERATION

Work order generation is the core of the software. First we will start with a global overview of the work order generation process. Then show an overview of the generated results of the generated work order. Finally, a workflow with the processes that are involved in creating work orders.

The following diagram shows a global workflow of processes:
According to the workflow figure shown above the system starts by getting the list of orders in the system. Then start the generation of work orders that will be explained next, comes after the generation of the work orders there will be the process of tracking the elements on real-time basis, which will be discussed in the following section. Finally the orders are delivered to the clients.

The following workflow shows work order generation processes:

![Workflow Diagram](image)

**Figure 27**

Work order generation can be started in two different ways:

1. Automatic which is the recommended way. Automatic is basically that the system will be having a schedule to run automatically in order to generate work orders i.e. every day or every week. The software will be taking care of running all the processes mentioned above and create the final orders.
2. On demand which is generating the work orders at any moment by the operator. On demand is basically important in case of an incidence or an emergency. The software will run the above-mentioned processes at this moment and generate work orders. However, it will only consider also the available resources and it will not delete already generated work orders.

In the following subsections we will be explaining the above-mentioned processes in details.

### 3.3.1 FOR EACH DISTRIBUTION CENTER

The software will be running the workorder generation process for each distribution center individually. First it will get all the distribution centers that are in the system, then for each distribution center it will do all the processes that will be explained in details in the following sections.
3.3.2 GET THE ORDERS THAT ARE NOT ASSIGNED

In this process the software will be getting all the purchase orders that have been added to the system and are ready to be delivered and not yet assigned a work order plan.

3.3.3 GET VEHICLES THAT ARE AVAILABLE

In this process the software will be getting all the vehicles that are currently in that distribution center and are not assigned a work order i.e. they are available for being assigned a workorder.

3.3.4 GET BOXES THAT ARE AVAILABLE

In this process the software will be getting all the boxes that are currently in that distribution center and are not assigned a work order i.e. they are available for being assigned a workorder.

3.3.5 CHECK THE AVAILABILITY OF BOXES IN RELATION TO VEHICLES

In Smart transportation system the boxes are treated separately from the vehicles, so after getting the information about the available boxes and the available vehicles the software must calculate the exact number of boxes that can fit into the vehicles. This process is important in order not to assign orders and specifically goods to boxes and then find out that there are no vehicles to carry these assigned boxes.

3.3.6 GROUP THE ORDERS BY DESTINATION

Group the orders by destination process is required to make sure that the orders are being split by the zone where they will be delivered before being assigned to the boxes. This process is important to make sure that orders that have completely different destinations are split apart while goods that have close destinations are gathered together. This would help in making more efficient work order because like this the routing will be accurate and that vehicles will be heading to certain zone then deliver the goods.

In smart transportation system’s code we have an interface that is called PurchaseOrdersGrouping that defines this functionality. However, at the moment there is a single implementation for this process. This implementation is based on receiving a list of purchase orders and returns back a grouped list of purchase order by destination.

The algorithm will be explained through the following figure assuming that we have a list of purchase orders input each has a destination:
This implementation is a basic implementation that does splitting by area but according to research and investigation there are more efficient alternative implementations for this algorithm as follows:

- Grouping by postal codes: Purchase orders’ destination postal codes could allow hierarchical grouping of areas. By grouping postal codes zones will be identified and split easier.
- Spatial clustering based algorithm: spatial clustering will be having as an input a group of coordinates and will start creating clusters based on the group of coordinates that are near as shown in the following diagram. However, the problem with spatial clustering is that some points could be grouped together because according to coordinates they are close but in real life there are no roads connecting the two points. So the best solution is that spatial clustering should be used together with routing algorithms to give best results.
3.3.7 ASSIGN ORDERS TO BOXES

The software according to different criteria does assigning orders’ goods to boxes process. In this process we have different restrictions, which are the size of the box and the size of the goods. In the software there is an interface called GoodsToBoxesStrategy that defines the functionality. However, the process functionality has been implemented using the first fit algorithm. The first fit algorithm will be having as input the orders each having a group of goods each of different volume in addition to the volume of the boxes. The algorithm starts by filling the boxes with orders to prioritize that goods that belong to the same order are grouped together, then if there is an order that does not fit in a box by its own, the goods inside will be divided among different boxes.

First fit algorithm in our case is dealing only with volumes i.e. it will treat a solid as a liquid, which is very optimistic and unreal. According to some research the best algorithm for this problem would be the bin packing 3D modelling. As it will give very accurate results in addition to maximum utilization.

3.3.8 ASSIGN BOXES TO VEHICLES

The software according to different criteria does assigning boxes to vehicles process. In this process we have the restriction of maximum vehicle capacity. In smart transportation system we have two main vehicle models. First, model one that can carry maximum one box and model two that can carry maximum two boxes. In the software there is an interface called GoodsToVehiclesStrategy that defines the functionality. However, the process functionality has been implemented through an algorithm that follows two rules:

- If a purchase order has been Split in more than one box, then the priority would be to model two.
- If only one box was sufficient then model one will have the priority.

As any other algorithm in the application for improvements an algorithm can be easily changed or modified to have a better performance.

3.3.9 CALCULATE THE ROUTE

First before we start with calculating the route, we will show a global overview of a scenario that could happen exactly after assigning the boxes to the vehicles:
According to the above shown scenario, a typical work order has a relation to many work order sections. We will assume the following identifiers that will be involved in the scenario in order to explain the scenario clearly:

- Vehicle 1 identifier: a
- Vehicle 2 identifier: b
- Wagon identifier: c
- Box identifier: d
- Distribution center identifier: e
- Train station origin identifier: f
- Intermodal platform origin identifier: g
- Lane number 1
- Train station destination identifier: h
- Intermodal platform destination identifier: i
- Lane number 2
- Customer 1 identifier: j
- Customer 2 identifier: k

1. Inside the distribution center:
   a. Assign orders to box id d
   b. Assign box id d to vehicle id a
2. Work order section 1:
   a. Vehicle id a to go from distribution center id e to train station origin id f using a given route and do the following actions at destination:
      i. Park in intermodal platform origin id g in lane number 1
3. Work order section 2:
   a. Intermodal platform origin id g do the following actions for the lane number 1:
      i. Move box id d from vehicle id a to wagon id c
4. Work order section 3:
   a. Wagon id c go from train station origin id f to train station destination id h and do the following actions at destination:
      i. Park in intermodal platform destination id i lane number 2
5. Work order section 4:
   a. Intermodal platform destination id i do the following actions for the lane number 2:
      i. Move box id d from wagon id c to vehicle id b
6. Work order section 5:
   a. Vehicle id b to go from train station destination id h to customer 1 id j using a given route and do the following actions at destination:
      i. Deliver specific goods to a specific address
7. Work order section 6:
   a. Vehicle id b to go from customer 1 id j to customer 2 id k using a given route and do the following actions at destination:
      i. Deliver specific goods to a specific address
8. Work order section 7:
   a. Vehicle id b to go from customer 2 id k to distribution center id e using a given route and do the following actions at destination:
      i. Park the vehicle

In addition to the above-mentioned fields of a typical work order section there are the following:

- Expected arrival time
- Real arrival time

This will help in having an estimate on when the goods will arrive to the final customer.

