Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL FINAL DE GRAU

TÍTOL DEL TFG: Plataforma de gamificació docent amb smartphones

TITULACIÓ: Grau en Enginyeria de Sistemes de Telecomunicació

AUTOR: Daniel Viles Florejachs

DIRECTOR: Roque Meseguer Pallares

DATA: 30 de gener del 2018

Títol: Plataforma de gamificació docent amb smartphones

Autor: Daniel Viles Florejachs

Director: Roque Meseguer Pallares

Data: 30 de gener del 2018

Resum

L'eina de gamificació escolar que es descriu en el present document sorgeix de voler incorporar noves funcionalitats a la plataforma de software ja existent. Aquesta es basa en realitzar preguntes als estudiants amb l'objectiu d'oferir motivació a l'estudi. Les preguntes són tipus multiresposta amb autocorrecció, resposta oberta i amb imatge. S'ha incorporat un temporitzador a cada pregunta com a repte per a l'estudiant.

L'arquitectura de l'eina està dissenyada per a ser utilitzada en una Aplicació Mòbil compatible en Android i iOS i, des d'un Panell d'Administració web accessible des de qualsevol sistema operatiu. La comunicació entre les dues plataformes la uneix una arquitectura orientada a serveis que conté una Interfícies de Programació d'Aplicacions (API).

Per a l'organització del present treball s'ha utilitzat l'eina Gantt Project on es planifica el desenvolupament de totes les tasques que es duen a terme per a la finalització del treball. Durant el desenvolupament s'executa les proves de validació corresponents i una vegada finalitzat, aquest s'emmagatzema al dipòsit GitHub on es verifica i detecta si hi ha algun problema al programari.

L'Aplicació Mòbil és l'eina de motivació d'estudi que utilitzaran els estudiants. El desenvolupament de les plataformes Android i iOS s'ha realitzat a través d'una aplicació híbrida en base a tecnologies web i una plataforma que permet la conversió als diferents sistemes. El disseny d'aquesta arquitectura segueix el model MVC i s'ha construït en base a ser una plataforma multiidiomes.

El Panell d'Administració és la plataforma web per a què els professors puguin crear, eliminar, visualitzar els qüestionaris, les respostes i els resultats. Per al disseny d'aquesta eina s'ha transformat la plataforma nativa per a una de nova que mostra una interfície d'usuari més moderna, ràpida i consistent i compatible també en diferents idiomes.

L'Arquitectura Orientada a Serveis que s'ha desenvolupat en aquest treball és l'encarregada de gestionar totes les interaccions entre els usuaris de l'Aplicació Mòbil i el Panell d'Administració amb el Back-End. La gestió d'aquestes dades es realitza a través d'una API que conté els Models (objectes de JavaScript) i Relacions (permeten relacionar els models entre si) que generen l'Arquitectura de Serveis.

Title: Teaching gamification platform with smartphones

Author: Daniel Viles Florejachs

Director: Roque Meseguer Pallares

Date: January 30, 2018

Overview

The educational gamification tool described in the present document has been accomplished as a result of wanting incorporate new functionalities to already existing software. This is based on creating questions for students to motivate them to study. The questions are either multiple choice with autocorrection or open answer with images. A timer has been implemented in every question to challenge the student.

The architecture of the tool is designed to be used as a Mobile Phone Application compatible with both Android and iOs and from a Administrative Web Panel accessible via any operating system. The communication between the two platforms is integrated by a Service Oriented Architecture that contains an Application Programming Interface (API).

The project organization for this work was done with Gannt Project where each task was planned and finished until completion. During the development, corresponding validation tests are executed and once finished, these are stored on GitHut where they can been verified and programming errors are detected.

The Mobile Application is a motivational tool for study that students will use. The development of the platforms Android and iOS was with a hibrid application based on web technology and a platform that permits the conversion to different systems. The design of this architecture follows the MVC model and is constituted based on being a multilingual platform.

The Administrative Panel is a web platform so that teachers can create, eliminate and view the tests, answers and students' results. The design of this tool is a transformation of the native platform that offers a modern user interface that is fast, consistent and compatible with different languages.

The Service Oriented Architecture that has been developed in this project is responsible for managing all the interactions between the users of the Mobile Application and the Panel Administration with the Back-End. The data management is accomplished with an API that contains the Models (JavaScript objects) and Relations (that permit relating the models to one another) that generate the Service Architecture.

ÍNDEX

INTRODUCCIÓ

El present treball de grau tracta del disseny d'una plataforma de software amb font oberta com a una **eina de gamificació escolar** per a realitzar activitats docents amb smartphones, amb l'objectiu d'aconseguir que els estudiants es motivin en l'estudi de la matèria docent que estigui realitzant. Aquesta eina té la finalitat de complir amb les exigències acadèmiques per evitar el fracàs o la deserció escolar. D'aquesta manera, l'eina estarà disponible com ajuda de manera eficaç per a superar aquest problema.

El desenvolupament d'aquesta plataforma s'ha realitzat a partir d'una ja existent anomenada **Classpip**, que cobreix les necessitats dels professionals de l'entorn educatiu, en què, s'ha dissenyat una nova eina que permet refrescar el contingut de la matèria als estudiants a través de **qüestionaris**.

L'objectiu general del disseny d'aquesta nova plataforma és de construir una **eina de motivació d'estudi basada en preguntes**. Els professors de l'entorn docent han de realitzar qüestionaris que continguin preguntes sobre el temari de l'assignatura que s'imparteixi. D'aquesta manera els estudiants poden realitzar aquests qüestionaris utilitzant els seus smartphones des de qualsevol lloc per a poder autoavaluar-se els seus propis coneixements assolits en la matèria que estiguin realitzant.

Per al disseny d'aquesta nova eina es té en compte una sèrie d'objectius específics que són la base de la construcció d'aquesta plataforma de software, els quals es nombren a continuació.

El primer objectiu específic es basa, en què, hi ha dos tipus de qüestionaris per a realitzar segons el tipus de pregunta, és a dir, existeix un tipus de qüestionari amb preguntes tipus test i **multiresposta** i existeix un altre tipus de qüestionari amb preguntes que contenen o no una **imatge** i amb **resposta oberta** per a redactar.

El segon objectiu específic tracta d'imposar a l'estudiant un desafiament alhora de realitzar les preguntes, és a dir, se li aplica **un repte a cada pregunta** del qüestionari. Per a posar en pràctica aquest repte s'ha introduït un **temporitzador** a cadascuna de les preguntes. Quan l'estudiant inicia el qüestionari apareix en la pantalla del seu smartphones la primera pregunta amb el display del temporitzador iniciat en compte enrere, aquest disposa d'un cert temps per a respondre la pregunta, en el cas de què finalitzi el temporitzador i l'estudiant no hagi respost encara, automàticament apareixerà en la pantalla la següent pregunta del qüestionari amb el seu temporitzador inicialitzat altra vegada.

El tercer objectiu específic és el de poder tenir una **correcció automàtica** de les preguntes que realitzen els estudiants. En aquesta nova plataforma de software només s'ha dissenyar l'autocorrecció de les preguntes que contenen multiresposta. Per tant, una vegada finalitzat el qüestionari els estudiants podran saber les respostes correctes i el resultat que han obtingut.

A la plataforma de software nativa **Classpip** se li ha realitzat actualment una sèrie de canvis que transforma l'eina dissenyada anteriorment en una de nova més **actual**, **moderna** i **eficaç**. Aquests canvis s'expliquen a continuació.

El primer canvi que s'ha realitzat fa possible que tota la plataforma estigui **disponible en tres idiomes diferents**, ja que anteriorment només ho estava en castellà. Per tant, això significa que qualsevol estudiant que accedeixi a l'aplicació mòbil o qualsevol professor que ho faci en el Panell d'Administració, aquest té en la pàgina principal un selector per a poder escollir l'idioma de la plataforma que estigui utilitzant. L'eina està dissenyada en tres idiomes diferents, amb català, castellà i anglès.

El segon canvi tracta de la transformació del Panell d'Administració natiu per un de més actual. L'antic Panell estava dissenyat amb el framework Bootstrap i l'actual ho està amb el framework **Angular Material**. Aquest disposa de components integrats i moderns d'**interfície d'usuari** que funciones a través de la web, mòbil i escriptori. El nou framework garanteix un rendiment finament **ajustat**, **ràpid** i **coherent** a la plataforma.

CAPÍTOL 1. INICIACIÓ AL PROJECTE

1.1. Motivació del projecte

El present projecte fi de Grau sorgeix degut a què el Departament d'Arquitectura de Computadors de la **Universitat Politècnica de Catalunya** [1] a l'any 2016 va desenvolupar una plataforma de software anomenada **Classpip**[1], des d'un punt de vista arquitectural, per a poder formar un entorn de **gamificació escolar**[2] mitjançant una Aplicació Mòbil i un Panell d'Administració.

Una vegada es va crear aquesta plataforma, el departament de la Universitat va decidir incorporar-ne noves funcionalitats. Una d'elles és la que es presenta en aquest projecte i que tracta d'una eina de gamificació escolar basada en preguntes que es realitzen als estudiants a través de qüestionaris. Aquesta eina està inspirada amb ***Socrative*** [2] i ***Kahoot*** [3].

L'elecció d'aquesta nova funcionalitat respon a dues circumstàncies: la primera està relacionada amb finalitats educatives per a poder oferir als estudiants una millor metodologia de motivació d'estudi; la segona és per l'avantatge que ofereix als estudiants alhora de realitzar el qüestionari, en què, aquests tenen l'opció de poder realitzar-lo en qualsevol espai i utilitzant qualsevol plataforma tecnològica.

1.2. Objectiu principal

L'objectiu principal d'aquest projecte és el desenvolupament d'aquesta eina de motivació d'estudi. On els professors realitzen qüestionaris amb preguntes relacionades amb el contingut de la matèria realitzada en les anteriors sessions, d'aquesta manera l'eina motiva als estudiants a refrescar el contingut de la matèria.

Aquest objectiu general es ramifica en tres objectius específics, els quals s'exposen a continuació:

1) Els qüestionaris contenen tres tipus de preguntes diferents: la primera és una pregunta multiresposta (**test**) amb autocorrecció; la segona és una pregunta amb resposta oberta per a desenvolupar (**textArea**) i sense autocorrecció; la tercera pregunta incorpora una imatge (**image**) i tampoc té correcció automàtica.

[1] Arquitectura de programari per a professors i estudiants que realitzen activitats de gamificació escolar dins dels entorns educatius a través de diferents plataformes com mòbils, tauletes i ordinadors.
[2] Consisteix en aplicar estratègies de joc (regles de jocs, punts ...), per ajudar o modificar el reforç de les conductes del estudiants.

2) Cadascun dels qüestionaris incorpora un repte per a l'estudiant. Aquest tracta de resoldre les preguntes en un temps determinat, per això s'ha introduït un **temporitzador** per a cada pregunta.

3) Per a poder corregir les preguntes s'han de crear dos tipus de qüestionaris diferents: el primer només ha de tenir preguntes del tipus multiresposta per a poder realitzar una **correcció automàtica**; al segon qüestionari hi ha l'opció de tenir preguntes de tipus amb resposta oberta i/o amb imatge, l'autocorrecció d'aquestes preguntes no serà possible en aquest projecte, no obstant, es deixa l'opció de poder realitzar-ho en un futur com a una nova funcionalitat.

CAPÍTOL 2. ARQUITECTURA DEL SOFTWARE

Aquest segon capítol tracta de l'**arquitectura de software** [4] desenvolupada en aquest projecte. Primer es defineix el projecte, seguidament s'expliquen els passos que han de realitzar els usuaris per interactuar amb l'eina i finalment es defineix l'arquitectura dissenyada.

2.1. Definició del projecte

Primer de tot, explicar que aquest projecte s'ha dissenyat i introduït dins d'una arquitectura de software ja desenvolupada, la qual és un projecte final de màster, és a dir, el que s'ha realitzat es introduir-hi un nou mòdul.

Aquest projecte és basa en una arquitectura de software destinada als professors i estudiants dins un entorn educatiu, tant per a les escoles o universitats, en què, es poden realitzar qüestionaris de diferents tipus (multiresposta, pregunta amb resposta oberta i pregunta amb imatge) i amb un cert repte (inclou un temporitzador per pregunta a realitzar). Aquesta eina de treball, els usuaris la poden utilitzar en diferents plataformes com poden ser ordenadors, tabletes i telèfons mòbils.

L'arquitectura està dissenyada per a què els estudiants la usin des d'una **Aplicació Mòbil**[3] compatible en Android i iOS i, els professors des d'un **Panell d'Administració**[4] web accessible des de qualsevol sistema operatiu, com per exemple Windows i Linux. En l'aplicació mòbil els estudiants podran demanar i realitzar els qüestionaris que els professors hagin creat anteriorment en el Panell d'Administració.

La interacció entre els estudiants i els professors la uneix una **arquitectura orientada a serveis**[5] que inclou una Interfície de Programació d'Aplicacions (API) [5], que conté tota la informació que permet realitzar la comunicació entre els usuaris de l'eina. En la figura 2.1 es mostra l'arquitectura del projecte.

[3] Consisteix en una aplicació informàtica creada amb la finalitat de ser executada en smartphones i tablets, amb això permet a l'usuari realitzar alguna tasca determinada. Permet la interacció de l'estudiant al sistema en el lloc on es produeix l'acció.

[4] És la interfície gràfica que permet als usuaris veure i manipular dades en el sistema. Els professors hi realitzen operacions a llarg termini o complexes, com ara crear, eliminar i visualitzar els qüestionaris.

[5] Arquitectura que s'encarrega de gestionar les dades entre tots els canals de comunicació entre els usuaris. Aquesta s'ocupa de totes les interaccions de l'usuari amb el Back-End.

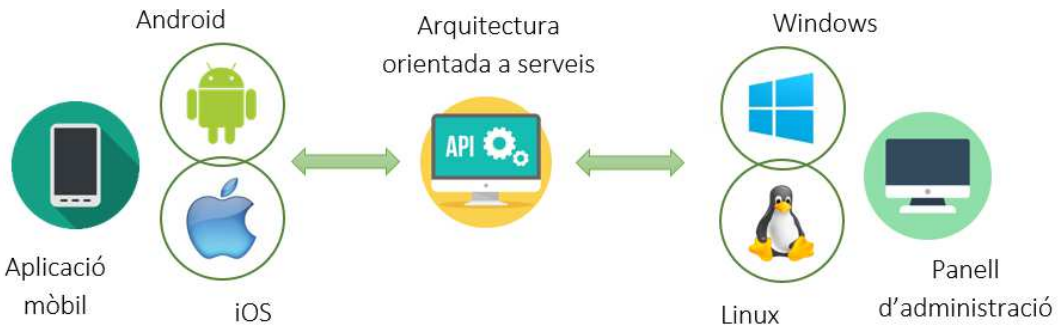


Figura 2.1 Arquitectura del projecte

2.2. Passos a realitzar pels usuaris

Tant els estudiants com els professors han de realitzar una sèrie de passos per a poder interactuar amb l'eina. Aquests passos defineixen l'escenografia del comportament de l'usuari.

2.2.1. Professors

Una vegada els professors han iniciat sessió en el Panell d'Administració web des de qualsevol ordenador, accediran a la pàgina dels qüestionaris on allí visualitzaran tots els que han creat, també podran generar-ne de nous i eliminar-los. Quan ja s'ha creat el qüestionari es genera un codi identificador per a què els estudiants puguin obtenir-lo. Finalment, quan el professor accedeixi dins d'un qüestionari es mostraran totes les respostes dels estudiants i els resultats corresponents. En la figura 2.2 es mostren els passos a realitzar pels professors.

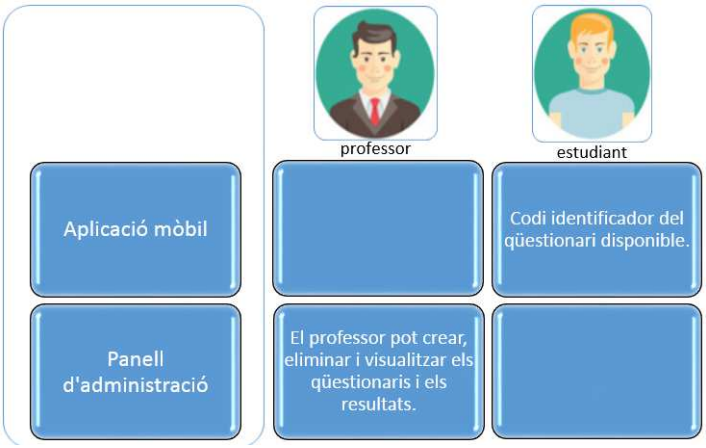


Figura 2.2 Passos a realitzar pels professors

2.2.2. Estudiants

De la mateixa manera, quan els estudiants han iniciat sessió en l'aplicació mòbil, accediran des del menú a l'apartat dels qüestionaris per a poder demanar-ne un i realitzar-lo. La manera d'obtenir-lo és amb un codi identificador que el professor facilita als estudiants. Una vegada realitzats els qüestionaris, els professors tindran accés a tota la informació d'aquests per a realitzar-hi la correcció. En la figura 2.3 es mostren els passos a realitzar pels estudiants.

Figura 2.3 Passos a realitzar pels estudiants

2.2.3. Respostes i resultats

Els professors poden visualitzar totes les respostes dels qüestionaris realitzats pels estudiants dins del Panell d'Administració. Des d'allí poden efectuar la correcció (amb resposta oberta) o visualitzar els resultats de les proves amb autocorrecció (amb multiresposta).

2.3. Arquitectura

En el subcapítol anterior s'ha definit el projecte i s'ha explicat l'escenografia del comportament dels usuaris, en aquest apartat es tracta de donar una visió general de tots els fragments que construeixen l'arquitectura de software del **Classpip** ja dissenyada anteriorment, per a poder entendre el seu funcionament i seguidament es dona una visió més profunda de les parts que s'han dissenyat en aquest projecte i introduït en l'arquitectura ja creada.

En la següent figura 2.4 es mostra l'esquema complert de tots els fragments que formen part del projecte i la tecnologia usada. Aquesta imatge s'ha extret de la memòria del projecte Classpip.

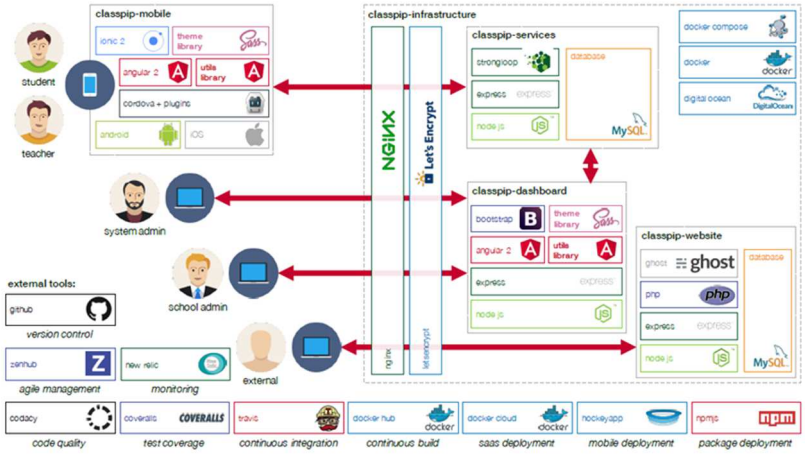


Figura 2.4 Arquitectura software Classpip

Tal com es pot veure, les quatre peces principals de l'arquitectura són: plataforma mòbil, espai de serveis, panell d'administració i un espai web. Des de l'aplicació mòbil els estudiants poden accedir a la infraestructura de serveis per a poder obtenir o guardar la informació dels qüestionaris.

D'altra banda, els professors poden comunicar-se des del panell d'administració a la infraestructura de serveis, per a poder guardar els qüestionaris creats i obtenir la informació dels qüestionaris realitzats i els resultats. Finalment, l'arquitectura també disposa d'un espai web per a finalitats de màrqueting i publicitat.

2.4. Components de l'arquitectura

En el subcapítol anterior s'ha vist de manera general tots els components que formen part de l'arquitectura. Ara és el moment de veure de forma més detallada el funcionament de les tres peces principals, en les que s'ha introduït les noves parts dissenyades en aquest projecte.

2.4.1. Arquitectura orientada a serveis

Aquesta és la peça clau de l'arquitectura on s'emmagatzema tota la informació d'aquesta eina i la que s'ocupa d'interconnectar els usuaris.

La Interfície de Programació d'Aplicacions (API) va ser dissenyada en codi obert amb **StrongLoop** [6], en què, es va generar amb **JavaScript**[6] sota el servidor

[6] JavaScript és un llenguatge de programació orientat a objectes que s'utilitza com a part d'un navegador web i que permet realitzar millores en la interfície d'usuari.

web **Express** [7] en l'entorn **Node.js** [8]. D'altra banda, tota la informació es guardava a una base de dades **MySql** [9] de manera persistent.

Figura 2.5 Logotips de l'Arquitectura Orientada a Serveis

En aquesta part de l'arquitectura s'ha introduït un nou disseny per a poder emmagatzemar tota la informació relacionada amb els qüestionaris que creen els professors, les respostes a aquests per part dels alumnes i els seus resultats.

2.4.2. Aplicació mòbil

Aquesta peça de l'arquitectura estava orientada per a què els estudiants i els professors poguessin interactuar entre ells. Per una banda, els professors premiaven als alumnes amb punts i medalles per les tasques realitzades per part dels alumnes. Per altra banda, els alumnes podien consultar aquests premis a través d'aquesta plataforma.

Aquesta aplicació es va dissenyar amb **Ionic 2** [10], **Angular 2** [11] i **Apache Cordova** [12]. Aquest últim permetia la conversió de la plataforma nativa a **Android** i **iOS**. Conjuntament es van utilitzar connectors i llibreries per a la correcta capacitat del telèfon.

Figura 2.6 Logotips de l'Aplicació Mòbil

En aquesta peça s'ha introduït tot el nou disseny per a què els estudiants puguin obtenir, resoldre i enviar els qüestionaris creats pels professors.

2.4.3. Panell d'administració

Aquest espai de l'arquitectura es va dissenyar per a què els professors poguessin interactuar directament amb l'API per gestionar els grups d'estudiants d'una assignatura i obtenir tota la informació de la plataforma.

Aquesta aplicació es va crear amb **Angular 2** i el disseny de la interfície de l'usuari amb **Bootstrap** [13]. A través del servidor web **Express** en l'entorn **Node.js** els usuaris podien visualitzar i interactuar amb l'eina.

Figura 2.7 Logotips del Panell d'Administració

En aquest espai s'ha introduït un nou apartat de disseny per a què els professors puguin visualitzar, crear i eliminar qualsevol dels seus qüestionaris. També es mostra tota la informació dels qüestionaris realitzats i els resultats de l'autocorrecció. Per al disseny de la interfície de l'usuari s´ha introduït el component **Angular Material**[7] [14].

2.4.4. Eines de suport externes

Ara que ja s'ha donat una visió de les principals parts de l'arquitectura és hora de mostrar les eines que donen suport a aquesta, per a poder tenir un correcte desenvolupament i realitzar les proves necessàries que garanteixen el seu apropiat funcionament.

El disseny de les tres parts de l'arquitectura **aplicació mòbil**, **panell d'administració** i **arquitectura orientada a serveis** estan guardats amb tres repositoris[8] [15] diferents a l'eina de control **GitHub** [16].

Per a totes les funcionalitats creades es realitza la corresponent prova amb **Coveralls** [17]. Aquesta eina verifica la qualitat, la seguretat i la duplicació del codi mitjançant **Codacy** [18].

Per a garantir la correcta robustesa del programari s'utilitza **HockeyApp** [19] per a la distribució beta de mòbils i informes de fallides.

[7] Eina de components i de disseny d'Angular.
[8] Lloc centralitzat on es guarden dades, habitualment bases de dades o arxius informàtics.

CAPÍTOL 3. DESENVOLUPAMENT DEL SOFTWARE

En el capítol anterior s'han explicat tots els fragments que construeixen l'arquitectura de software del Classpip i les parts que s'han dissenyat en aquest projecte. En aquest nou capítol es mostra la metodologia que s'ha seguit per a l'organització del projecte, també s'explica el procés de desenvolupament del programari, les proves de validació que es duen a terme, com es construeixen les aplicacions en aquesta arquitectura dissenyada i la seguretat d'aquestes.

3.1. Organització del projecte

Amb l'objectiu d'obtenir una bona organització del projecte s'ha utilitzat l'organitzador de projectes **Gantt project** [20] que utilitza la tècnica del **diagrama de Gantt**[9]. Aquests es pot veure en la següent figura 3.1 i s'observa l'evolució del desenvolupament que s'ha obtingut en la realització d'aquest projecte.

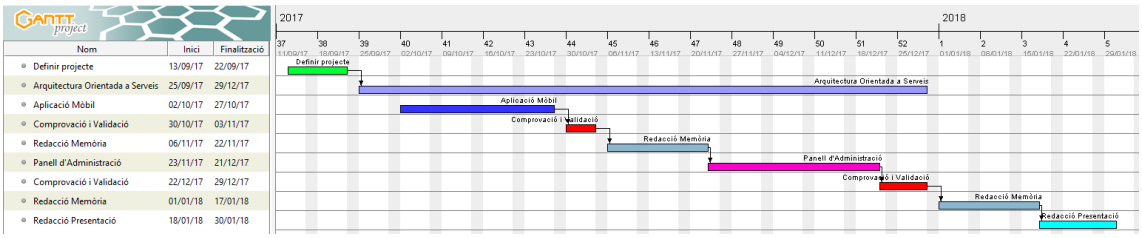


Figura 3.1 Diagrama de Gantt

En la part esquerra de la imatge anterior s'observen totes les tasques realitzades amb les corresponents dates d'inici i fi i, a la part dreta de la imatge hi ha les durades de les tasques en forma de diagrama de barres, en què, es pot veure un procés seqüencial gairebé en totes les tasques, és a dir, fins a què no s'ha finalitzat una tasca no en pot començar-ne una altra.

Les tasques que s'han realitzat en aquest projecte han estat les següents:

- **Definir projecte**: principals idees del projecte, definir l'estructura de l'arquitectura de software i l'organització del projecte.
- **Arquitectura Orientada a Serveis**: definir les relacions de les Classes.
- **Aplicació Mòbil**: crear les Classes, mètodes, serveis i plataforma.
- **Comprovació i Validació**: realitzar les proves corresponents que validen el correcte funcionament.
- **Redacció Memòria**: redactar les parts realitzades anteriorment.

[9] Eina on es representa el temps previst per a les diferents activitats a realitzar en un projecte.

- **Panell d'Administració**: crear les Classes, mètodes, serveis i plataforma.
- **Comprovació i Validació**: realitzar les proves corresponents que validen el correcte funcionament.
- **Redacció Memòria**: redactar les parts realitzades anteriorment.
- **Redacció Presentació**: redactar la presentació del projecte.

En la següent figura 3.2 es mostra l'organització del projecte amb el **diagrama de Pert**[10].

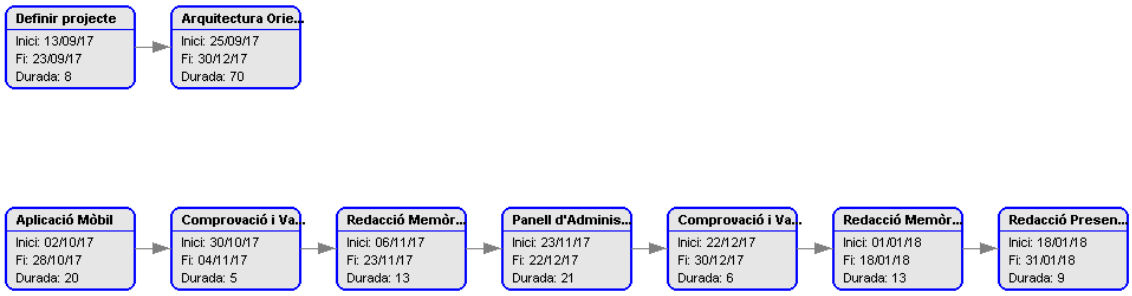


Figura 3.2 Diagrama de Pert

3.2. Desenvolupament del programari

Per a desenvolupar el programari de l'arquitectura de software d'aquest projecte s'ha utilitzat el codi del projecte Classpip ja dissenyat i que està en el dipòsit compartit a la plataforma GitHub. Una vegada finalitzat el present projecte, aquest s'emmagatzema altra vegada al dipòsit compartit.

GitHub permet verificar i detectar si hi ha algun problema en el programari a través d'una compilació automatitzada. Aquesta comprovació es realitza amb el servidor d'integració continua **Travis CI** [21], en què, executa uns scripts del fitxer de configuració .travis.yml que comproven totes les etapes de la configuració. En aquest procés també es comprova la **Qualitat del Codi** i la **Cobertura de Proves** amb **Coveralls** [22].

3.3. Proves de validació

Durant tot el procés del desenvolupament del projecte s'han creat unes proves d'unitat addicional per assegurar-se que la lògica del constructor funciona com s'espera. Aquestes proves es generen en el fitxer .spec.ts i s'executen amb el corredor de proves **Karma.js** [23].

[10] Representació gràfica de les relacions entre les tasques del projecte.

Karma s'ocupa de testejar els tests de **Javascript** a mesura que es vagin construint, això ajuda a què per a qualsevol decisió el desenvolupador es donarà compte immediatament. Karma es configura amb Jasmine.js com framework de testing llevat que es decideixi el contrari.

3.4. Construcció de les aplicacions

Una vegada dissenyades les aplicacions, per a què els estudiants puguin realitzar els qüestionaris i per a què els professors els creïn, i haver verificat el seu correcte funcionament es procedeix a la seva construcció a través d'una compilació externa de **Travis CI**. Hi ha diferents mètodes de compilació segons el tipus d'aplicació creada.

3.4.1. Construcció aplicació mòbil

Per a poder oferir **l'Aplicació Mòbil** als estudiants en qualsevol sistema operatiu com ara **Android** i **iOS**, primer es construeix l'aplicació en local i després s'executen característiques de **Cordona** i **Ionic 2** que permeten la generació de la plataforma corresponent.

3.4.2. Construcció aplicació Web

El **Panell d'Administració** per als professors és l'aplicació Web d'aquest projecte, la qual té un fitxer de configuració **dockerized**[11] que permet generar l'aplicació a través de les ordres que en conté.

3.5. Seguretat

Per a l'ús del Panell d'Administració Web, en què, els professors creen els qüestionaris que seran resolts pels estudiants, l'arquitectura de software Classpip ja disposa de la configuració d'un **certificat HTTPS**[12] que garanteix la seguretat de l'aplicació. L'algorisme per a la signatura que també està implementat és **SHA-256** [24] amb encriptació **RSA**[13], aquest admet la seguretat de la **capa de transport (TLS)** 1.2 [25] sobre HTTP amb la cèl·lula **AES128** [26].

[11] Abstracció i automatització de la virtualització de nivell de sistema operatiu.
[12] Capçalera d'URI que indica una connexió segura.
[13] Sistema criptogràfic de clau pública.

CAPÍTOL 4. APLICACIÓ MÒBIL

En l'anterior capítol s'ha mostrat l'organització del projecte, com es desenvolupa el programari, les proves de validació que se li apliquen i la construcció de les aplicacions d'aquest projecte. En aquest capítol es mostra l'Aplicació Mòbil que s'ha dissenyat per aquest projecte i alguns exemples de com usar-la.

4.1. Introducció

Tal com s'ha explicat al principi, l'objectiu principal d'aquest projecte es desenvolupar una eina de motivació d'estudi per als estudiants i que aquesta s'utilitza a través dels telèfons mòbils. Els estudiants podran realitzar els tres diferents tipus de formats de qüestionari i des de qualsevol plataforma **Android** i **iOS**.

Per a desenvolupar les plataformes Android i iOS s'ha utilitzat l'**aplicació híbrida**[14] en base tecnologies web que ja està incorporada en l'arquitectura Classpip i que seguidament s'explica. Aquesta aplicació es compon de dos **frameworks** [27] i una plataforma que permeten la conversió als diferents sistemes.

També s'explica el model de **Model-Vista-Controlador** [28] (**MVC**) que s'ha usat per a desenvolupar l'aplicació d'aquest projecte i finalment, es mostrar la compatibilitat que se l´hi ha donat sent una **plataforma multiidiomes**.

4.2. Aplicació híbrida

La utilització d'aquesta aplicació híbrida permet desenvolupar el codi dels qüestionaris en una tecnologia Web i després visualitzar-la en un navegador que incorpora la pròpia aplicació. Una vegada desenvolupat el codi s'utilitza la plataforma **Cordova** que facilita la conversió al codi **Android** i **iOS**.

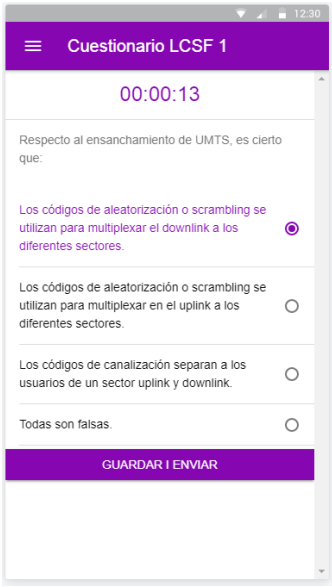
4.2.1. Framework Ionic

Ionic [29] és l'eina que s'ha utilitzat en aquest projecte per a realitzar el format i l'aparença dels qüestionaris, aquesta aparença ha estat possible a través d'uns estils incorporats al **llenguatge HTML** [30]. Tal i com es va explicar al principi, els tres objectius específics són: els qüestionaris contenen tres tipus de preguntes diferents (**test**, **textArea** i **image**); cadascun dels qüestionaris incorpora un temporitzador per a cada pregunta; les preguntes del tipus

[14] Aplicació de disseny utilitzant pàgines web.

multiresposta es pot realitzar una correcció automàtica. A continuació es mostren les imatges i part del codi desenvolupat per als tres tipus de preguntes i el temporitzador.

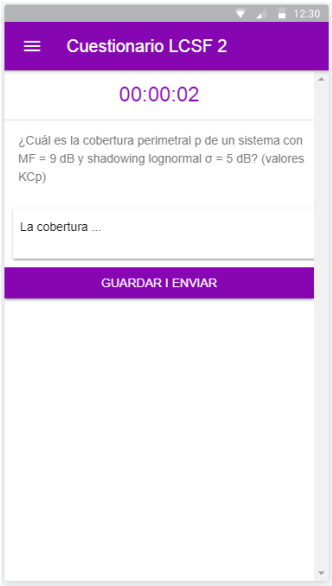
Pregunta multiresposta (**test**) en la figura 4.1:




```
<form (submit)="doSubmitAnswer($event)" [formGroup]="questionForm">
<ion-list text-wrap radio-group formControlName="questionsSend">
<ion-list-header>
{{ questionsSend.name }}
</ion-list-header>
<ion-item>
<ion-label >{{ questionsSend.answer[0].name }}</ion-label >
<ion-radio  value="{{ questionsSend.answer[0].name }}"
checked="false"></ion-radio>
</ion-item>
<ion-item>
<ion-label >{{ questionsSend.answer[1].name }}</ion-label >
<ion-radio  value="{{ questionsSend.answer[1].name }}"></ion-radio>
</ion-item>
<ion-item>
<ion-label >{{ questionsSend.answer[2].name }}</ion-label >
<ion-radio  value="{{ questionsSend.answer[2].name }}"></ion-radio>
</ion-item>
<ion-item>
<ion-label >{{ questionsSend.answer[3].name }}</ion-label >
<ion-radio  value="{{ questionsSend.answer[3].name }}"></ion-radio>
</ion-item>
<button ion-button full type="submit" id="click"
[disabled]="!questionForm.value.questionsSend">{{
'QUESTIONNAIRE.SUBMIT' | translate }}</button>
</ion-list>
</form>
```

Figura 4.1 Questionnaire.html

Pregunta amb resposta oberta (**textArea**) en la figura 4.2:




```
<form (submit)="doSubmitAnswer($event)" [formGroup]="questionForm">
    <ion-list text-wrap formControlName="questionsSend">
      <ion-list-header>
        {{ questionsSend.name }}
      </ion-list-header>
      <ion-card text-wrap>
        <ion-textarea text-wrap value=""></ion-textarea >
      </ion-card>
      <button ion-button full type="submit"
[disabled]="!questionForm.value.questionsSend">{{
'QUESTIONNAIRE.SUBMIT' | translate }}</button>
    </ion-list>
</form>
```

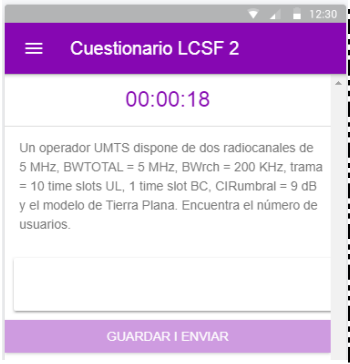
Figura 4.2 QuestionnaireTextArea.html

Pregunta amb resposta oberta i imatge (**image**) en la figura 4.3:


```
<form (submit)="doSubmitAnswer($event)" [formGroup]="questionForm">
    <ion-card text-wrap formControlName="questionsSend">
      <img src = "{{ questionsSend.image }}"/>
      <ion-list-header>
        {{ questionsSend.name }}
      </ion-list-header>
      <ion-textarea text-wrap value="">
      </ion-textarea >
      <button ion-button full type="submit"
[disabled]="!questionForm.value.questionsSend">{{
'QUESTIONNAIRE.SUBMIT' | translate }}</button>
    </ion-card>
 </form>
```

Figura 4.3 QuestionnaireImage.html

Temporitzador de cada pregunta en la figura 4.4:




```
<div>
  <ion-item class="no-bottom-border item">
    <h1 text-center color="primary" class="timer-button timer-
text">{{timer.displayTime}}</h1>
  </ion-item>
</div>
```

Figura 4.4 Questionnaire.html

En l'annex 7.1 es pot veure la resta del codi del temporitzador de l'aplicació mòbil d'aquest projecte.