As seen calculating the route is a process that there are other sub processes depending upon. After showing a scenario of different work order sections we can conclude that work order section 1, work order section 3, work order section 5, work order section 6 and work order section 7 each had a route. However, the question is when to use a vehicle and when to use a wagon and when to use both. The following figure will help us explain more the algorithm:

![Figure 31](image-url)
According to the above algorithm the system will start by calculating the route of a vehicle and the route by train, then will compare the alternatives according to the total time spent. Then it will check the following:

- Vehicle’s route is greater than 100 kms. However, this is editable and depends mainly on the batteries capacity.
- Check if the train route is viable.

Depending on the above-mentioned cases the system will either return a vehicle route or a train route.

If the system decides to return a train route this should be including the following:

- Work order section for the vehicle to go from distribution center to train station 1 and this could be divided into subsections if there is a need to pass through an operation center in order to charge batteries
- Work order section for the movement of boxes from vehicle to wagon using the intermodal platform
- Work order section for the wagon to go from train station 1 to train station 2
- Work order section for the movement of boxes from wagon to vehicle using the intermodal platform
- Work order section for the vehicle to go from train station 2 to customer 1 and this could be divided into subsections if there is a need to pass through an operation center in order to charge batteries
- Work order section for the vehicle to go from customer 1 to customer 2 and so on.

Going into details about the work order section that will be containing the train route. The software has different train routes that have been adjusted in the system and each train route is including a route through various train stations including the schedule, which is the date and time specifically. Accordingly the software will be able to calculate the expected departure time and the expected arrival times consecutively.

On the other hand if there is a section that includes vehicles or that the system decides to return a vehicle route there is an algorithm that defines how the system calculate routes for the vehicles.

In order for the system to calculate the vehicle route there is an interface called GeoService that defines the functionality. However, the process functionality has been implemented using the following free services based on OpenStreetMap cartography.

These services are given addresses and it returns back the best route.

- Nominatim (http://nominatim.openstreetmap.org): Geocoding service. Resolves and address to a latitude/longitude
- OSRM (http://project-osrm.org/): Routing service. Provides a route connecting two or more points.

Other implementation alternatives to consider are:

- A routing and geocoding service based on data in a database. Any technology following the Routing and Geocoder OGC services will suffice.
- OpenLS (Location Services): http://www.opengeospatial.org/standards/ols
An external routing service:


As seen these services are given addresses and return the best route between theses addresses. However, in order to send the addresses there is an algorithm that has been implemented in order give the service the right order of addresses that the vehicle should follow.

Since the vehicle route calculation has in consideration the following information.

- Origin point.
- List of delivery points.

The first step is deciding the order of the deliveries. A recursive algorithm searches, at each recursion level, the closer delivery.

The distances to each node are routing distances, not linear distances.

The behaviour of the algorithm is illustrated below.

Step 1: Calculate the distance to each one of the delivery points. Select the closer delivery point, and then repeat the algorithm for the rest of the delivery points.

Step 2: First delivery point is set to A. The program continues, as there are more delivery points to calculate from for the next step.
Step 3: Second delivery point is set to B. The program continues, as there are more delivery points to calculate from for the next step.

![Diagram](image1)

**Figure 34**

Step 4: Third delivery point is set to C. The program checks the number of delivery points to calculate, and as there is only 1 delivery point, automatically sets this delivery point as the next one.

![Diagram](image2)

**Figure 35**

Final step: Calculate the return trip to the distribution center.

If the route has more distance than the vehicle capacity, it is important to search for an operation center in the middle of the itinerary to change the batteries.

To find close operation centers to a route, the algorithm explained in this section can be applied, adding an operation center as a delivery point. To get close operations centers to a set of points, the following rules can be applied.

- Take into consideration only the operation centers in the area.
- Calculate the distance from the operation center to each one of the delivery points.
- Place the operation center just before its closer delivery point.
- Run the algorithm described in this section.

It is important to remark that each delivery is a new work order section, having an action of destination to deliver a specific set of goods to the customer.
### 3.3.10 CALCULATE THE ENERGY

Calculating the energy functionality has been defined in the BatteryDischargeModel interface.

The out-of-the-box implemented behaviour is defined below.

The required energy for a truck route can be calculated taking into consideration the following parameters:

- Route profile
- Route length
- The current season (winter, summer, autumn, spring)
- The route shift (night, day)

Once all the work order sections are calculated, this information can be easily obtained.

The route profile information is the percentage of each type of route in the complete route. Each type of route corresponds to a vehicle energy consumption profile. There are 5 different profiles:

- Urban_slow
- Urban_free
- Delivery
- Rural
- Motorway

For each profile, 3 factors of a 2\textsuperscript{nd} grade polynomial function are stored in the database table power_parametres. Below, are the steps to calculate the energy required for a specific route.

Given all this information, the calculation of the energy required is defined as:

$$totalEnergyWasted = \sum_{profile} energyRequired(profile)$$

Given a specific profile, the energy calculation is done applying the following formula:

- \(a, b\) and \(c\) are the polynomial factors.
- “\(distance\)” is the total distance for the selected profile.
- “\(Vehicle\ mass + Load\)” is the total weight of the truck (including boxes and goods).

\[ Wasted\ energy\ [kW\cdot h] = (a \cdot (Vehicle\ mass + Load) \cdot kg)^2 + b \cdot (Vehicle\ mass + Load) \cdot kg + c \cdot distance[km] \]

Once the totalEnergyWasted value has been calculated. We have to divide this value by 12 (the amount of kWh for a battery), in order to get the number of batteries required.

The following formula should be calculated again, adding the weight of the required batteries in the “\(Vehicle\ mass + Load\)” value.