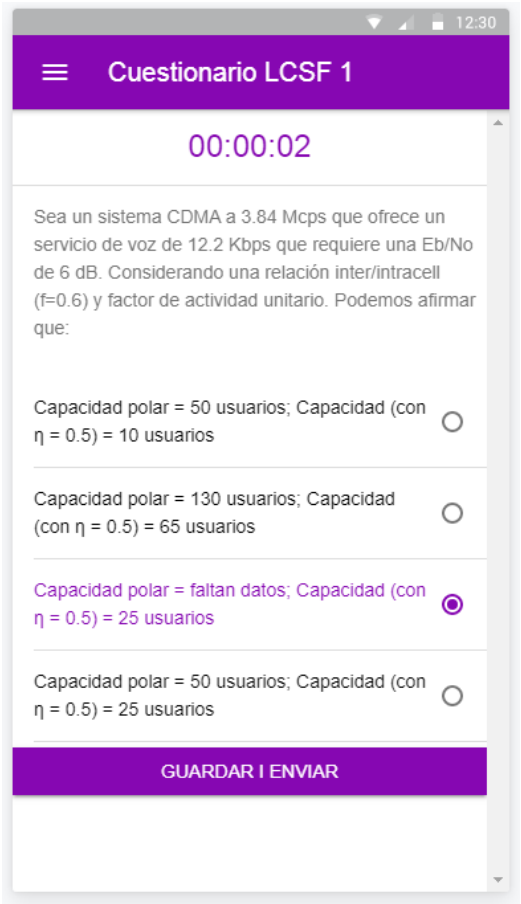
4.2.2. Framework Angular

Per a què l'aparença dels qüestionaris funcioni es necessita crear una lògica a l'aplicació, en què, Ionic utilitza el framework **d'Angular** [31]. Aquesta lògica es desenvolupa en el **llenguatge Typescript** [32]. En aquest projecte s'han dissenyat els **models**, les **pàgines**, els **mòduls** i els **serveis** de l'aplicació.

Per al desenvolupament dels components anteriors nombrats com ara les pàgines, es necessiten quatre fitxers diferents, els quals s'anomenen a continuació.

- **.html** (per a l'aparença dels qüestionaris amb **Ionic**). Figura 4.5
- **.scss** (plantilla de disseny per a la pàgina del qüestionari amb **Angular**). Figura 4.5
- **.spect.ts** (per a realitzar les proves de test amb **Angular**). Figura 4.5
- **.ts** (conté la **Classe** i els **mètodes** que maneges la pàgina del qüestionari amb **Angular**). A continuació de la Figura 4.5

Com a exemple, a continuació es mostra una part del codi Angular dels tres fitxers.




```
page-questionnaire {
  .card-ios,
  .card-md {
    width: 100%;
  }

  .card-content-md,
  .card-content-ios {
    padding: 3px 5px;
  }
}
```

```
let fixture: ComponentFixture<QuestionnairePage> =
null;
/* tslint:disable */
let instance: any = null;
/* tslint:enable */

describe('Pages: QuestionnairePage', () => {

  beforeEach(async(() =>
TestUtils.beforeEachCompiler([QuestionnairePage]).
then(compiled => {
    fixture = compiled.fixture;
    instance = compiled.instance;
  })));

  it('should create the QuestionnairePage',
async(() => {
    expect(instance).toBeTruthy();
  }));
});
```

Figura 4.5 Imatge Questionnaire

```
@Component({
  selector: 'page-questionnaire',
  templateUrl: './questionnaire.html'
})
export class QuestionnairePage {

  public myQuestionnaire: Questionnaire;
  public questions: Array<Question>;
  public questionsAnswers: Array<Question>
  public myCredentials: Credentials;
  questionForm;
  public timeInSeconds: number;
  public timer: PTimer;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {

    this.questionForm = new FormGroup({
       "questionsSend": new FormControl({value: this.questionsSend, disabled:
false})
    });

    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.questions = this.navParams.data.questions;

  }

  public ionViewDidEnter(): void {
    this.ionicService.removeLoading();
  }

  private getQuestions(refresher?: Refresher): void {

this.questionnaireService.getMyQuestionnaireQuestions(this.myCredentials).fina
lly(() => {
      refresher ? refresher.complete() : null;
    }).subscribe(
      ((value: Array<Question>) => this.questions = value),
      error =>
this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }
...
}
```

Questionnaire.ts

4.2.3. Plataforma Cordova

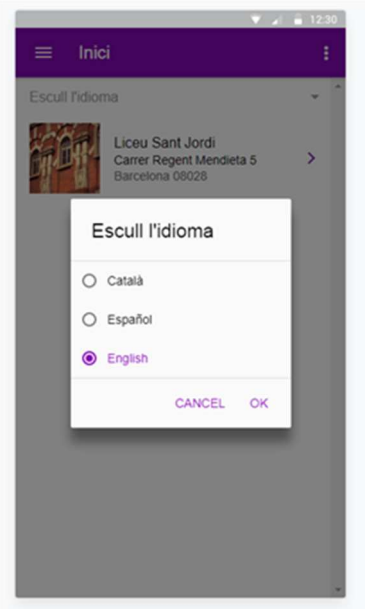
La peça clau per al desenvolupament de les aplicacions és la plataforma **Cordova** i que també es troba incorporada a l'arquitectura Classpip. Aquesta permet la conversió del codi generat utilitzant el framework d'**Angular** i el llenguatge **Typescript**, als llenguatges natius dels sistemes d'**Android** i **iOS**. Per tant, aquesta plataforma és de gran utilitat per als desenvolupadors d'aplicacions

mòbils perquè ajuda a la creació d'aquestes sense haver de dissenyar-les amb el llenguatge natiu.

4.3. Plataforma multiidiomes

L'aplicació mòbil d'aquest projecte a part d'haver-se desenvolupat per a plataformes **Android** i **iOS**, s'ha introduït un selector a la pàgina d'inici de l'arquitectura Classpip ja dissenyada anteriorment, per a poder escollir l'idioma de la plataforma. No obstant, també s'ha hagut de modificar el codi de la plataforma nativa Classpip per a què tota l'aplicació estigui disponible amb els diferents idiomes i no només el mòdul per a realitzar els qüestionaris. Per tant, els estudiants que hagin de realitzar els qüestionaris tenen l'opció de poder tenir els menús en **català**, **castellà** i **anglès**.

En la següent figura 4.6 es mostra la pàgina d'inici amb el selector d'idiomes.




```
<!-- LANGUAGE SELECTOR -->

  <ion-list>
    <ion-item>
      <ion-label>{{'HOME.LANGUAGE' translate }}</ion-label>
      <ion-select (ionChange)="choose($event)">
        <ion-option*ngFor="letitemofidioms" [value]="item.value">
          {{ item.label }}
        </ion-option>
      </ion-select>
    </ion-item>
  </ion-list>
```

Figura 4.6 Selector d'idiomes

En l'annex 7.2 es pot veure la resta del codi de la plataforma multiidioma de l'aplicació mòbil d'aquest projecte.

4.4. Arquitectura d'aplicació

En aquest subcapítol s'explica el patró de l'arquitectura de software que s'ha utilitzat per a la construcció d'aquesta aplicació, aquest és el **Model-Vista-**

Controlador (**MVC**). Aquest model es composa de tres components per a separar la visualització de la informació de la interacció del usuari. El **Model** i el **Controlador** serien la part de la interacció i la **Vista** la part de la representació.

En la següent figura 4.7 es mostra el diagrama que representa la relació entre el **Model**, la **Vista** i el **Controlador** per a entendre la lògica de **l'Aplicació Mòbil** d'aquest projecte.

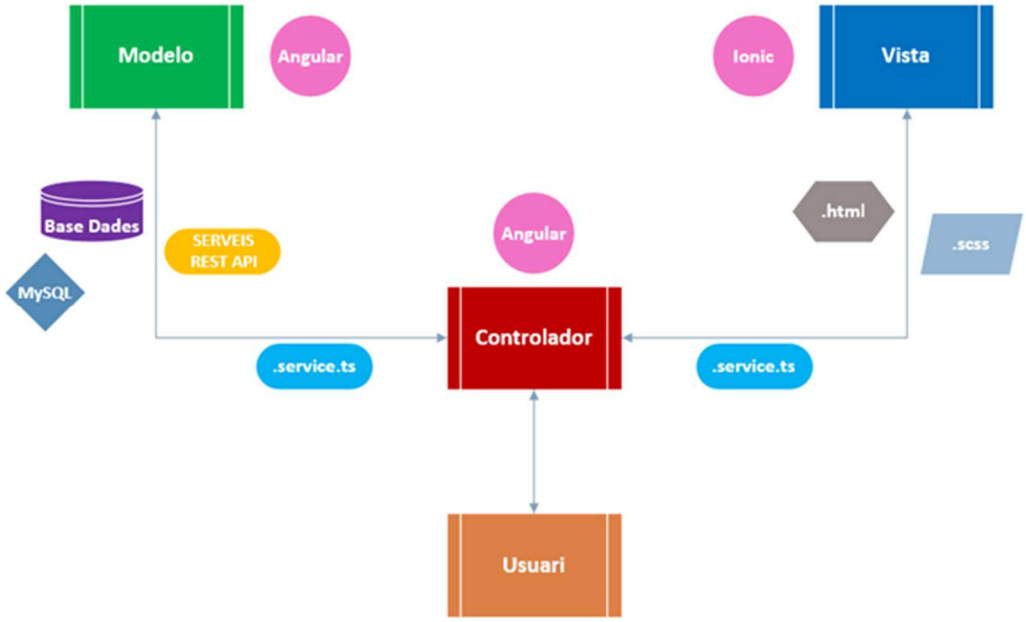


Figura 4.7 Diagrama de l'arquitectura de software

En els següents subapartats s'explica com a exemple d'un qüestionari, el patró de l'arquitectura de software d'aquest projecte.

4.4.1. Model

Tal com s'ha explicat al subcapítol anterior, per a la creació dels models s'ha utilitzat el llenguatge Typescript amb el framework d'Angular. Aquests models són els objectes de l'arquitectura dissenyada, que contenen les Classes de cadascun i que tota la informació que contenen es guarda a la Base de Dades. Cada Classe conté el constructor de l'objecte i els mètodes per a obtenir i/o modificar-lo.

A continuació es mostra el codi del model **Questionnaire** que s'ha desenvolupat en aquest projecte i també es mostra el codi del model **Question**.

```
export class Questionnaire {

  private _id: string;
  private _name: string;

  constructor(id?: string, name?: string) {
      this._id = id;
      this._name = name;
  }

  static toObject(object: any): Questionnaire {
    let result: Questionnaire = new Questionnaire();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
    }
    return result;
  }

  static toObjectArray(object: any): Array<Questionnaire> {
    let resultArray: Array<Questionnaire> = new Array<Questionnaire>();
    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(Questionnaire.toObject(object[i]));
      }
    }
    return resultArray;
  }

  public get id(): string {
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
  }

  public get name(): string {
    return this._name;
  }

  public set name(value: string) {
    this._name = value;
  }
}
```

Definició del model qüestionari (questionnaire.ts)

```
export class Question {
  private _id: string, _name: string, _type: string, _image: string, _time: number;
  private _answer: Array<Answer>, _correctAnswer: Array<CorrectAnswer>;

  constructor( id?: string, name?: string, type?: string, image?: string, time?: number )
{
    this._id = id;
    this._name = name;
    this._type = type;
    this._image = image;
    this._time = time;
  }

  static toObject(object: any): Question {
    let result: Question = new Question();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
      result.type = object.type;
      result.image = object.image;
      result.time = object.time;
    } return result;
  }
```

```
  static toObjectArray(object: any): Array<Question> {
    let resultArray: Array<Question> = new Array<Question>();
    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(Question.toObject(object[i]));
      } } return resultArray;
  }

  public get id(): string { return this._id; }

  public set id(value: string) { this._id = value; }

  public get name(): string { return this._name; }

  public set name(value: string) { this._name = value; }

  public get type(): string { return this._type; }

  public set type(value: string) { this._type = value; }

  public get image(): string { return this._image; }

  public set image(value: string) { this._image = value; }

  public get time(): number { return this._time; }

  public set time(value: number) { this._time = value; }

  public get answer(): Array<Answer> { return this._answer; }

  public set answer(value: Array<Answer>) { this._answer = value; }

  public get correctAnswer(): Array<CorrectAnswer> { return this._correctAnswer; }

  public set correctAnswer(value: Array<CorrectAnswer>) { this._correctAnswer = value; }
```

Definició del model pregunta (question.ts)

En l'annex 7.3 es poden veure la resta dels models de l'aplicació mòbil d'aquest projecte.

4.4.2. Vista

El framework d'**Ionic** és el que s'utilitza en aquest projecte per a donar format i aparença a l'aplicació dels qüestionaris. En el subcapítol anterior s'ha mostrat part del codi desenvolupat per als tres tipus de preguntes i el temporitzador i, que estan formats amb els estils iònics sobre el llenguatge **HTML**.

En la següent figura 4.8 es mostren les dues plataformes compatibles per als usuaris d'aquesta aplicació. La part esquerra representa el sistema **Android** i a la dreta el sistema **iOS**.

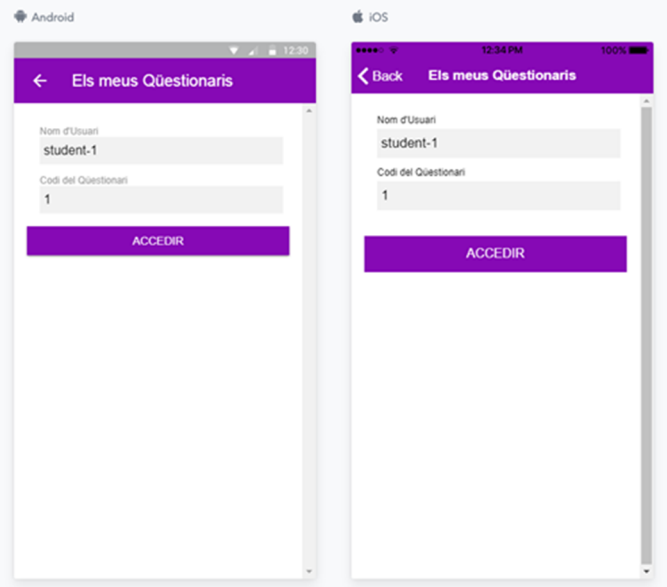


Figura 4.8 Plataformes Android i iOS

4.4.3. Controlador

El **Controlador** és la part clau per a què la lògica de l'aplicació funcioni. Aquest és l'encarregat de recollir les peticions dels usuaris i la informació que rep de la **Base de Dades** per a ser enviada a la pantalla.

Aquesta lògica es desenvolupa en el fitxer .ts que conté la **Classe** i els **mètodes** que manegen la pàgina amb **Angular**. Dins de la Classe es realitzen les peticions als serveis angulars encarregats d'obtenir les dades de la **API**, aquests serveis també anomenats proveïdors estan creats amb el llenguatge **TypeScript** i hi ha un servei per a cada objecte, d'aquesta manera és possible obtenir tota la informació de cada model.

A continuació es mostra primer la part del codi de la **Classe Questionnaire** on es realitza la petició al servei i després la part del codi del proveïdor que consulta les dades a l'objecte en la **API**.

```
export class QuestionnairePage {
  constructor(public questionnaireService: QuestionnaireService){}

  private getQuestions(refresher?: Refresher): void {
  this.questionnaireService.getMyQuestionnaireQuestions(this.myCredentials).finally(()
=> {
      refresher ? refresher.complete() : null;
    }).subscribe(
      ((value: Array<Question>) => this.questions = value),
      error => this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }}
```

Classe Qüestionari (Questionnaire.ts)

```
export class QuestionnaireService {

  constructor(){}

  public getMyQuestionnaireQuestions(credentials: Credentials):

Observable<Array<Question>> {

    var ret: Array<Question> = new Array<Question>();

    return Observable.create(observer => {

      this.getQuestionnaireQuestions(credentials).subscribe(

        questions => {

          questions.forEach(question => {

            this.getQuestionAnswers(question.id).subscribe(

              answers => {

                question.answer = answers;

                      ret.push(question);

                      if (ret.length === questions.length) {

                        observer.next(ret);
                        observer.complete();

                      }

              }, error => observer.error(error))

          });

        }, error => observer.error(error)
      )
    });
  }
}
```

Proveïdor del Qüestionari (Questionnaire.service.ts)

En l'annex 7.4 es poden veure la resta de les classes de l'aplicació mòbil d'aquest projecte.

4.5. Captures de pantalla

En aquest subcapítol es mostren algunes de les imatges de l'aplicació mòbil realitzada. En l'annex 7.5 es mostra el flux complert d'imatges de l'aplicació mòbil d'aquest projecte.

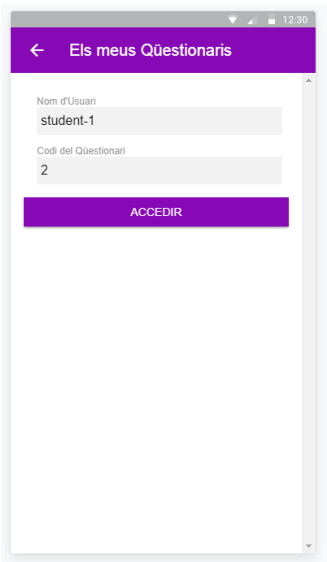


Figura 4.9 Get Questionnaire

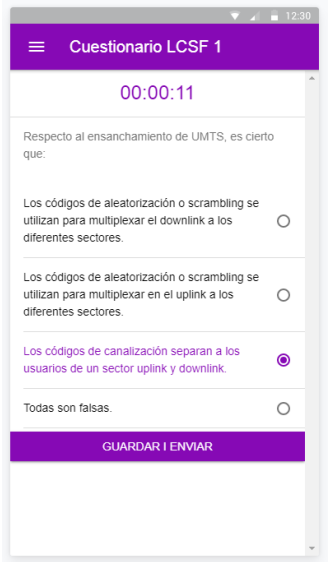


Figura 4.10 Question test

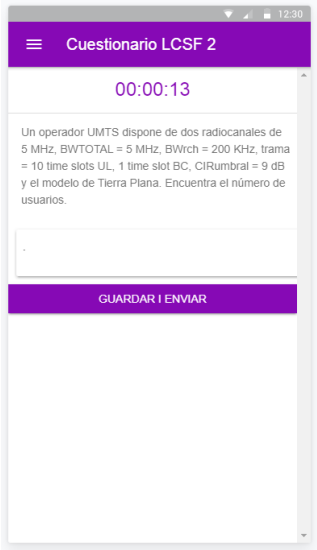


Figura 4.11 Question textArea

Figura 4.12 Question image

CAPÍTOL 5. PANELL D'ADMINISTRACIÓ

En el capítol anterior s'ha explicat l'**Aplicació Mòbil** dissenyada per aquest projecte. En aquest capítol es mostra el **Panell d'Administració** que es va desenvolupar per al client i alguns exemples de com usar-lo.

5.1. Introducció

El **Panell d'Administració** és l'eina que es va desenvolupar a la plataforma **Classpip** per a què l'utilitzessin els professors i l'administrador de la universitat. En aquesta plataforma si ha afegit una nova secció que permet **crear**, **eliminar** i **visualitzar** els qüestionaris. No obstant, aquesta eina també permet visualitzar les **respostes** i els **resultats** dels estudiants que han realitzat els qüestionaris.

En aquest capítol s'explica la transformació de la plataforma existent en una de nova que és compatible amb diferents idiomes. Seguidament es parla de l'arquitectura d'aquest **Panell d'Administració** on també s'ha transformat la interfície de l'usuari existent per una de nova més ràpida i consistent. Finalment, es mostren algunes captures de pantalla visualitzant el seu funcionament.

5.2. Plataforma multiidiomes

La plataforma existent que utilitzen els professors només estava dissenyada en un sol idioma (castellà), actualment aquest **Panell d'Administració** ha estat modificat per a què tots els seus usuaris puguin escollir un dels tres idiomes que s'han introduït. Això significa que s'ha hagut de modificar el codi de la plataforma nativa **Classpip** per a què tota l'aplicació estigui disponible amb els diferents idiomes i no només la secció dels qüestionaris.

S'ha introduït un selector a la pàgina d'inici de l'eina per a poder escollir l'idioma de la plataforma, d'aquesta manera hi ha la possibilitat de tenir els menús en català, castellà i anglès.

A continuació es mostra part del codi que permet la visualització del selector d'idiomes.

```
<div class="home-content">
  <mat-card-content>
    <button id="button" mat-raised-button (click)="choose()">{{ 'HOME.LANGUAGE' |
translate }}</button>
  </mat-card-content>
</div>
```

Pàgina d'inici (home.html)

En la següent figura 5.1 es mostra la pàgina d'inici de la plataforma nativa amb el selector d'idiomes que s'ha introduït per a què tota la plataforma ja existent pugui ser escollida pels diferents idiomes.

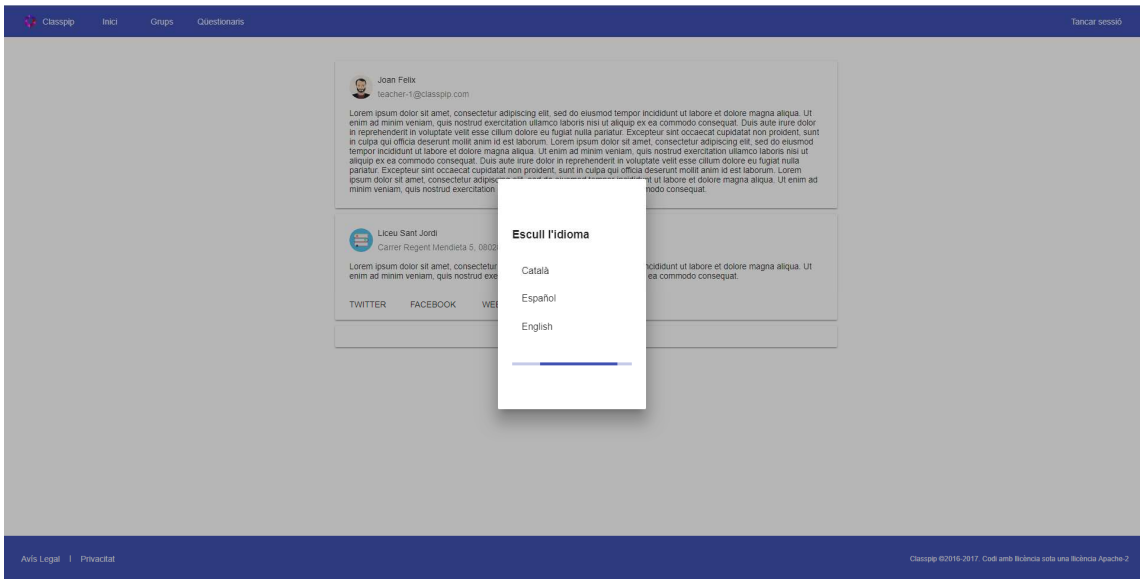


Figura 5.1 Selector de l'idioma

En l'annex 7.6 es pot veure la resta del codi de la plataforma multiidioma del **Panell d'Administració** d'aquest projecte.

5.3. Arquitectura Panell d'Administració

En aquest subcapítol s'explica la nova arquitectura de software dissenyada en el **Panell d'Administració**. La nativa plataforma **Classpip** està construïda utilitzant el **Contenidor d'Angular 2** i el servidor Web **Express** per a mostrar-la, tot ell dins del **Contenidor Node JS** que s'encarrega d'executar les dues aplicacions.

Aquesta eina és accessible per als usuaris des de qualsevol ordinador i plataforma mòbil perquè s'executa a través de la web. L'anterior **Panell d'Administració** estava construït amb el framework **Bootstrap UI**[15] a l'estil de **Front-End** [40].

Actualment, s'ha transformat la interfície de l'usuari existent de la nativa plataforma **Classpip** per una de nova més ràpida, consistent i versàtil. S'ha substituït el framework **Bootstrap UI** pel framework **Angular Material**[16]. Aquest treballa amb components d'interfície complerts i moderns que funcionen a la Web, al mòbil i a l'escriptori.

[15] Bootstrap és un joc d'eines de codi obert per al desenvolupament amb HTML, CSS i JS.
[16] Components de disseny de materials per a angular.

En els següents subapartats s'explica la forma de crear, eliminar, visualitzar els qüestionaris, les respostes i els resultats dels estudiants que els han realitzat.

En la següent figura 5.2 es mostra la pàgina principal dels qüestionaris, on es visualitza el llistat dels qüestionaris existents (només els propis del professor que ha iniciat sessió) amb el seu nom i el seu número d'identificador, el botó per a eliminar un qüestionari i el botó per a crear-ne un de nou.

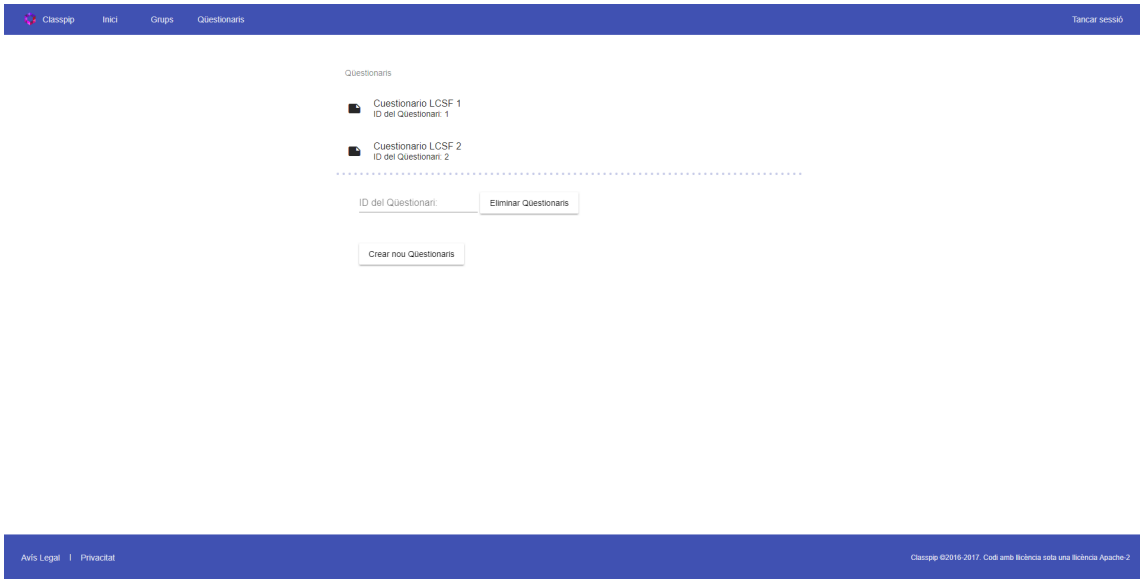


Figura 5.2 Pàgina dels qüestionaris (questionnaires.html)

5.3.1. Crear qüestionari

En la nova secció anomenada Qüestionaris hi ha el botó que permet crear un nou qüestionari. Tal i com es va explicar al principi, un dels tres objectius específics és que els qüestionaris contenen tres tipus de preguntes diferents (**test**, **textArea** i **image**), per tant, en aquest subapartat es mostra la manera de crear els tres tipus de preguntes.

En la primera pàgina s'ha d'introduir el nom del qüestionari, el temps total (en segons) que el professor vol que duri el qüestionari (aquest es reparteix per parts iguals per a cadascuna de les pregunta), el nombre total de preguntes que tindrà, seleccionar el tipus de qüestionari segons si es tracta de respostes obertes o múltiples i la data en què s'està creant el qüestionari.

A continuació es mostra part del codi de la primera pàgina i en la figura 5.3 es visualitza la pàgina.

```
<div class="createQuestionnaire-content">
  <mat-form-field hintLabel="{{ 'QUESTIONNAIRE.TEXT3' | translate }}">
    <input matInput #input maxlength="40" [(ngModel)]="name" placeholder="{{
'QUESTIONNAIRE.TEXT4' | translate }}">
    <mat-hint align="end">{{input.value?.length || 0}}/40</mat-hint>
  </mat-form-field>

  <mat-form-field hintLabel="{{ 'QUESTIONNAIRE.TEXT5' | translate }}">
    <input matInput #input maxlength="4" [(ngModel)]="time" placeholder="{{
'QUESTIONNAIRE.TEXT6' | translate }}">
  </mat-form-field>

  <mat-form-field hintLabel="{{ 'QUESTIONNAIRE.TEXT7' | translate }}">
    <input matInput #input maxlength="100" [(ngModel)]="number" placeholder="{{
'QUESTIONNAIRE.TEXT8' | translate }}">
  </mat-form-field>

  <mat-form-field>
    <mat-select [(value)]="selected" placeholder="{{ 'QUESTIONNAIRE.TEXT9' | translate
}}">
      <mat-option value="optionRespuestaAbierta">{{ 'QUESTIONNAIRE.TEXT10' | translate
}}</mat-option>
      <mat-option value="optionRespuestaMultiple">{{ 'QUESTIONNAIRE.TEXT11' | translate
}}</mat-option>
    </mat-select>
    <mat-hint align="end">{{ 'QUESTIONNAIRE.TEXT1' | translate }}</mat-hint>
  </mat-form-field>

  <form>
    <input type="date" name="bday" [(ngModel)]="date">
  </form>

  <button mat-raised-button (click)="createQuestionnaire()">{{ 'QUESTIONNAIRE.NEXT' |
translate }}</button>
  <button mat-raised-button (click)="cancel()">{{ 'QUESTIONNAIRE.BACK' | translate
}}</button>
</div>
```

Primera pàgina (createQuestinnaire.html)

Figura 5.3 Primera pàgina per crear qüestionari

Una vegada omplerta la primera pàgina de la creació del qüestionari, la següent que apareix és segons el tipus de resposta escollida en la primera pàgina.

En el cas de que l'elecció hagi estat la de resposta oberta, en les següents pàgines s'ha d'introduir les preguntes que formaran el qüestionari. En aquestes s'ha d'introduir la pregunta, seleccionar si la pregunta és amb resposta oberta o si la pregunta conté una imatge i introduir la URL de la imatge si escau.

A continuació es mostra part del codi de la segona pàgina i en la figura 5.4 es visualitza la pàgina.

```
<div class="createQuestionnaireTextArea1-content">

  <mat-form-field>
    <textarea [(ngModel)]="question" matInput placeholder="{{ 'QUESTIONNAIRE.QUESTION' |
translate }}"></textarea>
  </mat-form-field>

  <mat-form-field>
    <mat-select [(value)]="selected" placeholder="{{ 'QUESTIONNAIRE.TEXT2' | translate
}}">
      <mat-option value="textArea">{{ 'QUESTIONNAIRE.ANSWER_OPEN' | translate }}</mat-
option>
      <mat-option value="image">{{ 'QUESTIONNAIRE.QUESTION_IMAGE' | translate }}</mat-
option>
    </mat-select>
    <mat-hint align="end">{{ 'QUESTIONNAIRE.TEXT1' | translate }}</mat-hint>
  </mat-form-field>

  <mat-form-field>
    <textarea [(ngModel)]="urlImage" matInput placeholder="{{ 'QUESTIONNAIRE.URL' |
translate }}"></textarea>
  </mat-form-field>

  <button mat-raised-button (click)="createQuestionnaire()">{{ 'QUESTIONNAIRE.NEXT' |
translate }}</button>
</div>
```

Segona pàgina (createQuestinnaireTextArea.html)

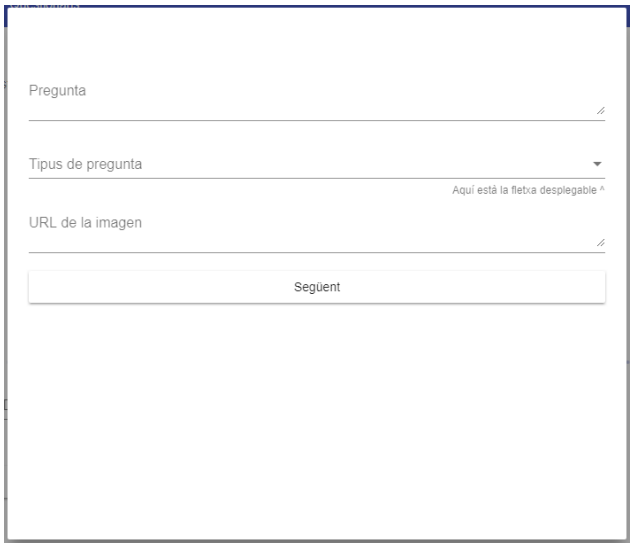


Figura 5.4 Segona pàgina per crear qüestionari de resposta oberta

En el cas de què l'elecció hagi estat la de resposta múltiple, en les següents pàgines s'ha d'introduir les preguntes que formaran el qüestionari. En aquestes s'ha d'introduir la pregunta, les quatre possibles respostes i la resposta correcta.

A continuació es mostra part del codi de la segona pàgina i en la figura 5.5 es visualitza la pàgina.

```
<div class="createQuestionnaireTest1-content">
  <mat-form-field>
    <textarea [(ngModel)]="question" matInput placeholder="{{ 'QUESTIONNAIRE.QUESTION' | translate }}"></textarea>
  </mat-form-field>

  <mat-form-field>
    <textarea [(ngModel)]="answer1" matInput placeholder="{{ 'QUESTIONNAIRE.ANSWER1' | translate }}"></textarea>
  </mat-form-field>

  <mat-form-field>
    <textarea [(ngModel)]="answer2" matInput placeholder="{{ 'QUESTIONNAIRE.ANSWER2' | translate }}"></textarea>
  </mat-form-field>

  <mat-form-field>
    <textarea [(ngModel)]="answer3" matInput placeholder="{{ 'QUESTIONNAIRE.ANSWER3' | translate }}"></textarea>
  </mat-form-field>
  <mat-form-field>
    <textarea [(ngModel)]="answer4" matInput placeholder="{{ 'QUESTIONNAIRE.ANSWER4' | translate }}"></textarea>
  </mat-form-field>
  <mat-form-field>
    <textarea [(ngModel)]="correctAnswer" matInput placeholder="{{ 'QUESTIONNAIRE.CORRECTANSWER' | translate }}"></textarea>
  </mat-form-field>
  <button mat-raised-button (click)="createQuestionnaire()">{{ 'QUESTIONNAIRE.NEXT' | translate }}</button>
</div>
```

Segona pàgina (createQuestinnaireTest.html)

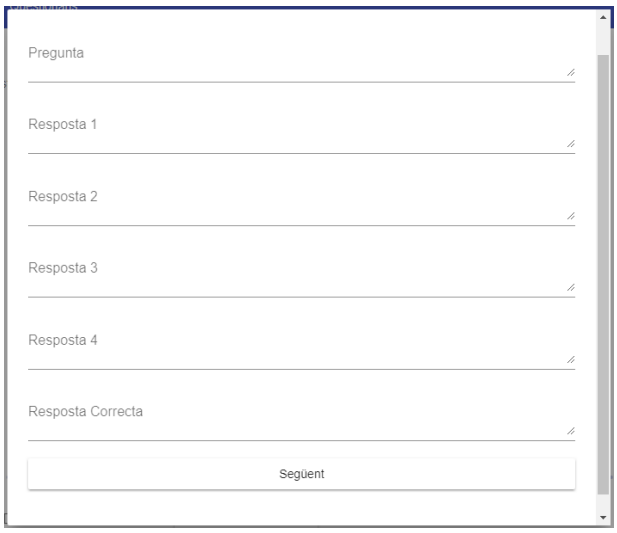


Figura 5.5 Segona pàgina per crear qüestionari de resposta múltiple

5.3.2. Eliminar qüestionari

En la secció dels Qüestionaris també hi ha el botó que permet eliminar un qüestionari existent. Per a realitzar-ho només fa falta introduir el número d'identificador del qüestionari en concret, aquest número apareix en la pàgina principal i sota el nom d'aquest. Una vegada introduït l'identificador i clicat el botó per a eliminar-lo, apareix un avís per a què es confirmi l'eliminació d'aquest.