For security reasons, a 10% reserve value is added to the final required value. Can be obtained by multiplying the final energy by 1.1.
The following diagram shows the workflow in order track orders:

As seen in the above work flow in order to track an order the user must access the application, once accessed on the map there will be a view of all the elements in circulation and by choosing an element the user will be able to find out about attached elements and the status of the orders.

In the following section we will be explaining in details the process of showing a real-time view of the elements on the map.

### 3.4.1 REAL-TIME MONITORING

The software system will be having a real time monitoring for the following elements:

- Vehicles
- Wagons

Real-time monitoring will help the users of the system in tracking the orders that have been assigned work orders as seen in the work order generation section.

Real-time monitoring is divided into two main sections:

- From element to system
- From system to user

#### 3.4.1.1 FROM ELEMENT TO SYSTEM

Real-time monitoring has been implemented using a rest service. This rest service is receiving a special formatted JSON object and process the object as shown in the following figure:
The workflow is explaining the rest service processes. However, in next subsections we will be explaining in details each process.

### 3.4.1.1.1 RECEIVE JSON OBJECT

Vehicles and wagons will be continuously sending a JSON object to the control platform through the tablet; this JSON object will be as follows:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle Id/ Wagon</td>
<td>Unique identifier of the vehicle/wagon</td>
</tr>
<tr>
<td>Driver Id (In case of vehicle)</td>
<td>Unique identifier of the driver</td>
</tr>
<tr>
<td>Route Id</td>
<td>Unique identifier of the route being followed</td>
</tr>
<tr>
<td>Current Location</td>
<td>GeoJSON formatted position of the current location.</td>
</tr>
<tr>
<td>Speed</td>
<td>Current speed of the vehicle.</td>
</tr>
<tr>
<td>Batteries information</td>
<td>Charge levels of the batteries</td>
</tr>
<tr>
<td>Status</td>
<td>Defines the status of the vehicle:</td>
</tr>
<tr>
<td></td>
<td>• Parked</td>
</tr>
<tr>
<td></td>
<td>• Loading / Unloading</td>
</tr>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td>Alerts</td>
<td>In case of alerts the vehicle has to inform the control</td>
</tr>
</tbody>
</table>
platform about that alert.

<table>
<thead>
<tr>
<th>Comments</th>
<th>Not urgent notifications that the driver wants to register.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxes information</td>
<td>Unique identifier of the boxes attached</td>
</tr>
<tr>
<td>Purchase orders list</td>
<td>List of purchase orders that have an article in the box.</td>
</tr>
<tr>
<td>1. Purchase Order Id</td>
<td>Unique identifier of the purchase order</td>
</tr>
<tr>
<td>2. Status</td>
<td>The status of that specific purchase order:</td>
</tr>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td></td>
<td>• Delivered (if the P.O. has been already delivered)</td>
</tr>
</tbody>
</table>

| Table 3                          |

Once this JSON object is received by the rest service we will move to the process explained next.

### 3.4.1.1.2 PROCESS JSON OBJECT

The rest service will start processing the JSON object doing the following:

- Extract the information mentioned above from the received JSON object
- Go through the information and check any updated information regarding the work orders assigned to that specific element.

### 3.4.1.1.3 SAVE IN CACHE MEMORY

The rest service will be saving the data in a cache memory for each element. Cache memory will help in achieving a better performance. Performance will be improved since accessing a cache memory is fast while a DB would be time consuming.

### 3.4.1.1.4 PUSH DATA IN A DATABASE ACCESS QUEUE

The rest service will be adding the data to a database access queue. This queue has been implemented using the singleton design pattern. It has been implemented this way in order to make sure that there is just a single access to database. This will help in avoiding multiple accesses to database and loss of information. Thus improving the functionality of the overall process.

### 3.4.1.2 FROM SYSTEM TO USER

On the other hand in order for the user to get a map with the elements monitored in real-time, there was a method implemented that is in charge of getting the data of any element and passing it to the front end map according to the processes shown in the following figure:
According to the above shown workflow. The system when is requested to get the data of any element, it first checks the cache memory if it exist then it returns it, otherwise it access the database and returns the required data.

After the required data is passed to the browser there is some processing in order to have a good looking map with all the details.
4. INCIDENCES MANAGEMENT

As mentioned in the requirements phase there will be Incidences that will be based on an incidence category and incident code.

In this section we will be explaining in details the following incidence categories and for each category the possible codes:

- Distribution Center incidences
- Intermodal platform incidences
- Vehicle/Route incidences
- Wagon/Route incidences
- Incidences at destination

4.3 DISTRIBUTION CENTER INCIDENCES

The distribution center operator will be the person in charge to detect these incidences and open an incident with the exact case and for each incidence there are some recommendations for actions to be done. However, some of the possible codes for an incidence case to be opened are explain below:

1. Goods that are to be delivered are not available at the distribution center or the documentation of the good is still not ready
2. Boxes to carry the goods is either not available at the distribution center or is damaged
3. Vehicles to carry the boxes is either not available at the distribution center or is damaged and needs repair
4. Batteries to be attached to the vehicle is either not available at the distribution center, damaged or is not charged
5. Driver did not show up

In each of the above-mentioned incidences the system will allow to open a new incidence and by mentioning the incidence code there will be a recommendation action.

4.4 INTERMODAL PLATFORM INCIDENCES

The operator will be the person in charge to detect these incidences and open an incident with the exact case and for each incidence there are some recommendations for actions to be done. However, some of the codes for an incidence case to be opened are explain below:
1. Truck is not aligned properly in order to automatically ship the boxes from the vehicle to the wagon or vice versa.
2. Vehicle still did not arrive at destination
3. One or more of the rails that carry the boxes in the intermodal platform is damaged and will not be able to perform the assigned task
4. Wagon still did not arrive at destination
5. There are more boxes than planned to be shifted from one element to another

In each of the above-mentioned incidences the system will allow to open a new incidence and by mentioning the incidence code there will be a recommendation action.

### 4.5 VEHICLE/ ROUTE INCIDENCES

The driver will be the person in charge to detect these incidences and send to the operator to open an incident with the exact case and for each incidence there are some recommendations for actions to be done. However, some of the codes for an incidence case to be opened are explain below:

1. The route that is assigned by the system is cut and the vehicle would not be able to go to destination through the assigned route, for any reason.
2. The destination planned is unreachable by the vehicle.
3. The driver decides to take another route other than the planned one; this could be because he knows a better route.
4. The driver faces a traffic jam that could affect the expected arrival time.
5. While performing the assigned work order the vehicle could get any failure.
6. While performing the assigned work order the vehicle gets short in power.