A continuació es mostra part del codi de la pàgina de l'avís i en la figura 5.6 es visualitza la pàgina.

```
<div class="deleteQuestionnaire-content">
<h3 mat-subheader>{{ 'QUESTIONNAIRE.DELETE1' | translate }} {{data.name}}?</h3>
<button mat-raised-button (click)="deleteQuestionnaire()">{{ 'QUESTIONNAIRE.DELETE2' |
translate }}</button>
<button mat-raised-button (click)="cancel()">{{ 'QUESTIONNAIRE.CANCEL' | translate
}}</button>
</div>
```

Pàgina avís (deleteQuestionnaire.html)

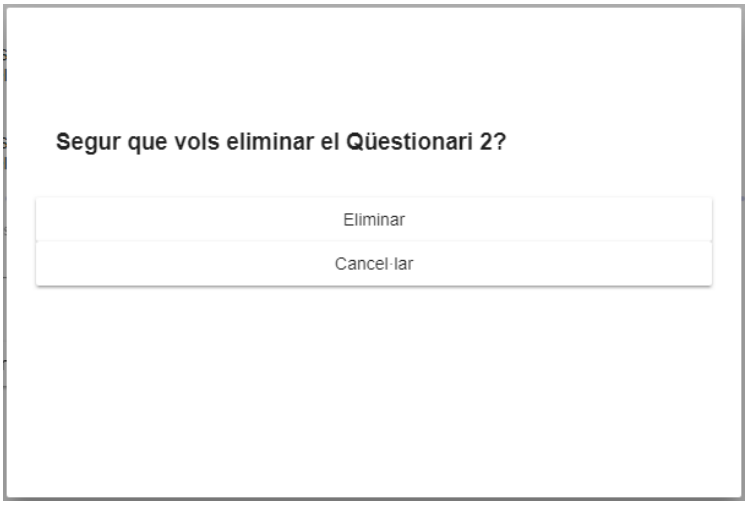


Figura 5.6 Pàgina avís de confirmació

5.3.3. Visualitzar qüestionari

En la secció dels Qüestionaris apareix el llistat dels qüestionaris existents i que han estat creats pel professor que ha iniciat la sessió. Si es clica sobre qualsevol d'ells es visualitzar el qüestionari en concret, en el cas de què s'hagi seleccionat un qüestionari amb preguntes multiresposta es mostra les preguntes, les possibles respostes i la resposta correcta. Si per contra s'ha seleccionat el qüestionari amb preguntes obertes i amb imatges, es mostra les preguntes i les imatges que corresponguin a la pregunta.

A continuació es mostra part del codi de la pàgina del qüestionari seleccionat i en la figura 5.7 es visualitza la pàgina.

```
<div class="questionnaire-content">

  <mat-nav-list *ngIf="myQuestionnaire">
    <h3 mat-header>{{myQuestionnaire.name}}</h3>
  </mat-nav-list>

  <mat-list-item *ngFor="let myQuestion of myQuestions">
    <mat-list-item>
      <h4>{{myQuestion.name}}</h4>
    </mat-list-item>

    <img mat-card-image src={{myQuestion.image}}>
    <mat-list-item *ngFor="let myAnswer of myQuestion.answer">
      <h5>{{myAnswer.name}}</h5>
    </mat-list-item>

    <mat-list-item *ngFor="let myCorrectAnswer of myQuestion.correctAnswer">
      <h5 id="h5">{{myCorrectAnswer.name}}</h5>
    </mat-list-item>
  </mat-list-item>

    <button mat-raised-button(click)="goToResultQuestionnaire()"</button>

</div>
```

Pàgina Qüestionari (questionnaire.html)

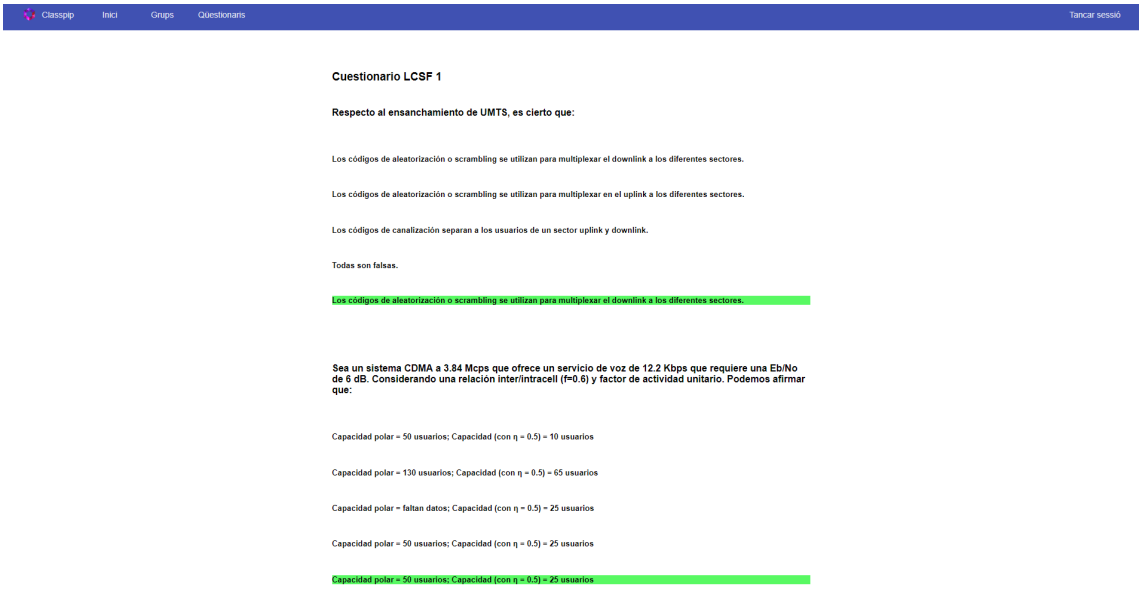


Figura 5.7 Pàgina Qüestionari

5.3.4. Visualitzar respostes i resultats del qüestionari

En el subapartat anterior s'ha explicat el procediment per a poder visualitzar els qüestionaris, una vegada es visualitzen, en la mateixa pàgina hi ha l'opció de poder consultar les respostes i els resultats del qüestionari que es mostra.

En aquesta pàgina es mostren les següents dades: l'identificador de l'estudiant; el nom i cognom de l'estudiant que ha realitzat aquest qüestionari; el número de preguntes encertades; el número de preguntes incorrectes; el resultat; el número total de preguntes que té el qüestionari; les respostes de l'estudiant.

A continuació es mostra part del codi de la pàgina dels resultats del qüestionari seleccionat i en la figura 5.8 es visualitza la pàgina.

```
<div class="questionnaireResults-content">
  <table>

<tr *ngFor="let resultQuestionnaire of resultsQuestionnaire">

        <td>{{resultQuestionnaire.studentId}}</td>
        <td>{{resultQuestionnaire.student._name}}
{{resultQuestionnaire.student._surname}}</td>
        <td>{{resultQuestionnaire.numAnswerCorrect}}</td>
        <td>{{resultQuestionnaire.numAnswerNoCorrect}}</td>
        <td>{{resultQuestionnaire.finalNote}}</td>
        <td>{{resultQuestionnaire.numTotalQuestions}}</td>
        <td>{{resultQuestionnaire.dataAnswers}}</td>

      </tr>

  </table>
</div>
```

Pàgina resultats Qüestionari (questionnaireResults.html)

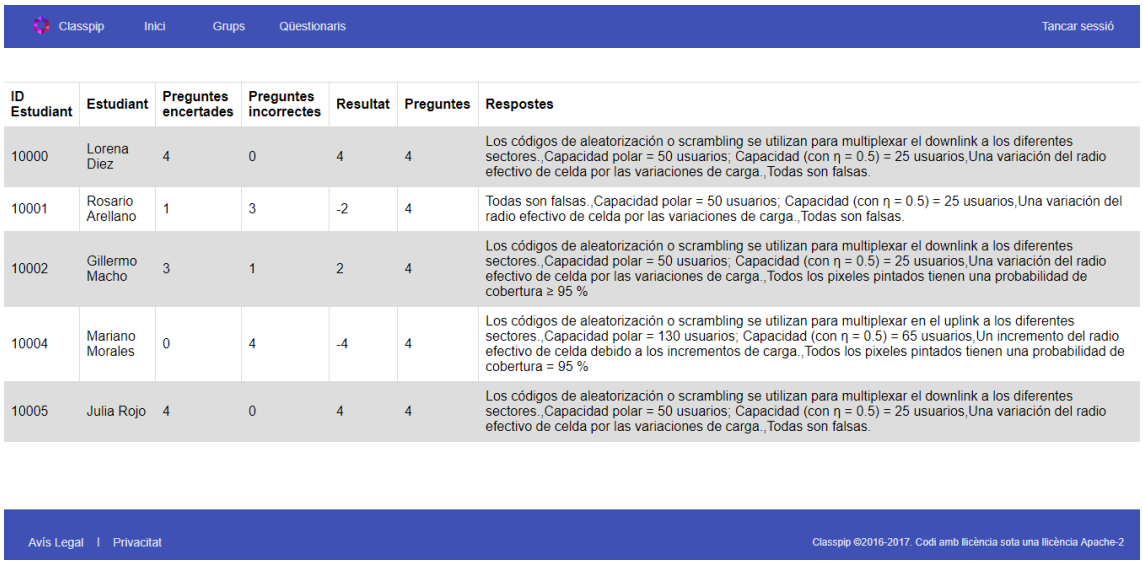


Figura 5.8 Pàgina resultats Qüestionari

En l'annex 7.7 es poden veure la resta de les classes del **Panell d'Administració** d'aquest projecte.

Per a l'autocorrecció de les preguntes amb multiresposta, actualment s'ha pres la decisió de què per a cada resposta incorrecta es resta una de correcta. Per a poder modificar aquesta correcció s'ha d'anar a l'arquitectura de l'aplicació mòbil en la pàgina dels resultats del qüestionari (resultQuestionnaire.ts) i modificar el codi de la funció ***getCorrectionQuestionnaire()*** per a què cada pregunta incorrecta resti el valor que es decideixi.

A continuació es mostra part del codi de la pàgina del resultats del qüestionari amb la funció que realitza l'autocorrecció.

```
export class ResultQuestionnairePage {

  public myResults: ResultQuestionnaire;
  public student: Student;
  public result: ResultQuestionnaire;
  public myQuestions: Array<Question>;
  public myQuestionnaire: Questionnaire;
  public myQuestionsCorrectAnswers: Array<Question>;
  public myCredentials: Credentials;
  public myAnswers: Array<string>;
  public dataAnswers: Array<string>;
  public numTotalQuestions: number = 0;
  public numAnswerCorrect: number = 0;
  public numAnswerNoCorrect: number = 0;
  public finalNote: number = 0;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {
    this.myQuestionsCorrectAnswers = this.navParams.data.myQuestionsCorrectAnswers;
    this.student = this.navParams.data.student;
    this.myQuestions = this.navParams.data.myQuestions;
    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.myCredentials = this.navParams.data.myCredentials;
    this.numTotalQuestions = this.navParams.data.numTotalQuestions;
    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
    this.finalNote = this.navParams.data.finalNote;
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.getCorrectionQuestionnaire();
  }

/**
  * Method to correct the results of the questionnaire
  */
  public getCorrectionQuestionnaire(): void {
    for(var i = 0; i < this.numTotalQuestions; i++){
      if(this.dataAnswers[i]===this.myQuestionsCorrectAnswers[i].correctAnswer[0].name){
        this.numAnswerCorrect += 1;
      }
      else{
        this.numAnswerNoCorrect += 1;
      }
    }
    this.finalNote = this.numAnswerCorrect - this.numAnswerNoCorrect;
      this.questionnaireService.saveResults(this.student,this.myQuestionnaire,
this.myQuestionnaire.name,this.myQuestionnaire.id,this.numTotalQuestions,
this.numAnswerCorrect,this.numAnswerNoCorrect,this.finalNote,this.dataAnswers).subscrib(
      ((value: ResultQuestionnaire) => this.result = value),
      error =>
        this.ionicService.showAlert(this.translateService.instant('APP.ERROR'), error));
  }}
```

Classe resultat qüestionari (resultQuestionnaire.ts)

5.4. Captures de pantalla

En aquest subcapítol es mosta alguna imatge de la secció Qüestionaris del **Panell d'Administrador**. En l'annex 7.8 es mostra el flux complert d'imatges del Panell d'aquest projecte.

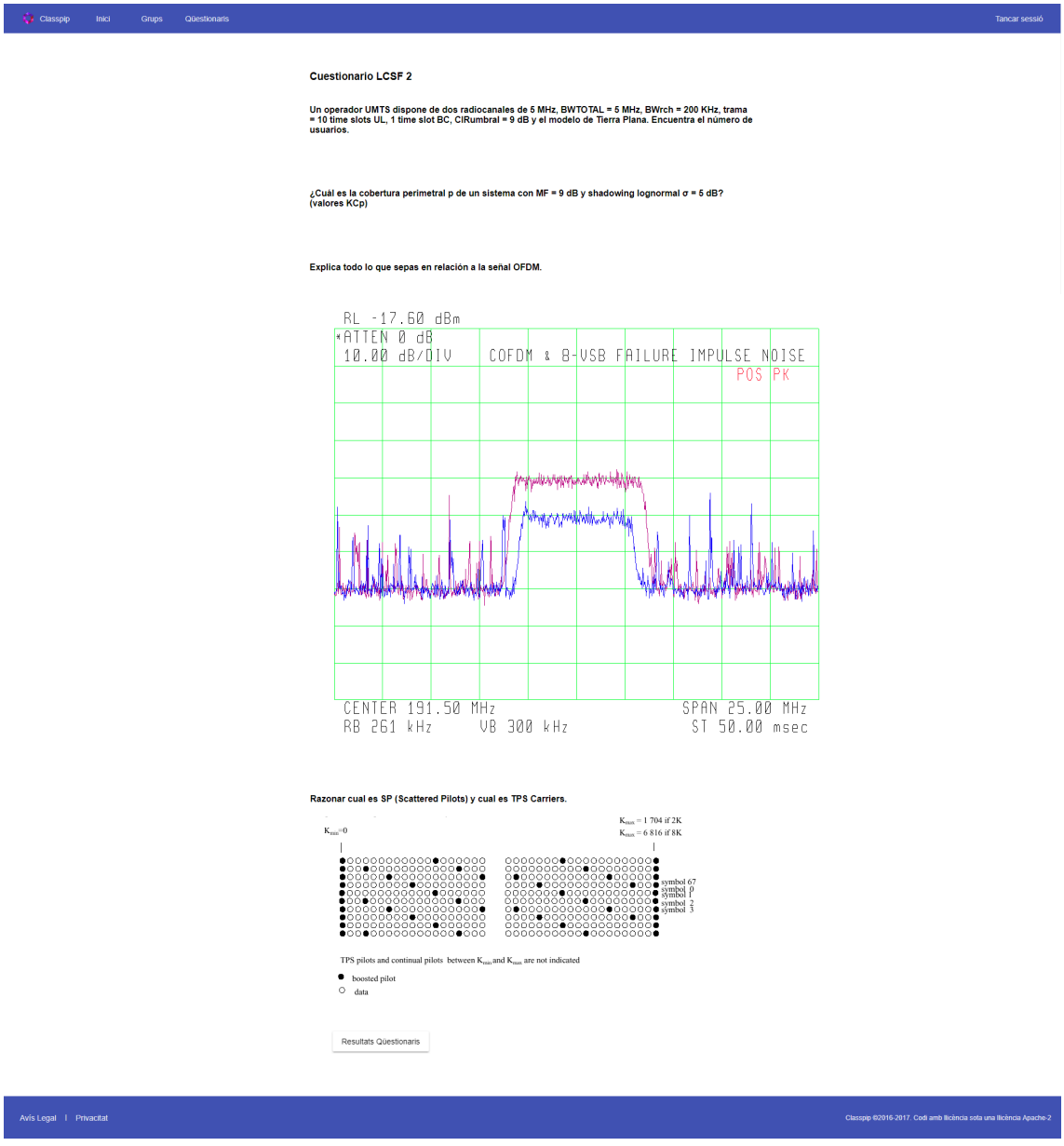


Figura 5.9 Pàgina resultats Qüestionari

CAPÍTOL 6. ARQUITECTURA ORIENTADA SERVEIS

En l'anterior capítol s'ha mostrat el Panell d'Administració que s'ha desenvolupat i alguns exemples de com usar-lo. Aquest capítol tracta de l'**Arquitectura Orientada a Serveis** que gestiona les interaccions entre l'usuari i el **Back-End** [33]. Aquesta arquitectura és l'encarregada de proveir la informació als usuaris que utilitzen les plataformes web i mòbil dissenyades en aquest projecte.

6.1. Introducció

L'objectiu principal d'aquest projecte és la creació d'una **Aplicació Mòbil** per als estudiants i un **Panell d'Administració** per als professors, en què, tots ells comparteixen informació. La gestió d'aquestes dades es realitza mitjançant una **Interfície** de **Programació** d'**Aplicacions** (**API**), en què, tots els usuaris hi estan connectats i la qual s'explica al llarg d'aquest capítol.

6.2. Arquitectura de serveis

Per a poder compartir les dades entre els usuaris, primer s'han de crear els **Models** i **Relacions**[17] que permeten la creació d'una **Arquitectura de Serveis** mitjançant un administrador de l'API, conegut com **Loopback** [34] i que aquest ja està en funcionament al Classpip.

6.3. Interfície de Programació d'Aplicacions

En aquest subapartat s'explica com s'ha configurat l'arquitectura d'aquest projecte per a poder compartir la informació entre els usuaris.

6.3.1. Models i Relacions

En aquest projecte s'han creat diferents models (**questionnaire**, **question**, **answer**, **correctAnswer** i **resultQuestionnaire**) que representen la informació de la **Base de Dades** [35] del **Back-End**. Cadascun d'aquests models contenen part de la informació que configura el conjunt de l'Arquitectura de Serveis. El contingut dels models es detalla a continuació.

- **Questionnaire**: identificador, nom i propietari.

[17] Representació d'entitats d'un sistema així com les seves interrelacions i propietats d'aquest.

- **Questions**: identificador, nom, tipus i temporitzador.
- **Answer**: identificador i nom.
- **CorrectAnswer**: identificador i nom.
- **ResultQuestionnaire**: nom qüestionari, número total preguntes, número respostes correctes, número respostes incorrectes, nota final qüestionari i les respostes dels estudiants.

Aquests models definits s'interpreten com Objectes de **JavaScript**, en què, **Loopback** en genera un conjunt d'operacions per a proporcionar una **REST API** [36]. A cadascun dels models creats es genera un conjunt de relacions que permet relacionar els models entre si.

A continuació es mostra el model **Questionnaire** del projecte i que està definit com un **objecte JSON** [37]. Aquest objecte especifica el **nom**, **base**, **id**, **propietats**, **relacions** i **acls**.

```
"name": "Questionnaire",
"plural": "questionnaires",
"base": "PersistedModel",
"idInjection": true,
"options": {
"validateUpsert": true
},
"properties": {
  "name": { "type": "string", "required": true },
  "date": { "type": "string", "required": true }
},
"validations": [],
"relations": {
  "teacher": { "type": "belongsTo", "model": "Teacher", "foreignKey": "teacherId" },
  "student": { "type": "belongsTo", "model": "Student", "foreignKey": "studentId" },
  "questions": { "type": "hasAndBelongsToMany", "model": "Question", "foreignKey": "questionId" }
},
"acls": [
  {"accessType":"*", "principalType":"ROLE", "principalId":"$everyone", "permission":"DENY" },
  {"accessType":"READ", "principalType":"ROLE", "principalId":"$everyone", "permission":"ALLOW" },
  {"accessType":"WRITE", "principalType":"ROLE", "principalId":"$everyone","permission":"ALLOW" }
],
"methods": {}
```

Definició qüestionari (questionnaire.json)

Les relacions que s'han especificat en el model JSON anterior són les següents:

- **BelongsTo**: un qüestionari pertany a un professor.
- **BelongsTo**: un qüestionari pertany a un estudiant.
- **HasAndBelongsToMany**: un qüestionari té moltes preguntes i les preguntes pertanyen a un qüestionari.

Una vegada creats els **models** i les **relacions** es genera per a cada model un fitxer **JavaScript**. Aquest fitxer conté un conjunt d'operacions que permeten la creació del qüestionari i l'assignació de les preguntes que conté el qüestionari per a poder realitzar les proves en local.

A continuació es mostra l'exemple.

```
app.models.Questionnaire.create([{
  id: 1,
  name: 'Cuestionario LCSF 1',
  date: '19/11/2017',
  teacherId: 1000
 }, {
  id: 2,
  name: 'Cuestionario LCSF 2',
  date: '19/11/2017',
  teacherId: 1000
}], function (err, questionnaires) {

// Assign questions to questionnaires
questionnaires[0].questions.add(questions[0], function (err) {
  if (err) throw err;
   questionnaires[0].questions.add(questions[1], function (err) {
    if (err) throw err;
     questionnaires[0].questions.add(questions[2], function (err) {
      if (err) throw err;
       questionnaires[0].questions.add(questions[3], function (err) {
        if (err) throw err;
         questionnaires[1].questions.add(questions[4], function (err) {
          if (err) throw err;
           questionnaires[1].questions.add(questions[5], function (err) {
            if (err) throw err;
             questionnaires[1].questions.add(questions[6], function (err) {
              if (err) throw err;
               questionnaires[1].questions.add(questions[7], function (err) {
                if (err) throw err;
                 process.nextTick(cb);
```

Creació qüestionari i l'assignació preguntes (create-questions.js)

En el codi anterior s'observa la creació de dos qüestionaris en format JavaScript, aquests estan definits amb el corresponent id, nom i amb l'identificador del professor que l'ha generat. Després d'haver creat els qüestionaris s'assignen les preguntes corresponents a aquests.

En l'annex 7.9 es poden veure els Models i Relacions d'aquest projecte.

6.3.2. Explorador de la Interfície de Programació d'Aplicacions

Després d'haver creat els **Models** i les **Relacions**, si s'executa la comanda ***npm run start*** s'inicia un servidor Web local a la direcció *http: // localhost: 3000 / explorer /* que conté els serveis locals publicats.

Aquesta Web representa una **interfície swagger[18]** amb totes les consultes que es poden realitzar a la Base de Dades des de qualsevol aplicació i que permeten compartit la informació entre els usuaris. D'altra banda, aquest explorador permet executar els mètodes creats en cada model, d'aquesta manera es pot provar l'API amb un client integrat.

[18] Interfície d'un servei web REST.

En la següent figura 6.1 es mostra l'explorador d'API de compilació.

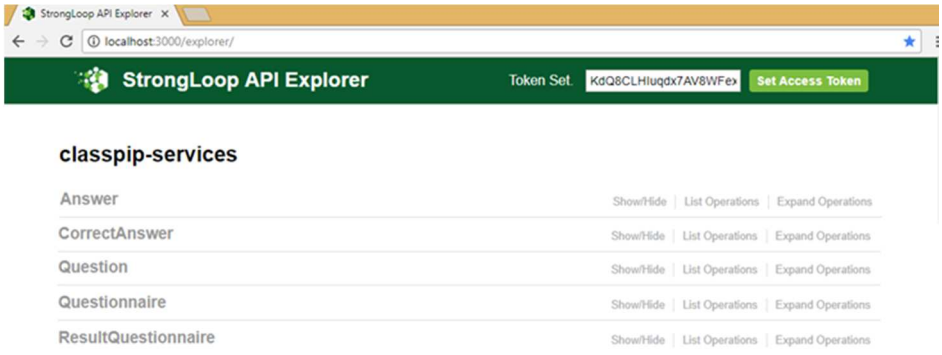


Figura 6.1 Interfície swagger

En la següent figura 6.2 es mostra el model **Questionnaire** del projecte amb alguns dels seus mètodes que s'han generat en l'explorador d'API de compilació.

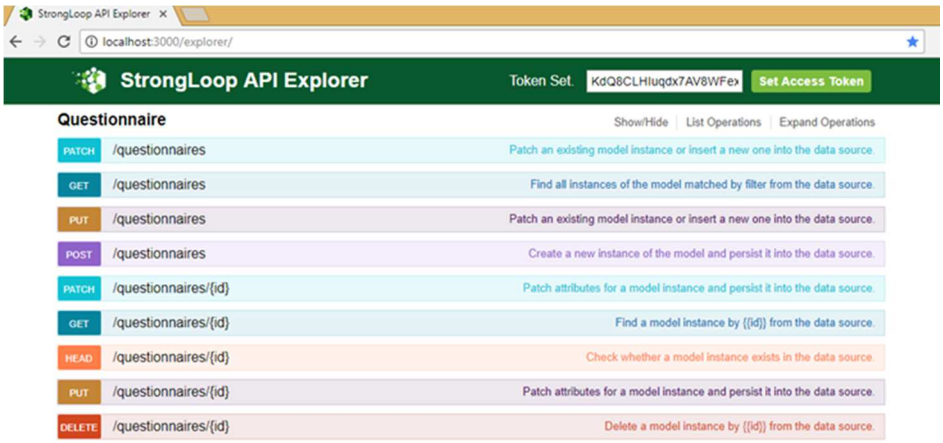


Figura 6.2 Interfície swagger del Qüestionari

En l'annex 7.10 es poden veure tots els serveis locals de l'explorador d'API de compilació.

Seguidament en la figura 6.3 s'executa el mètode **GET /questionnaires** amb el client integrat, d'aquesta manera es realitza la prova que permet consultar tots els qüestionaris que hi ha a la Base de Dades.

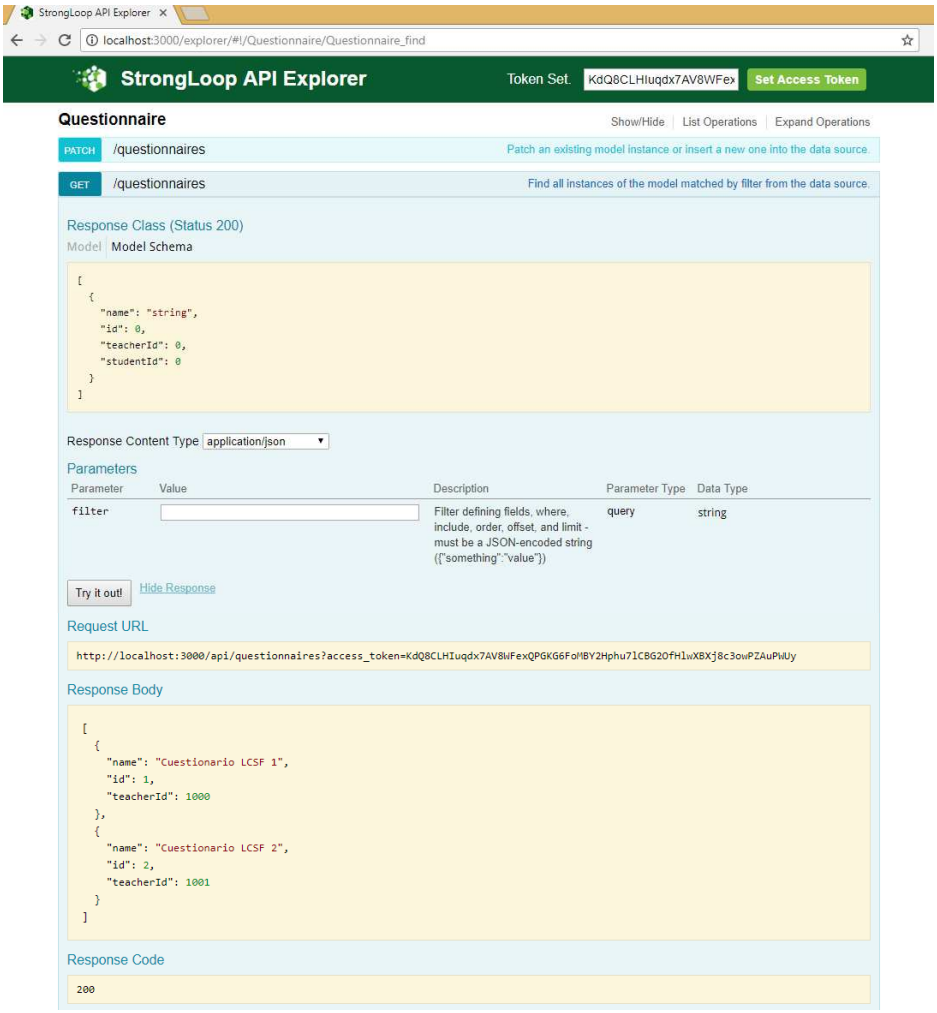


Figura 6.3 Interfície swagger del GET/Qüestionari

6.4. Seguretat

En el subapartat de Models i Relacions s'ha mostrat el model **Questionnaire** del projecte, l'última part de l'objecte JSON s'han definit una sèrie de permisos (**ACL**) [38] que es mostren a continuació.

```
"acls": [
  {"accessType":"*", "principalType":"ROLE", "principalId":"$everyone", "permission":"DENY" },
  {"accessType":"READ", "principalType":"ROLE", "principalId":"$everyone", "permission":"ALLOW" },
  {"accessType":"WRITE", "principalType":"ROLE", "principalId":"$everyone","permission":"ALLOW" }
],
```

Definició ACL (questionnaire.json)

La configuració anterior especifica els següents permisos:

- Es neguen (DENY) tots els mètodes a tothom.
- Es permet (ALLOW) la lectura del qüestionari a tothom.
- Es permet (ALLOW) l'escriptura del qüestionari només al propietari.

En l'annex 7.8 de Models i Relacions es mostren la resta dels permisos dels models d'aquest projecte.

6.5. Model de la Base de Dades

Per a mostrar tota l'arquitectura de la Base de Dades s'ha dissenyat un **diagrama UML** [39] on s'hi representa tota la informació, és a dir, es mostres totes les **taules** i les **relacions** que formen la **Base de Dades** d'aquest projecte.

La següent figura 6.4 en mostra el seu contingut.

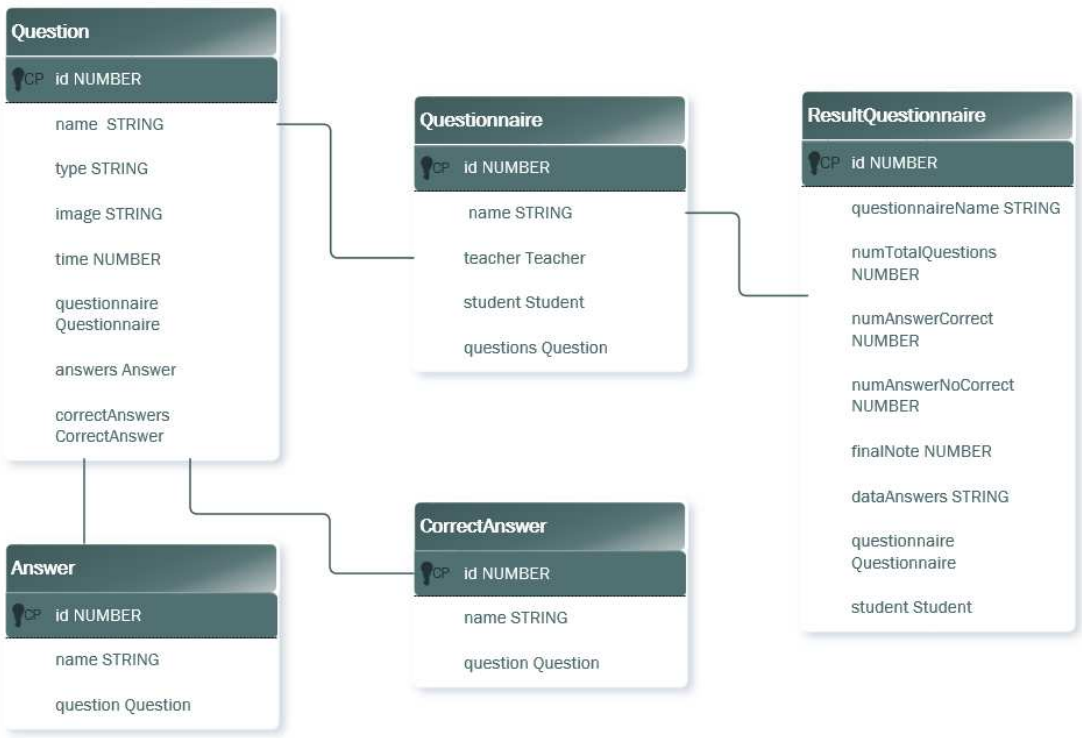


Figura 6.4 Model de l'arquitectura del projecte

En el diagrama anterior s'aprecien relacions entre les taules, la combinació d'aquestes permet el correcte funcionament de l'arquitectura dissenyada. Aquestes relacions s'expliquen a continuació.

Questionnaire:
- **belongsTo**: teacher (un qüestionari pertany a un professor).
- **belongsTo**: student (un qüestionari pertany a un estudiant).
- **hasAndBelongsToMany**: qüestions (un qüestionari té moltes preguntes i les preguntes pertanyen a un qüestionari).

Question:
- **belongsTo**: questionnaire (una pregunta pertany a un qüestionari).
- **hasAndBelongsToMany**: answers (una pregunta té moltes respostes i les respostes pertanyen a la pregunta).
- **hasAndBelongsToMany**: correctAnswers (una pregunta té un resposta correcta i la resposta correcta pertanyen a la pregunta).

Answer:
- **belongsTo**: question (una resposta pertany a una pregunta).

CorrectAnswer:
- **belongsTo**: question (una resposta correcta pertany a una pregunta).

ResultQuestionnaire:
- **belongsTo**: questionnaire (el resultat del qüestionari pertany a un qüestionari).
- **belongsTo**: Student (el resultat del qüestionari pertany a un estudiant).

Tal i com s'ha vist, el model qüestionari està relacionat amb els models professor i estudiant. Aquests models estan implementats en l'arquitectura Classpip ja dissenyada. El mateix passa al model del resultat del qüestionari que està relacionat amb el model estudiant.

CONCLUSIONS I TREBALL FUTUR

El present treball s'ha dedicat al disseny d'una nova **eina de gamificació escolar** inspirada en altres eines ja existents com poden ser **Socrative** i **Kahoot**, aquesta eina s'ha realitzat per a què l'estudiant que l'utilitzi l'ajudi i el motivi a estudiar la matèria que conté les assignatures que estigui cursant.

En el desenvolupament del disseny de l'arquitectura de software que ha donat lloc el present treball s'han assolit els **objectius específics** inicialment plantejats en quan a:

- Dos tipus de qüestionaris per a realitzar segons el tipus de pregunta: preguntes tipus test i **multiresposta**; preguntes que contenen una **imatge**; preguntes amb **resposta oberta** per a redactar.