In each of the above-mentioned incidences the system will allow to open a new incidence and by mentioning the incidence code there will be a recommendation action.

### 4.6 WAGON/ ROUTE INCIDENCES

The driver will be the person in charge to detect these incidences and send to the operator to open an incident with the exact case and for each incidence there are some recommendations for actions to be done. However, some of the codes for an incidence case to be opened are explain below:

1. While performing the assigned work order the wagon could get any failure.
2. The route that is assigned by the system is not available and the wagon would not be able to go to destination through the assigned route, for any reason.

In each of the above-mentioned incidences the system will allow to open a new incidence and by mentioning the incidence code there will be a recommendation action.
The driver will be the person in charge to detect these incidences and send to the operator to open an incident with the exact case and for each incidence there are some recommendations for actions to be done. However, some of the codes for an incidence case to be opened are explain below:

1. Goods are not delivered because of wrong direction.
2. Due to any of the above sections of incidence there will be a delay in delivery.
3. Delivery is done but the client informs that the packs received are not the same as the ones he expected.
4. Additional goods have been delivered to a wrong client.
5. The client receives his goods incomplete.
6. The client receives the expected goods but in a bad condition.
7. The driver reaches the destination but there is nobody waiting for collection.

In each of the above-mentioned incidences the system will allow to open a new incidence and by mentioning the incidence code there will be a recommendation action.
5. ARCHITECTURE OVERVIEW

Since all the requirements have been gathered and have been well defined, now we need to define the architecture that the smart transportation system should follow. In this chapter we will be going through the architectural patterns and design that have been implemented, in addition to how the software is structured and the communication between the different components in order to fulfil the requirements.

5.1 ARCHITECTURE OBJECTIVES

Smart transportation system is a generic web application that is being designed for logistics reasons. However, since it is a generic application then it should have an architecture that is based on some objectives in order not to face problems in the future. In addition to the non-functional requirements of the system, the following subsections are the main non-functional requirements and constraints that were considered while designing the architecture of the software:

5.1.1 SCALABILITY

It is very important that the software be scalable in order to serve an increase in users. At the moment we will run the application on a single server. However, it will allow for scale out horizontally.

5.1.2 COSTS

Costs was an important factor in smart transportation system, since we are limited by costs in addition to the software should be stable. The result is using open source software in order to cut costs.

5.1.3 CUSTOMIZABLE

The software as mentioned earlier is a generic solution for logistic purposes. However, this means that the software should be customizable depending on future customer business model and business logic. Thus all the functions should extensible without affecting other components of the system. In addition this will help in having easily maintainable software.

5.2 APPLICATION COMPONENTS

5.2.1 APACHE WEB SERVER

Apache Web Server is an HTTP Server. It is used as the entry point of all the requests going to the system. Its main responsibility is handling all the requests, and redirects them to the
application component responsible of processing each request. Additionally, apache can be configured to balance load between different servers, and is a critical component to ensure application scalability and availability.

5.2.2 TOMCAT SERVER
Tomcat is an application container responsible of running the main web application that implements the control platform. Tomcat handles everything the application requires in the runtime, for instance, it handles the incoming connections, the threads required to run the application and the database connection pool. At least, and instance of Tomcat is required to run the application, and another one is required to run GeoServer, defined below.

5.2.3 GEOSERVER
GeoServer is an open-source java-based GIS Server. Its responsibility is handling the geospatial data in the application. It connects to the database, and provides standard OGC functions and web services to access the data. Maps rendering, data querying and geographic functionality requests are always redirected to GeoServer.

5.2.4 MEMCACHED:
MemCached is a cache layer added between Apache HTTP Server and Tomcat instances. Its main responsibility is store recently used data ready to serve, in order to avoid accesses to the database. This approach reduces the number of accesses to the database, resulting in better database performance, and faster responses to the users.

5.2.5 POSTGRESQL + POSTGIS DATABASE:
PostGIS is a PostgreSQL extension that extends PostgreSQL database functionalities with a set of functions and data types to efficiently handle geographic data. The application uses a single database to organize and store the application business data and the GIS data. GeoServer is out-of-the-box compatible with PostGIS databases.
Smart transportation system has been designed using a single server pattern. However, the following patterns have been taken into account in case required a better performance and or higher scalability.

5.4.1 SINGLE SERVER

First we have been using a single server application for development and testing. The following diagram shows the design:
This design will be running on a single server. However, this approach has some drawbacks as follows:

- **Performance**: Depends on the configuration of the server: CPU speed, available memory.
- **Availability**: Every software and hardware element is a single point of failure: if it breaks, the entire system hangs.
- **Security**: If web server has been attacked, then all data will be accessed.
- **Low cost**
- **Low complexity**

### 5.4.2 DATABASE REPLICA\ION

This is an alternative approach that can be applied to the architecture where there is more than one database being used as seen in the figure below:

This approach improves the data availability. However, server synchronization is very important in order to make sure that data is consistent. Databases can run on different servers allowing server specialization to be achieved and better horizontal scalability.
5.4.3 SERVER REPLICATION

This is an alternative approach that can be applied to the architecture where there is more than one server being used as seen in the figure below:

![Figure 42]

This approach makes use of the apache server being used as a load balancer in order to distribute the work between the two servers equally. This approach will serve when there is an increase in the number of users leading to an increase in database access causing the server to get bottlenecked.

5.4.4 DATABASE SHARDING

This is an alternative approach that can be applied to the architecture where there is more than one database serving the same server as seen in the figure below:

![Figure 43]
This approach is common when the database has big datasets. This approach splits the data according to an applied rule i.e. depending on the data we decide where to place it.

As seen in the diagram we have different Tomcat + Database. The apache server can be configured to redirect a request based on different options, for instance, the URL, the origin IP address, will support this.

From the above-mentioned sections, different approaches each improving a different aspect either scalability or availability. However, we can use different techniques together as seen in the figure below:

![Diagram of Apache server, Tomcat servers, and databases](image)

As seen in the figure above we have applied database replication, server replication and database sharding.

### 5.5 DESIGN PATTERNS

Smart transportation system has been designed to have the following patterns taking into account other patterns. The following subsections will explain the patterns being used and how the system has been designed to allow for scalability and better performance.
5.5.1 LAYERED ARCHITECTURE STYLE

In Smart transportation system we have used 3-layered architecture as seen in the figure below:

![Layered Architecture Diagram]

Figure 45

Using three layered architecture design pattern will help in grouping similar functionalities into the same layer. However, communication between layers is loosely coupled. Each layer can also be a different tier in order to improve scalability. This design pattern will improve the following:

- Isolation of concerns
- Manageability of the code
- Performance if distributed among different tiers
- Reusability
- Testability

5.5.2 DEPENDENCY INJECTION PATTERN

In Smart transportation system we have used the dependency injection pattern throughout the application. This will improve the reusability of the code by implementing more efficient
algorithms without having to change the main functionality. Moreover, this will manage the communication between the different components and allow for less-coupled design.

5.5.3 MODEL-VIEW-CONTROLLER PATTERN

In Smart transportation system we have used the model-view-controller design pattern.

Model:

- Contains core functionality
- Encapsulates the appropriate data, and exports procedures that perform application-specific processing

View:

- Obtains the data through the model
- Displays the data to the user
- Depending on the view data can be displayed differently

Controller:

- Accepts user input as events
- Translates events to service requests for the model or display requests for the view

5.5.4 SERVICE ORIENTED ARCHITECTURE PATTERN

In Smart transportation system we have used the service oriented architecture design pattern. This approach has been developed in order to do some processes using a set of services. Services are loosely coupled because they are running independently on different machines and accessed through interfaces.

Since there will be interactions between the tablets placed in the vehicles and wagons and the control platform, so it is important that the software would be having a set of services that will be receiving incoming data and process it. In addition we have been using external services for calculating the coordinates of a given address as well as calculating the route for given coordinates.

SOA approach has benefits such as:

- Domain alignment
- Abstraction
• Discoverability
• Interoperability
• Rationalization.

5.5.5 DATA ACCESS OBJECT PATTERN

In Smart transportation system we have applied the data access object pattern in order to separate low level database accesses from the business logic. This will also be useful in case of migrating to different database.

5.6 DATABASE ACCESS

Smart transportation system is data intensive application. It is very important how database will be configured in order to handle the amount of data. In the following sections we will show the different techniques used in order to achieve this goal and have a good performance application.

5.6.1 DATABASE CONNECTION

Smart transportation system is developed using spring framework and spring boot in concrete and running on a tomcat web server. Spring framework is a Java web development framework that implements most of the typical web application requirements and functionalities including the database connection based on standards. However, spring manages a pool of JDBC connections to the PostgreSQL Database. In addition spring configures Hibernate, a technology that simplifies data access by providing an intuitive mapping to data using Java classes.

5.6.2 MEMCACHED

Smart transportation system has real time monitoring of elements. Real-time monitoring is data intensive because it requires a lot of data to be able to show the current location of any element. If the only data source that is available is the database there will be over accessing to database through reads and writes. This will affect the database being busy in addition to the overall performance of the system.

The memcached approach is based on placing a cache layer between the system and the database. This will remove a big load from the database. This approach will be applied for real-time monitoring where the data will be stored in the cache memory and sent to a method that is responsible for taking care of the data by saving it in the database. In general a cache
memory has a better performance than the database, so it will improve the overall performance of the system. The figure below illustrates a normal scenario where a lot of elements are sending their data to the system and how the system treats the data.

As seen in the figure above, the system get data from different elements and save the data in the memcached and at the same time place it in a queue that will be responsible for saving it in the database. On the other hand for the end user to be able to have a map with all the elements location, the system starts by checking the elements data in the memcached if it exists it returns it and if not it access the database to get the data.

5.7 TECHNOLOGIES

Smart transportation system has been implemented using different technologies. In the following subsections the technologies used will be explained. These technologies have been chosen in order to achieve the objectives mentioned earlier in this document.

5.7.1 JAVA

Java is one of the most stable programming languages. It has a lot of documentation and support. Moreover, java has been proven to be one of the best languages for web applications.

5.7.2 SPRING BOOT

Spring boot is a framework built on top on Java. It describes a set of rules and conventions considered useful to build the most typical features every web application requires. Spring boot provides authentication mechanisms, data access mechanisms and integrations with most of the tools used nowadays for web development.

5.7.3 THYMELEAF

Thymeleaf is a template processor. Spring boot provides a good support for the MVC design pattern along with the thymeleaf framework, thus allowing the development of interactive views in an easy manner binding the server side data with the views.
5.7.4  TWITTER BOOTSTRAP

Twitter Bootstrap is a library designed to make it easy to create accessible, cross-browser, mobile-friendly and good-looking websites. Bootstrap provides good-looking styles for having highly usable web application.

5.7.5  JAVASCRIPT

JavaScript is a front end programming language. Javascript has been used in order to pre-process the data in the views in addition to some animation.

5.7.6  LEAFLET.JS

Leaflet.js is a minimal JavaScript mapping library. It provides the most basic functions out-of-the-box, but there are plenty of plugins developed to enhance its features and provide a full-featured map library.

5.7.7  HIBERNATE

Hibernate is an Object/Relational Mapping (ORM) library. It makes it easier for the developer to interact with the database, and to perform operations on it.

5.7.8  POSTGRESQL + POSTGIS

PostgreSQL is the database engine used in smart transportation system. There are plenty of different Relational Database Management Systems (RDBMS), but PostgreSQL was chosen mainly because PostGIS, a library extending PostgreSQL functionalities to implement GIS features.
6. SMART TRANSPORTATION SYSTEM DATA MODEL

This section describes the data model used by the application to store all the application data.

To make it easier to understand, each main concept on the application is defined in a separated subsection. For each concept, a data model diagram is shown, and then a definition for each table shown in the diagram.

6.1 VEHICLE

Vehicle data model:
### Vehicle

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer value that represents the identifier unique key of each vehicle. Primary key.</td>
</tr>
<tr>
<td>Plate_number</td>
<td>Text value that represents the plate number of the vehicle.</td>
</tr>
<tr>
<td>Purchase_order_id</td>
<td>Integer value that references the purchase order id the vehicle is currently assigned to. Foreign key pointing to table purchase_order, attribute id.</td>
</tr>
<tr>
<td>Vehicle_model</td>
<td>Integer value that references vehicle model identifier. Foreign key pointing to table vehicle_model, attribute id.</td>
</tr>
<tr>
<td>Current_Location</td>
<td>Text value that represents the actual vehicle location.</td>
</tr>
</tbody>
</table>
Aviability: Text value that represents the availability of the vehicle.

**Place_dc_id**: Integer value that references the distribution center identifier in which the vehicle is. Foreign key pointing to table `distribution_center`, attribute id.