- Desafiament alhora de realitzar les preguntes, en què, se li aplica un **repte** a cada pregunta del qüestionari. S'ha introduït un temporitzador a cadascuna de les preguntes.

- **Correcció automàtica** de les preguntes que contenen multiresposta. Finalitza't el qüestionari els estudiants poden saber les respostes correctes i el resultat que han obtingut.

Una vegada desenvolupat el disseny de la plataforma s'ha realitzat les proves de validació necessàries per a la comprovació del correcte funcionament de l'**Aplicació Mòbil** i del **Panell d'Administració**.

Per tant, la primera conclusió que es pot treure d'aquest treball és que la planificació inicial s'ha completat satisfactòriament, el disseny de la nova plataforma de software s'ha desenvolupat i finalitzat tal i com s'esperava des d'un bon principi. Aquest va ser l'objectiu principal i darrerament es pot dir que s'ha complert.

Al llarg del present treball s'ha utilitzat diferents tecnologies per al desenvolupament del disseny de l'eina de gamificació escolar. Aquestes tecnologies han estat **Ionic 2**, **Angular 2**, **HTML5** i **Angular Material**, totes elles han permès construir les plataformes per l'**Aplicació Mòbil** i pel **Panell d'Administració**.

Com a conclusió tècnica es pot recomanar positivament l'ús de les noves tecnologies utilitzades com **Ionic 2** i **Angular 2**. En què, **Ionic 2** ha fet possible crear aplicacions increïbles en una sola base de codi per a qualsevol plataforma, amb la web. Mentre que **Angular 2** ha permès el desenvolupament d'una aplicació Web **Front-End** i és un dels frameworks més populars per a desenvolupar aplicacions modernes i escalables al costat del client.

Un altre aspecte satisfactori i important a recomanar és la transformació que s'ha realitzat a la plataforma nativa del **Panell d'Administració** en el present treball.

L'antic Panell estava dissenyat amb el framework **Bootstrap**, que té un joc d'eines de codi obert per al desenvolupament amb **HTML**, **CSS** i **JS**. Actualment, el **Panell d'Administració** utilitza el framework **Angular Material** que té components de disseny de materials per a **Angular 2** que permeten construir una interfície d'usuari moderna i versàtil.

Com a conclusió personal és important explicar la importància i el repte que ha estat per a mi el desenvolupament del disseny de l'arquitectura de software del present treball, ja que abans d'iniciar aquest treball no tenia els coneixements necessaris per a poder utilitzar les noves tecnologies **Ionic 2** i **Angular 2** per a la creació d'aquesta plataforma. Ha estat un previ i constant aprenentatge de les eines utilitzades. No obstant, és per a mi una gran satisfacció el desafiament que he tingut durant el treball i l'assoliment dels coneixements obtinguts sobre l'ús d'aquestes tecnologies.

D'altra banda, una altra conclusió personal és sobre la importància que té per a mi haver assolit els coneixements de l'ús d'aquestes noves tecnologies i haver-los posat en pràctica amb la realització del present treball. Crec que aquests coneixements m'obriran les portes en el mon laboral d'avui en dia perquè la tecnologia avança diàriament igual que les empreses es reciclen paral·lelament amb la tecnologia.

A més a més de les conclusions i del present treball es vol anomenar des del punt de vista com a desenvolupador, les possibles tasques a realitzar en un futur treball per ampliar la present plataforma software dissenyada, amb la finalitat de tenir una eina de gamificació complerta per a la **motivació d'estudi**.

Des del punt de vista tecnològic seria favorable que els qüestionaris existents en la base de dades es pogués editar les preguntes, les possibles respostes i la resposta correcta d'aquest. Actualment els qüestionaris només es poden crear i eliminar per complert.

Un altre punt tecnològic relacionat amb la correcció dels qüestionaris que realitzen els estudiants és la possible correcció automàtica de les preguntes amb resposta oberta, ja que actualment només es realitza l'autocorrecció a les preguntes amb multiresposta.

L'últim punt tecnològic a comentar és la inclusió de l'opció de poder descarregar les respostes i els resultats dels qüestionaris realitzats en format PDF. D'aquesta manera els professors poden tenir-los emmagatzemats en local.

BIBLIOGRAFIA

[1] Universitat Politècnica de Catalunya [online] <https://www.upc.edu/ca> [Consulta: 1 Ag. 2017]
[2] Socrative [online] <https://www.socrative.com/> [Consulta: 3 Ag. 2017]
[3] Kahoot [online] <https://kahoot.com/> [Consulta: 5 Ag. 2017]
[4] Wikipedia. Arquitectura de software [online] <https://es.wikipedia.org/wiki/Arquitectura_de_software> [Consulta: 8 Ag. 2017]
[5] Wikipedia. Interfaz de programación de aplicaciones [online] <https://en.wikipedia.org/wiki/Application_programming_interface> [Consulta: 9 Ag. 2017]
[6] StrongLoop [online] <https://strongloop.com/> [Consulta: 14 Ag. 2017]
[7] Express [online] <http://expressjs.com/es/> [Consulta: 18 Ag. 2017]
[8] Node.js [online] <https://nodejs.org/es/> [Consulta: 22 Ag. 2017]
[9] Wikipedia. MySQL [online] <https://es.wikipedia.org/wiki/MySQL> [Consulta: 23 Ag. 2017]
[10] Ionic 2 [online] <http://ionic.io/2> [Consulta: 2 Set. 2017]
[11] Angular 2 [online] <http://www.angular2.com/> [Consulta: 5 Set. 2017]
[12] Apache Cordova [online] <https://cordova.apache.org/> [Consulta: 7 Set. 2017]
[13] Bootstrap [online] <https://getbootstrap.com/> [Consulta: 10 Set. 2017]
[14] Angular Material [online] <https://material.angular.io/> [Consulta: 12 Set. 2017]
[15]
[16] Git Hub [online] <https://github.com/> [Consulta: 16 Set. 2017]
[17] Coveralls [online] <https://coveralls.io/> [Consulta: 19 Set. 2017]
[18] Codacy [online] <https://www.codacy.com/> [Consulta: 20 Set. 2017]
[19] Hockey App [online] <https://hockeyapp.net/> [Consulta: 23 Set. 2017]
[20] Wikipedia. GanttProject [online] <https://es.wikipedia.org/wiki/GanttProject> [Consulta: 4 Oct. 2017]
[21] GanttProject [online] <https://www.ganttproject.biz/> [Consulta: 6 Oct. 2017]
[22] Travis CI [online] <https://travis-ci.org/> [Consulta: 9 Oct. 2017]
[23] Karma.js [online] <https://karma-runner.github.io/1.0/index.html> [Consulta: 9 Oct. 2017]
[24] Wikipedia. SHA-2 [online] <https://es.wikipedia.org/wiki/SHA-2> [Consulta: 11 Oct. 2017]
[25] Wikipedia. Transport Layer Security [online] <https://es.wikipedia.org/wiki/Transport_Layer_Security> [Consulta: 17 Oct. 2017]
[26] Wikipedia. Advanced Encryption Standard [online] <https://es.wikipedia.org/wiki/Advanced_Encryption_Standard> [Consulta: 20 Oct. 2017]
[27] Wikipedia. Framework [online] <https://es.wikipedia.org/wiki/Framework> [Consulta: 24 Oct. 2017]
[28] Wikipedia. Model-Vista-Controlador [online] <https://ca.wikipedia.org/wiki/Model-Vista-Controlador> [Consulta: 30 Oct. 2017]

[29] Ionic docs [online] <https://ionicframework.com/docs/> [Consulta: 1 Nov. 2017]
[30] Wikipedia. HTML [online] <https://es.wikipedia.org/wiki/HTML> [Consulta: 5 Nov. 2017]
[31] Angular [online] <https://angular.io/> [Consulta: 7 Nov. 2017]
[32] Wikipedia. TypeScript [online] <https://es.wikipedia.org/wiki/TypeScript> [Consulta: 10 Nov. 2017]
[33] Wikipedia. Front-end y back-end [online] <https://es.wikipedia.org/wiki/Front-end_y_back-end> [Consulta: 13 Nov. 2017]
[34] Wikipedia. Loopback [online] <https://es.wikipedia.org/wiki/Loopback> [Consulta: 17 Nov. 2017]
[35] Wikipedia. Base de datos [online] <https://es.wikipedia.org/wiki/Base_de_datos> [Consulta: 19 Nov. 2017]
[36] API REST [online] <http://wiki.servicenow.com/index.php?title=REST_API#gsc.tab=0> [Consulta: 22 Nov. 2017]
[37] Wikipedia. JSON [online] <https://es.wikipedia.org/wiki/JSON> [Consulta: 23 Nov. 2017]
[38] Wikipedia. Lista de control de acceso [online] <https://es.wikipedia.org/wiki/Lista_de_control_de_acceso> [Consulta: 27 Nov. 2017]
[39] Wikipedia. Lenguaje unificado de modelado [online] <https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado> [Consulta: 29 Nov. 2017]
[40] Wikipedia. Front-end y back-end [online] <https://es.wikipedia.org/wiki/Front-end_y_back-end> [Consulta: 1 Des. 2017]

CAPÍTOL 7. ANNEXOS

7.1. Timer plataforma mòbil

En aquest subcapítol es mostra el codi del Timer de l'Aplicació Mòbil.

```
  ngOnInit() {
    this.initTimer();
  }

  hasFinished() {
    return this.timer.hasFinished;
  }

  pauseTimer() {
    this.timer.runTimer = false;
  }
  resumeTimer() {
    this.startTimer();
  }

  initTimer() {
    if (!this.timeInSeconds){
      this.timeInSeconds = this.questionsSend.time;
    }

    this.timer = <PTimer>{
      time: this.timeInSeconds,
      runTimer: false,
      hasStarted: false,
      hasFinished: false,
      timeRemaining: this.timeInSeconds
    };
    this.timer.displayTime                                                    =
this.getSecondsAsDigitalClock(this.timer.timeRemaining);
    this.startTimer();
  }

  getSecondsAsDigitalClock(inputSeconds: number) {
    var sec_num = parseInt(inputSeconds.toString(), 10); // don't forget the
second param
    var hours = Math.floor(sec_num / 3600);
    var minutes = Math.floor((sec_num - (hours * 3600)) / 60);
    var seconds = sec_num - (hours * 3600) - (minutes * 60);
    var hoursString = '';
    var minutesString = '';
    var secondsString = '';
    hoursString = (hours < 10) ? "0" + hours : hours.toString();
    minutesString = (minutes < 10) ? "0" + minutes : minutes.toString();
    secondsString = (seconds < 10) ? "0" + seconds : seconds.toString();
    return hoursString + ':' + minutesString + ':' + secondsString;
  }

  startTimer() {
    this.timer.hasStarted = true;
    this.timer.runTimer = true;
    this.timerTick();
  }

  timerTick() {
    setTimeout(() => {

      if (!this.timer.runTimer){
```

```
        return;
      }
      this.timer.timeRemaining--;
      this.timer.displayTime=
this.getSecondsAsDigitalClock(this.timer.timeRemaining);
      if (this.timer.timeRemaining > 0) {
        this.timerTick();
      }
      else {
        this.timer.hasFinished = true;
        if (this.timer.timeRemaining === 0) {
          this.doSubmitEmptyAnswer();
        }
      }
    }, 1000);
}
```

7.2. JSONs platorma mòbil multiidioma

En aquest subcapítol es mostren els JSONs del projecte que fan possible que l'Aplicació Mòbil sigui multiidioma.

7.2.1. Json idioma Català

```
{
  "APP": {
    "ERROR": "ERROR",
    "WAIT": "Si us plau, espera...",
    "OK": "OK"
  },
  "MENU": {
    "TITLE": "Menú",
    "PROFILE": "El meu perfil",
    "QUESTIONNAIRE": "Els meus Qüestionaris",
    "LOGOUT": "Tancar sessió"
  },
  "POPOVER": {
    "PROFILE": "El meu perfil",
    "QUESTIONNAIRE": "Els meus Qüestionaris",
    "LOGOUT": "Tancar sessió"
  },
  "ROLE-SELECT": {
    "TEACHER": "Docent",
    "STUDENT": "Estudiant",
    "SCHOOL-LEADER": "Director de l'Escola"
  },
  "LOGIN": {
    "TITLE": "Classpip",
    "USERNAME": "Nom d'Usuari",
    "PASSWORD": "Contrasenya",
    "LOGIN": "ENTRAR",
    "FORGOT": "He oblidat la contrasenya",
    "REGISTER": "Necessito crear un nou compte"
  },
    "GETQUESTIONNAIRE": {
    "TITLE": "Els meus Qüestionaris",
    "USERNAME": "Nom d'Usuari",
    "CODE": "Codi del Qüestionari",
    "LOGIN": "ACCEDIR"
  },
```

```
    "QUESTIONNAIRES": {
    "TITLE": "Els meus Qüestionaris",
    "RESULTS": "Els meus Resultats"
  },
    "QUESTIONNAIRE": {
    "TITLE": "El meu Qüestionari",
    "USERNAME": "Creat per",
    "CODE": "Codi del Qüestionari",
    "SUBMIT": "Guardar i continuar",
    "RESULT": "Consultar els resultats",
    "FINISH": "Has acabat de realitzar el qüestionari. Consulta la solució i els
teus resultats o tanca el qüestionari.",
    "FINISH1": "Has acabat de realitzar el qüestionari. Els teus resultats es
publicaran en breu.",
    "LOGIN": "ACCEDIR",
    "LOGOUT": "Tancar Qüestionari",
    "TEXT1": "Solució del",
    "TEXT2": "Resposta correcta:",
    "TEXT3": "Resultats obtinguts per",
    "TEXT4": "Les teves respostes:",
    "TEXT5": "Resultats:",
    "TEXT6": "Total preguntes:",
    "TEXT7": "Total respostes acertades:",
    "TEXT8": "Total respostes no acertades:",
    "TEXT9": "Nota final:"
  },
  "HOME": {
    "TITLE": "Inici",
    "TEACHERS": "Professors",
    "STUDENTS": "Estudiants",
    "FACEBOOK": "Facebook",
    "TWITTER": "Twitter",
    "WEBSITE": "Web",
    "LANGUAGE": "Escull l'idioma"
  },
  "SCHOOL": {
    "TITLE": "La meva Escola"
  },
  "PROFILE": {
    "TITLE": "El meu perfil",
    "ACCOUNT": "Compte",
    "NAME": "Nom",
    "USERNAME": "Nom d'Usuari",
    "EMAIL": "Correu Electrónic",
    "CHANGEPASS": "Cambiar Contrasenya",
    "RESOURCES": "Recursos",
    "HELP": "Ajuda",
    "TERMS": "Privacitat i Termes",
    "VERSION": "Versió de la app"
  },
  "TERMS": {
    "TITLE": "Privacitat i Termes"
  },
  "HELP": {
    "TITLE": "Ajuda"
  },
  "TEACHERS": {
    "TITLE": "Professors"
  },
  "TEACHER": {
    "NAME": "Nom",
    "USERNAME": "Nom d'Usuari",
    "EMAIL": "Correu Electrónic"
  },
  "STUDENTS": {
    "TITLE": "Estudiants"
  },
  "STUDENT": {
```

```
    "NAME": "Nom",
    "USERNAME": "Nom d'Usuari",
    "EMAIL": "Correo Electrónic"
  },
  "ERROR": {
    "LOGIN_FAILED": "El nom d'usuari i / o contrasenya invàlides.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "El teu correu no ha estat verificat
encara, si us plau, revisa el teu correu.",
    "INTERNAL_ERROR": "Hi ha hagut un error intern al servidor. Si us plau
contacta amb el teu administrador."
}
```


7.2.2. Json idioma Castellà


```
{
  "APP": {
    "ERROR": "ERROR",
    "WAIT": "Por favor, espera...",
    "OK": "OK"
  },
  "MENU": {
    "TITLE": "Menú",
    "PROFILE": "Mi Perfil",
    "QUESTIONNAIRE": "Mis Cuestionarios",
    "LOGOUT": "Cerrar sessión"
  },
  "POPOVER": {
    "PROFILE": "Mi Perfil",
    "QUESTIONNAIRE": "Mis Cuestionarios",
    "LOGOUT": "Cerrar sessión"
  },
  "ROLE-SELECT": {
    "TEACHER": "Docente",
    "STUDENT": "Estudiante",
    "SCHOOL-LEADER": "Director de Escuela"
  },
  "LOGIN": {
    "TITLE": "Classpip",
    "USERNAME": "Nombre de Usuario",
    "PASSWORD": "Contraseña",
    "LOGIN": "ENTRAR",
    "FORGOT": "Olvidé mi contraseña",
    "REGISTER": "Necesito crear una nueva cuenta"
  },
    "GETQUESTIONNAIRE": {
    "TITLE": "Mis Cuestionarios",
    "USERNAME": "Nombre de Usuario",
    "CODE": "Código del Cuestionario",
    "LOGIN": "ACCEDER"
  },
    "QUESTIONNAIRES": {
    "TITLE": "Mis Cuestionarios",
    "RESULTS": "Mis Resultados"
  },
    "QUESTIONNAIRE": {
    "TITLE": "Mi Cuestionario",
    "USERNAME": "Creado por",
    "CODE": "Código del Cuestionario",
    "SUBMIT": "Guardar y continuar",
    "RESULT": "Consultar resultados",
    "FINISH": "Has terminado de realizar el cuestionario. Consulta la solución
y tus resultados o cierra el cuestionario.",
```

```
    "FINISH1": "Has terminado de realizar el cuestionario. Tus resultados se
publicarán en breve.",
    "LOGIN": "ACCEDER",
    "LOGOUT": "Cerrar Cuestionario",
    "TEXT1": "Solución del",
    "TEXT2": "Respuesta correcta:",
    "TEXT3": "Resultados obtenidos por",
    "TEXT4": "Tus respuestas:",
    "TEXT5": "Resultados:",
    "TEXT6": "Total preguntas:",
    "TEXT7": "Total respuestas acertadas:",
    "TEXT8": "Total respuestas no acertadas:",
    "TEXT9": "Nota final:"
  },
  "HOME": {
    "TITLE": "Inicio",
    "TEACHERS": "Profesores",
    "STUDENTS": "Estudiantes",
    "FACEBOOK": "Facebook",
    "TWITTER": "Twitter",
    "WEBSITE": "Web",
    "LANGUAGE": "Escoge el idioma"
  },
  "SCHOOL": {
    "TITLE": "Mi Escuela"
  },
  "PROFILE": {
    "TITLE": "Mi Perfil",
    "ACCOUNT": "Cuenta",
    "NAME": "Nombre",
    "USERNAME": "Nombre de Usuario",
    "EMAIL": "Correo Electrónico",
    "CHANGEPASS": "Cambiar Contraseña",
    "RESOURCES": "Recursos",
    "HELP": "Ayuda",
    "TERMS": "Privacidad y Términos",
    "VERSION": "Versión de la app"
  },
  "TERMS": {
    "TITLE": "Privacidad y Términos"
  },
  "HELP": {
    "TITLE": "Ayuda"
  },
  "TEACHERS": {
    "TITLE": "Profesores"
  },
  "TEACHER": {
    "NAME": "Nombre",
    "USERNAME": "Nombre de Usuario",
    "EMAIL": "Correo Electrónico"
  },
  "STUDENTS": {
    "TITLE": "Estudiantes"
  },
  "STUDENT": {
    "NAME": "Nombre",
    "USERNAME": "Nombre de Usuario",
    "EMAIL": "Correo Electrónico"
  },
  "ERROR": {
    "LOGIN_FAILED": "El nombre de usuario y/o la contraseña son inválidas.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "Tu correo no ha sido verificado aún, por
favor, revisa tu correo.",
    "INTERNAL_ERROR": "Ha ocurrido un error interno en el servidor. Por favor
contacta con tu administrador. "
  }
}
```