**Place_oc_id**: Integer value that references the operation center identifier in which the vehicle is parked. Foreign key pointing to table `operation_center`, attribute id.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer value that represents the unique identifier of each model of vehicle. Primary key.</td>
</tr>
<tr>
<td>Model</td>
<td>Text value that describes the model features.</td>
</tr>
<tr>
<td>Max_speed</td>
<td>Integer value that represents the maximum speed of a model.</td>
</tr>
<tr>
<td>Power</td>
<td>Integer value that represents the power of a model.</td>
</tr>
<tr>
<td>Year</td>
<td>Integer value that represents the model year creation</td>
</tr>
<tr>
<td>Boxes attach</td>
<td>Integer value that represents the number of boxes this model can carry.</td>
</tr>
<tr>
<td>Type_battery</td>
<td>Integer that represents type of battery used by this model.</td>
</tr>
<tr>
<td>Max_batteries</td>
<td>Integer than show the maximum number of batteries in this model.</td>
</tr>
<tr>
<td>Type_box</td>
<td>Integer value which represents the type of boxes carried by this model.</td>
</tr>
<tr>
<td>Max_weight_boxes</td>
<td>Integer value that represents the maximum weight of the boxes that can be carried by this model.</td>
</tr>
<tr>
<td>Weight</td>
<td>Integer that represents the model weight.</td>
</tr>
</tbody>
</table>

Table 4

**Vehicle model**: Entity representing vehicle models. Currently there are two models available: model 1 and model 2, but the system is extensible to allow different vehicle models.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer that identifies a vehicle history item. Primary key.</td>
</tr>
<tr>
<td>Vehicle</td>
<td>Integer that references a vehicle identifier. Foreign key pointing to table <code>vehicle</code>, attribute id.</td>
</tr>
</tbody>
</table>

Table 5

**Vehicle History**: Entity designed to store the real-time data sent by vehicles.
<table>
<thead>
<tr>
<th>Moment</th>
<th>Integer that represents the moment in which position is taken.</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>String value that represents the state of a vehicle in a moment.</td>
</tr>
<tr>
<td>Comments</td>
<td>Text value that represents the possible comments about a state of a vehicle.</td>
</tr>
<tr>
<td>Location</td>
<td>Text value that represents the coordinates of a vehicle in a moment.</td>
</tr>
</tbody>
</table>

Table 6

The entities defining “purchase order”, “distribution center” and “operation center” are defined in further subsections in this document.

6.2 BOXES

These entities are involved in the boxes data model:
**Box**: Entity that represents a box.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer that represents a box identifier. Primary key.</td>
</tr>
<tr>
<td>Purchase_order_id</td>
<td>Integer value that references the purchase order the box is currently assigned to. Foreign key pointing table purchase_order, attribute id.</td>
</tr>
<tr>
<td>Box_detail</td>
<td>Integer value that represents the box detail. Foreign key pointing table box_detail, attribute id</td>
</tr>
<tr>
<td>Serial</td>
<td>Integer that represents the box serial number.</td>
</tr>
<tr>
<td>Availability</td>
<td>Text value that describe the availability of the box.</td>
</tr>
<tr>
<td>Place_vehicle_id</td>
<td>Integer that references the vehicle assigned to the box. Foreign key pointing table vehicle, attribute id</td>
</tr>
<tr>
<td>Place_dc_id</td>
<td>Integer that references the distribution center assigned</td>
</tr>
</tbody>
</table>
Foreign key pointing to table distribution_center, attribute id.

| PLACE_WAGON_ID | Integer that references the wagon assigned to the box. Foreign key pointing to table wagon, attribute id. |
| PLACE_OC_ID    | Integer value that references the operation center identifier in which the vehicle is parked. Foreign key pointing to table operation_center, attribute id. |
| PLACE_IP_ID    | Integer value that references the intermodal platform assigned to the box. Foreign key pointing to table intermodal platform, attribute id. |

**Box_detail:** Entity that stores information about specific box models

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the unique identifier of a box detail. Primary key.</td>
</tr>
<tr>
<td>Lengthe</td>
<td>Numeric value with double precision. Represents the external length of a box.</td>
</tr>
<tr>
<td>Widthe</td>
<td>Numeric value with double precision. Represents the external width of a box.</td>
</tr>
<tr>
<td>Heigthe</td>
<td>Numeric value with double precision. Represents the external height of a box.</td>
</tr>
<tr>
<td>Length</td>
<td>Numeric value with double precision. Represents the internal length of a box.</td>
</tr>
<tr>
<td>Width</td>
<td>Numeric value with double precision. Represents the internal width of a box.</td>
</tr>
<tr>
<td>Height</td>
<td>Numeric value with double precision. Represents the internal height of a box.</td>
</tr>
<tr>
<td>Capacity</td>
<td>Numeric value with double precision. Represents the box capacity.</td>
</tr>
<tr>
<td>Net_weight</td>
<td>Integer value with double precision that represent the net weight of a box.</td>
</tr>
<tr>
<td>Max_load</td>
<td>Integer value with double precision. Represents the max load of a box.</td>
</tr>
<tr>
<td>Model</td>
<td>Text that represents the model of a box.</td>
</tr>
</tbody>
</table>

"vehicle", "operation_center", "wagon" and "purchase order" are entities defined in other subsections of this document.
These entities are involved in the wagons data model:

**Wagon**: Entity that represents a particular wagon.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the unique identifier of a wagon. Primary key</td>
</tr>
<tr>
<td>Plate_number</td>
<td>Integer that represents the plate number of a wagon.</td>
</tr>
<tr>
<td>Purchase_order_id</td>
<td>Integer value that references the purchase order that the wagon is currently working on. Represents the actual order that is attending this wagon. Foreign key pointing to table purchase_order, attribute id.</td>
</tr>
<tr>
<td>Wagon_model</td>
<td>Integer value that references the model of a wagon. Foreign key pointing to wagon_model table, attribute id.</td>
</tr>
</tbody>
</table>

**Wagon_model**: Entity that describes wagon’s features.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the identifier of a wagon model. Primary key</td>
</tr>
<tr>
<td>Model</td>
<td>Text value that describe a wagon model.</td>
</tr>
<tr>
<td>Max_speed</td>
<td>Integer value that represents the max speed this wagon model can reach.</td>
</tr>
<tr>
<td>Power</td>
<td>Integer value that represents the power of a wagon model.</td>
</tr>
<tr>
<td>Year</td>
<td>Integer value that represents the year of creation of a wagon model.</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Boxes_attach</td>
<td>Integer value that represents the number of boxes that can be carried by this wagon model.</td>
</tr>
<tr>
<td>Type_battery</td>
<td>Integer that represents the type of battery used by this wagon model.</td>
</tr>
<tr>
<td>Max_batteries</td>
<td>Integer that shows the maximum number of batteries in this wagon model.</td>
</tr>
<tr>
<td>Type_box</td>
<td>Integer that represents the type of boxes carried by this wagon model.</td>
</tr>
<tr>
<td>Max_weight_boxes</td>
<td>Integer that represents the maximum weight of boxes that can be carried by this wagon model.</td>
</tr>
<tr>
<td>Max_regen_power</td>
<td>Integer that represents the maximum regeneration power of this wagon model.</td>
</tr>
</tbody>
</table>

**Table 10**

“purchase_order” is defined in another subsection of this document.

6.4 BATTERIES

These entities are involved in the batteries data model:
Battery: Entity that represents batteries.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the unique identifier of a battery. Primary key</td>
</tr>
<tr>
<td>Purchase order</td>
<td>Integer value that references the purchase order the battery is assigned to. Represents the actual order that is using this battery. Foreign key pointing to table purchase_order, attribute id.</td>
</tr>
<tr>
<td>Wagon_id</td>
<td>Integer that references the wagon that is using this battery. Foreign key pointing to table wagon, attribute id.</td>
</tr>
<tr>
<td>Vehicle_id</td>
<td>Integer that references the vehicle that is using this battery. Foreign key pointing to table vehicle, attribute id</td>
</tr>
<tr>
<td>Serial</td>
<td>Integer that represents the battery serial number.</td>
</tr>
<tr>
<td>Battery_model</td>
<td>Integer value that references the model of a battery. Foreign key pointing to battery_model table, attribute id</td>
</tr>
</tbody>
</table>
Recharge_slot: Integer value that defines the recharge slot of a battery

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the unique identifier of a battery model. Primary key</td>
</tr>
<tr>
<td>Model</td>
<td>Text that represents the description of a battery model.</td>
</tr>
<tr>
<td>Type</td>
<td>Text that represents the type of a battery model.</td>
</tr>
<tr>
<td>Year</td>
<td>Integer value that represents the year of creation of a battery model.</td>
</tr>
<tr>
<td>Capacity</td>
<td>Integer value that represents the capacity of a battery model.</td>
</tr>
<tr>
<td>Voltage</td>
<td>Integer value that represents the voltage of a battery model.</td>
</tr>
<tr>
<td>Max_charge_power</td>
<td>Integer value that represents the maximum charge power of a battery model.</td>
</tr>
<tr>
<td>Max_discharge_power</td>
<td>Integer value that represent the maximum discharge power of a battery.</td>
</tr>
<tr>
<td>Dimensions_std</td>
<td>Integer value that represents the dimensions of a battery model.</td>
</tr>
<tr>
<td>Weight</td>
<td>Integer that represents the weight of a battery model.</td>
</tr>
</tbody>
</table>

“vehicle“, “purchase_order“ and "wagon“ are entities defined in another subsection of this document.

6.5 PURCHASE ORDER

These entities are involved in the purchase order data model:
**Purchase_order**: Entity that represents an order made by a customer.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the purchase order unique identifier. Primary key.</td>
</tr>
<tr>
<td>Route_id</td>
<td>Integer value that references an order route. Foreign key pointing route to table, attribute id.</td>
</tr>
<tr>
<td>Customer_id</td>
<td>Integer value that references customer. Foreign key pointing customer table, attribute id.</td>
</tr>
<tr>
<td>Destination</td>
<td>Text that represents the order destination.</td>
</tr>
<tr>
<td>State</td>
<td>Text that represents the order state.</td>
</tr>
<tr>
<td>Arrival_time</td>
<td>Integer value that represents order arrival time.</td>
</tr>
<tr>
<td>Comments</td>
<td>Text that represents the order comments.</td>
</tr>
<tr>
<td>Geo_destin</td>
<td>Text that represents geo destination of the order.</td>
</tr>
<tr>
<td>Origin</td>
<td>Integer value that represents order origin.</td>
</tr>
</tbody>
</table>
Serial

Integer value that represents order Serial.

Expected_arrival_time

Integer value that represents the expected arrival time of the order.

Departure_time

Integer value that represents order’s departure time.

Expected_departure_time

Integer value that represents order’s expected departure time.

Table 13

“route” and “customer” data models are defined in another subsection of this document.

6.6 INTERMODAL PLATFORM

These entities are involved in the intermodal platform data model:

Figure 52

Intermodal_platform: Entity that represents an intermodal platform.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that describes an intermodal platform unique identifier. Primary key</td>
</tr>
<tr>
<td>Year</td>
<td>Integer value that represents the year of creation of the intermodal platform.</td>
</tr>
<tr>
<td>Model</td>
<td>Text value that describes the model of the intermodal platform.</td>
</tr>
<tr>
<td>Intermodal_platform_model</td>
<td>Integer value that references the intermodal platform model of the intermodal model. Foreign key pointing table intermodal_platform_model, attribute id.</td>
</tr>
<tr>
<td>Train_station</td>
<td>Integer value that references the train station of the intermodal model. Foreign key pointing to table train_station, attribute id.</td>
</tr>
</tbody>
</table>

Table 14

Intermodal_platform_model: Entity that describes intermodal platform models features.
<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that represents the unique identifier of an intermodal platform model. Primary key</td>
</tr>
<tr>
<td>Type</td>
<td>Integer that represents the model of an intermodal platform model.</td>
</tr>
<tr>
<td>Max_batteries</td>
<td>Integer value that represents the maximum number of batteries that this model can attach.</td>
</tr>
<tr>
<td>Num_lines_boxes</td>
<td>Integer value that represents the number of lines that this model have for handling the transfer of boxes</td>
</tr>
<tr>
<td>Num_boxes_per_line</td>
<td>Integer value that represents the max number of boxes that a line can handle at any time</td>
</tr>
</tbody>
</table>

Table 15

“train_station” is an entity defined in another subsection of this document.

6.7 TRAIN STATION

This is the entity are involved in the train station data model:

![Train_station entity](image)

**Train_station**: Entity that represents a train station.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Integer value that describe a train station unique identifier. Primary key</td>
</tr>
<tr>
<td>Name</td>
<td>Text value that represents the name of train station</td>
</tr>
<tr>
<td>Address</td>
<td>Text value that describes the address of the train station</td>
</tr>
<tr>
<td>Geom</td>
<td>Text value that represents the actual longitude and latitude of the train station</td>
</tr>
</tbody>
</table>

Table 16

6.8 INCIDENCE

These entities are involved in the incidence entity:
**Incidence**: Entity that represents an incidence.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id_incidence</td>
<td>Integer value that describe an incidence unique identifier. Primary key</td>
</tr>
<tr>
<td>Incidence_codes_id_code</td>
<td>Integer value that references the incidence codes id of the incidence codes. Foreign key pointing table incidence_codes, attribute id_code.</td>
</tr>
<tr>
<td>Incidence_categories_id_category</td>
<td>Integer value that references the incidence categories of the incidence categories. Foreign key pointing to table incidence_categories, attribute id_category.</td>
</tr>
<tr>
<td>description</td>
<td>Text value that describes the incidence.</td>
</tr>
<tr>
<td>status</td>
<td>Text value that describes the status of the incidence in case it is solved or still open.</td>
</tr>
<tr>
<td>Time_opened</td>
<td>Date-Time value that shows the exact time when the incidence case was opened.</td>
</tr>
<tr>
<td>Time_closed</td>
<td>Date-Time value that shows the exact time when the incidence case was closed.</td>
</tr>
<tr>
<td>Affected_dc_id</td>
<td>Integer value that references the affected distribution center from the list of distribution centers. Foreign key pointing table distribution_center, attribute id.</td>
</tr>
<tr>
<td>Affected_oc_id</td>
<td>Integer value that references the affected operation center from the list of operation centers. Foreign key pointing table operation_center, attribute id.</td>
</tr>
<tr>
<td>Affected_wo_id</td>
<td>Integer value that references the affected work order from the list of work orders. Foreign key pointing table work_order, attribute id.</td>
</tr>
<tr>
<td>Affected_vehicle_id</td>
<td>Integer value that references the affected vehicle from the list of vehicles. Foreign key pointing to table vehicle, attribute id.</td>
</tr>
<tr>
<td>Affected_driver_id</td>
<td>Integer value that references the affected driver from the list of drivers.</td>
</tr>
</tbody>
</table>
drivers. Foreign key pointing table driver, attribute id.

<table>
<thead>
<tr>
<th>Affected_battery_id</th>
<th>Integer value that references the affected battery from the list of batteries. Foreign key pointing table battery, attribute id.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affected_box_id</td>
<td>Integer value that references the affected box from the list of boxes. Foreign key pointing table box, attribute id.</td>
</tr>
</tbody>
</table>

**Table 17**

**Incidence_categories**: Entity that describes incidence categories.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id_category</td>
<td>Integer value that represents the unique identifier of an incidence category. Primary key</td>
</tr>
<tr>
<td>Category_code</td>
<td>Integer that represents the code of the incidence category.</td>
</tr>
</tbody>
</table>

**Table 18**

**Incidence_codes**: Entity that describes incidence codes.

<table>
<thead>
<tr>
<th>COLUMN NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id_code</td>
<td>Integer value that represents the unique identifier of an incidence code. Primary key</td>
</tr>
<tr>
<td>Incidence_code</td>
<td>Integer that represents the code of the incidence.</td>
</tr>
<tr>
<td>Incidence_categories_id_category</td>
<td>Integer value that references the incidence categories of the incidence categories. Foreign key pointing table incidence_categories, attribute id_category.</td>
</tr>
</tbody>
</table>

**Table 19**
7. TESTING

Testing comes last in the software development lifecycle. However, it is very important to test the software many times before it is sent to production. Smart transportation system has been tested in three different ways. The following subsections will explain in details the testing phases that the software went through.

7.1 UNIT TESTING

The first phase that was done in parallel with the development process was the unit testing. Unit testing is basically testing each component individually and make sure that the methods are functioning as required. For example in smart transportation system all the algorithms have been tested using JUnit testing in order to make sure that the outcome is exactly as required.

7.2 INTEGRATION TESTING

The second phase in testing was the integration testing, integration testing comes after the unit testing and the idea behind is to make sure that the different components are well integrated and interact with each other according to the required manners. Since the unit testing was performed before so less issues were discovered.

The integration test being used was the big bang method. This method is based on several components being integrated then undergoes the integration test. This was done for each requirement individually.

7.3 SYSTEM TESTING

The last phase of testing that the smart transportation system has been through is the system testing, system testing is a final test that is done after all the system has been integrated in order to make sure that the system is implemented according to the functional requirements requested in the early stages and that the system overall goal is achieved.
8. CONCLUSION AND FUTURE WORK

Smart transportation system application development have followed different phases in an iterative manner as any other software. Iterations start from the requirements analysis phase followed by the design phase, then implementation and finally testing phase then some evolution as seen throughout the document.

This document has included different sections. Starting from the requirements phase where all the requirements have been gathered. Functional requirements section is including an I-star model that shows the interaction among the different actors that form part of the requirement in addition to use cases with mock-ups for that requirement. Following was the non-functional requirements where the some criteria have been defined in order to value the software’s operations. Second was the time planning, which is a time plan that the project should be following in order to be accomplished on time. Business logic comes third defining all the processes that have been applied in the system and how they can be reflected in the system. Fourth is the incidence management section where all the possible incidences that the system could face are mentioned. Fifth came the architecture phase explaining the architecture that has been chosen in order for the system to achieve its goals and possible alternative solutions that can be applied to the system to meet other non-functional requirements. Sixth was the data model, since smart transportation system is a data intensive software, this section came to show how the database has been designed in order for the software to be capable of handling the huge amount of data. Finally the testing phase, in the testing phase showed the different testing phases that the software went through. To sum up this software will have a direct impact for the potential customers that will adapt it to their infrastructure.

Smart transportation system functionality has been achieved. Since this software should offer the best results some future work should be done as mentioned earlier throughout the document. Future work in smart transportation system would be including a better algorithm for distributing goods into the boxes, the future algorithm should be applying 3 dimensional modelling of both elements in order to achieve the best results and maximum utilization of the elements. In addition to spatial clustering which will be grouping the orders according to destination. Spatial clustering should be working in parallel to a routing system in order to give the best results. Finally, we would like to apply some system learning. System learning will be explained through a simple example. If we assume that the vehicles have been through a route for several times, according to the planning phase x time has been assigned for this trip. However, the reality is that the vehicle spends y time, so the system should be intelligent enough in analysing this difference in time and plan according to the time spent y.
9. REFERENCES