7.2.3. Json idioma Anglès


```
{
  "APP": {
    "ERROR": "ERROR",
    "WAIT": "Please wait ...",
    "OK": "OK"
  },
  "MENU": {
    "TITLE": "Menu",
    "PROFILE": "My profile",
    "QUESTIONNAIRE": "My Questionnaires",
    "LOGOUT": "Logout"
  },
  "POPOVER": {
    "PROFILE": "My profile",
    "QUESTIONNAIRE": "My Questionnaires",
    "LOGOUT": "Logout"
  },
  "ROLE-SELECT": {
    "TEACHER": "Teacher",
    "STUDENT": "Student",
    "SCHOOL-LEADER": "School Principal"
  },
  "LOGIN": {
    "TITLE": "Classpip",
    "USERNAME": "Username",
    "PASSWORD": "Password",
    "LOGIN": "GET IN",
    "FORGOT": "I forgot my password",
    "REGISTER": "I need to create a new account"
  },
    "GETQUESTIONNAIRE": {
    "TITLE": "My Questionnaires",
    "USERNAME": "Username",
    "CODE": "Code of the Questionnaire",
    "LOGIN": "GET IN"
  },
    "QUESTIONNAIRES": {
    "TITLE": "My Questionnaires",
    "RESULTS": "My Results"
  },
    "QUESTIONNAIRE": {
    "TITLE": "My Questionnaire",
    "USERNAME": "Created by",
    "CODE": "Code of Questionnaire",
    "SUBMIT": "Save and continue",
    "RESULT": "Consult results",
    "FINISH": "You have completed the questionnaire. Check the solution and your
results or close the questionnaire.",
    "FINISH1": "You have completed the questionnaire. Your results will be posted
shortly.",
    "LOGIN": "GET IN",
    "LOGOUT": "Close questionnaire",
    "TEXT1": "Solution of",
    "TEXT2": "Correct answer:",
    "TEXT3": "Results obtained by",
    "TEXT4": "Your answers:",
    "TEXT5": "Results:",
    "TEXT6": "Total questions:",
    "TEXT7": "Total answers right:",
    "TEXT8": "Total unanswered answers:",
    "TEXT9": "Final note:"
  },
  "HOME": {
    "TITLE": "Home",
    "TEACHERS": "Teachers",
```

```
    "STUDENTS": "Students",
    "FACEBOOK": "Facebook",
    "TWITTER": "Twitter",
    "WEBSITE": "Web",
    "LANGUAGE": "Choose the language"
  },

  "SCHOOL": {
    "TITLE": "My school"
  },

  "PROFILE": {
    "TITLE": "My profile",
    "ACCOUNT": "Account",
    "NAME": "Name",
    "USERNAME": "Username",
    "EMAIL": "Email",
    "CHANGEPASS": "Change Password",
    "RESOURCES": "Means",
    "HELP": "Help",
    "TERMS": "Privacy and Terms",
    "VERSION": "Version of the app"
  },

  "TERMS": {
    "TITLE": "Privacy and Terms"
  },

  "HELP": {
    "TITLE": "Help"
  },
  "TEACHERS": {
    "TITLE": "Teachers"
  },
  "TEACHER": {
    "NAME": "Name",
    "USERNAME": "Username",
    "EMAIL": "Email"
  },
  "STUDENTS": {
    "TITLE": "Students"
  },
  "STUDENT": {
    "NAME": "Name",
    "USERNAME": "Username",
    "EMAIL": "Email"
  },
  "ERROR": {
    "LOGIN_FAILED": "The username and / or password are invalid.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "Your email has not been verified yet,
please check your email.",
    "INTERNAL_ERROR": "An internal error has occurred on the server. Please
contact your administrator. "
  }
}
```


7.3. Models aplicació mòbil

En aquest subcapítol es mostren els Models de l'arquitectura de l'Aplicació Mòbil del projecte.

7.3.1. Model Qüestionari

```
export class Questionnaire {

  private _id: string;
  private _name: string;

  constructor(id?: string, name?: string) {
      this._id = id;
      this._name = name;
  }

  /* tslint:disable */
  static toObject(object: any): Questionnaire {
    /* tslint:enable */
    let result: Questionnaire = new Questionnaire();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
    }
    return result;
  }

  /* tslint:disable */
  static toObjectArray(object: any): Array<Questionnaire> {
    /* tslint:enable */
    let resultArray: Array<Questionnaire> = new Array<Questionnaire>();
    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(Questionnaire.toObject(object[i]));
      }
    }
    return resultArray;
  }

  public get id(): string {
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
  }

  public get name(): string {
    return this._name;
  }

  public set name(value: string) {
    this._name = value;
  }}
```

7.3.2. Model Question

```
import { Answer} from './answer'
import { CorrectAnswer} from './correctAnswer'

export class Question {

  private _id: string;
  private _name: string;
  private _type: string;
  private _image: string;
```

```
  private _time: number;
  private _answer: Array<Answer>;
  private _correctAnswer: Array<CorrectAnswer>;

  constructor( id?: string, name?: string, type?: string, image?: string,
time?: number ) {
    this._id = id;
    this._name = name;
    this._type = type;
    this._image = image;
    this._time = time;
  }

  /* tslint:disable */
  static toObject(object: any): Question {
    /* tslint:enable */
    let result: Question = new Question();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
      result.type = object.type;
      result.image = object.image;
      result.time = object.time;
    }
    return result;
  }

  /* tslint:disable */
  static toObjectArray(object: any): Array<Question> {
    /* tslint:enable */
    let resultArray: Array<Question> = new Array<Question>();
    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(Question.toObject(object[i]));
      }
    }
    return resultArray;
  }

  public get id(): string {
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
  }

  public get name(): string {
    return this._name;
  }

  public set name(value: string) {
    this._name = value;
  }

  public get type(): string {
    return this._type;
  }

  public set type(value: string) {
    this._type = value;
  }

  public get image(): string {
    return this._image;
  }

  public set image(value: string) {
```

```
    this._image = value;
  }

  public get time(): number {
    return this._time;
  }

  public set time(value: number) {
    this._time = value;
  }

  public get answer(): Array<Answer> {
    return this._answer;
  }

  public set answer(value: Array<Answer>) {
    this._answer = value;
  }

  public get correctAnswer(): Array<CorrectAnswer> {
    return this._correctAnswer;
  }

  public set correctAnswer(value: Array<CorrectAnswer>) {
    this._correctAnswer = value;
  }
}
```

7.3.3. Model Answer

```
export class Answer {

  private _id: string;
  private _name: string;

  constructor( id?: string, name?: string ) {
    this._id = id;
    this._name = name;
  }

  /* tslint:disable */
  static toObject(object: any): Answer {
    /* tslint:enable */
    let result: Answer = new Answer();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
    }
    return result;
  }

  /* tslint:disable */
  static toObjectArray(object: any): Array<Answer> {
    /* tslint:enable */
    let resultArray: Array<Answer> = new Array<Answer>();
    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(Answer.toObject(object[i]));
      }
    }
    return resultArray;
  }

  public get id(): string {
```

```
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
  }

  public get name(): string {
    return this._name;
  }

  public set name(value: string) {
    this._name = value;
  }
}
```

7.3.4. Model Correct Answer

```
export class CorrectAnswer {

  private _id: string;
  private _name: string;

  constructor( id?: string, name?: string ) {
    this._id = id;
    this._name = name;
  }

  /* tslint:disable */
  static toObject(object: any): CorrectAnswer {
    /* tslint:enable */
    let result: CorrectAnswer = new CorrectAnswer();
    if (object != null) {
      result.id = object.id;
      result.name = object.name;
    }
    return result;
  }

  /* tslint:disable */
  static toObjectArray(object: any): Array<CorrectAnswer> {
    /* tslint:enable */
    let resultArray: Array<CorrectAnswer> = new Array<CorrectAnswer>();

    if (object != null) {
      for (let i = 0; i < object.length; i++) {
        resultArray.push(CorrectAnswer.toObject(object[i]));
      }
    }
    return resultArray;
  }

  public get id(): string {
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
  }

  public get name(): string {
    return this._name;
  }
```

```
  public set name(value: string) {
    this._name = value;
  }

}
```

7.3.5. Model Result Questionnaire

```
import { Questionnaire } from './questionnaire';
import { Student } from './student';

export class ResultQuestionnaire {

  private _id: string;
  private _questionnaireName: string;
  private _questionnaireId: number;
  private _questionnaire: Questionnaire;
  private _numTotalQuestions: number;
  private _numAnswerCorrect: number;
  private _numAnswerNoCorrect: number;
  private _finalNote: number;
  private _studentId: number;
  private _student: Student;
  private _dataAnswers: Array<string>;

  constructor(id?: string, questionnaireName?: string, questionnaireId?:
number, numTotalQuestions?: number, numAnswerCorrect?: number,
numAnswerNoCorrect?: number, finalNote?: number, studentId?: number,
dataAnswers?: Array<string>) {
    this._id = id;
    this._questionnaireName = questionnaireName;
    this._questionnaireId = questionnaireId;
    this._numTotalQuestions = numTotalQuestions;
    this._numAnswerCorrect = numAnswerCorrect;
    this._numAnswerNoCorrect = numAnswerNoCorrect;
    this._finalNote = finalNote;
    this._studentId = studentId;
    this._dataAnswers = dataAnswers;
  }

  /* tslint:disable */
  static toObject(object: any): ResultQuestionnaire {
    /* tslint:enable */
    let result: ResultQuestionnaire = new ResultQuestionnaire();
    if (object != null) {
      result.id = object.id;
      result.questionnaireName = object.questionnaireName;
      result.questionnaireId = object.questionnaireId;
      result.numTotalQuestions = object.numTotalQuestions;
      result.numAnswerCorrect = object.numAnswerCorrect;
      result.numAnswerNoCorrect = object.numAnswerNoCorrect;
      result.finalNote = object.finalNote;
      result.studentId = object.studentId;
      result.dataAnswers = object.dataAnswers;
    }
    return result;
  }

  public get id(): string {
    return this._id;
  }

  public set id(value: string) {
    this._id = value;
```

```
  }

  public get questionnaireName(): string {
    return this._questionnaireName;
  }

  public set questionnaireName(value: string) {
    this._questionnaireName = value;
  }

  public get questionnaireId(): number {
    return this._questionnaireId;
  }

  public set questionnaireId(value: number) {
    this._questionnaireId = value;
  }

  public get questionnaire(): Questionnaire {
    return this._questionnaire;
  }

  public set questionnaire(value: Questionnaire) {
    this._questionnaire = value;
  }

  public get numTotalQuestions(): number {
    return this._numTotalQuestions;
  }

  public set numTotalQuestions(value: number) {
    this._numTotalQuestions = value;
  }

  public get numAnswerCorrect(): number {
    return this._numAnswerCorrect;
  }

  public set numAnswerCorrect(value: number) {
    this._numAnswerCorrect = value;
  }

  public get numAnswerNoCorrect(): number {
    return this._numAnswerNoCorrect;
  }

  public set numAnswerNoCorrect(value: number) {
    this._numAnswerNoCorrect = value;
  }

  public get finalNote(): number {
    return this._finalNote;
  }

  public set finalNote(value: number) {
    this._finalNote = value;
  }

  public get studentId(): number {
    return this._studentId;
  }

  public set studentId(value: number) {
    this._studentId = value;
  }

  public get student(): Student {
    return this._student;
```

```
  }

  public set student(value: Student) {
    this._student = value;
  }

  public get dataAnswers(): Array<string> {
    return this._dataAnswers;
  }

  public set dataAnswers(value: Array<string>) {
    this._dataAnswers = value;
  } }
```

7.4. Classes aplicació mòbil

En aquest subcapítol es mostren les Classes de l'arquitectura de l'Aplicació Mòbil del projecte.

7.4.1. Classe Qüestionari

```
export class QuestionnairePage {

  public myQuestionnaire: Questionnaire;
  public questions: Array<Question>;
  public questionsAnswers: Array<Question>
  public myCredentials: Credentials;
  public indexNum: number;
  public numTotalQuestions: number;
  public numQuestions: number;
  public numAnswerCorrect: number;
  public numAnswerNoCorrect: number;
  public finalNote: number = 0;
  public questionsSend: Question;
  public resultQuestionnaire: ResultQuestionnaire;
  public dataAnswers  = [];

  questionForm;

  public timeInSeconds: number;
  public timer: PTimer;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public utilsService: UtilsService,
    public ionicService: IonicService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {

    this.questionForm = new FormGroup({
      "questionsSend":  new  FormControl({value:  this.questionsSend,  disabled:
false})
    });

    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.questions = this.navParams.data.questions;

    this.indexNum = this.navParams.data.indexNum;
    this.numTotalQuestions = this.questions.length;
```

```
    this.questionsSend = this.questions[this.indexNum];

    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.myCredentials = this.navParams.data.myCredentials;

  }

  /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {

    this.ionicService.removeLoading();
  }

  /**
   * This method returns the questions list of the
   * current questionnaire
   * @param {Refresher} Refresher element
   */
  private getQuestions(refresher?: Refresher): void {


this.questionnaireService.getMyQuestionnaireQuestions(this.myCredentials).fina
lly(() => {
      refresher ? refresher.complete() : null;
    }).subscribe(
      ((value: Array<Question>) => this.questions = value),
      error                                                                 =>
this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }


 /**
   * This method manages the call to the service for performing a doSubmitAnswer
   * against the public services
   */
  public doSubmitAnswer(event) {

    this.timer.hasStarted = false;
    this.timer.runTimer = false;

    this.dataAnswers.push(this.questionForm.value.questionsSend);
    this.indexNum += 1;

    if((this.indexNum) < this.numTotalQuestions){
        this.navController.setRoot(Questionnaire1Page,    {    myQuestionnaire:
this.myQuestionnaire,     myCredentials:     this.myCredentials,     questions:
this.questions,        indexNum:        this.indexNum,        numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
      ((value:                              Student)                          =>
this.navController.setRoot(CompletedQuestionnairePage,    {    student:    value,
myQuestionnaire:            this.myQuestionnaire,            numTotalQuestions:
this.numTotalQuestions,         numAnswerCorrect:         this.numAnswerCorrect,
numAnswerNoCorrect:    this.numAnswerNoCorrect,    finalNote:    this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>
```

```
this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
      }

      event.preventDefault();
    }


  /**
   * This method manages the call to the service for performing a
doSubmitEmptyAnswer
   */
    public doSubmitEmptyAnswer() {

    this.dataAnswers.push('empty');
    this.indexNum += 1;

    if((this.indexNum) < this.numTotalQuestions){
        this.navController.setRoot(Questionnaire1Page,    {    myQuestionnaire:
this.myQuestionnaire,     myCredentials:     this.myCredentials,     questions:
this.questions,        indexNum:        this.indexNum,        numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
      ((value:                              Student)                              =>
this.navController.setRoot(CompletedQuestionnairePage,    {    student:    value,
myQuestionnaire:            this.myQuestionnaire,            numTotalQuestions:
this.numTotalQuestions,         numAnswerCorrect:         this.numAnswerCorrect,
numAnswerNoCorrect:    this.numAnswerNoCorrect,    finalNote:    this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
      }
    }
}
```

7.4.2. Classe Qüestionari Imatge

```
export class QuestionnaireImagePage {

  public myQuestionnaire: Questionnaire;
  public questions: Array<Question>;
  public questionsAnswers: Array<Question>
  public myCredentials: Credentials;
  public indexNum: number;
  public numTotalQuestions: number;
  public numQuestions: number;
  public numAnswerCorrect: number;
  public numAnswerNoCorrect: number;
  public finalNote: number = 0;
  private questionsSend: Question;
  private resultQuestionnaire: ResultQuestionnaire;
  public dataAnswers  = [];

  questionForm;
```

```
  public timeInSeconds: number;
  public timer: PTimer;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public utilsService: UtilsService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {

    this.questionForm = new FormGroup({
      "questionsSend": new FormControl({value: this.questionsSend, disabled:
false})
    });

    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.questions = this.navParams.data.questions;
    this.indexNum = this.navParams.data.indexNum;
    this.numTotalQuestions = this.questions.length;
    this.questionsSend = this.questions[this.indexNum];
    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.myCredentials = this.navParams.data.myCredentials;
  }

  /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {

    this.ionicService.removeLoading();
  }

  /**
   * This method returns the questions list of the
   * current questionnaire
   * @param {Refresher} Refresher element
   */
  private getQuestions(refresher?: Refresher): void {


this.questionnaireService.getMyQuestionnaireQuestions(this.myCredentials).fina
lly(() => {
      refresher ? refresher.complete() : null;
    }).subscribe(
      ((value: Array<Question>) => this.questions = value),
      error                                                                =>
this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }


 /**
   * This method manages the call to the service for performing a doSubmitAnswer
   * against the public services
   */
  public doSubmitAnswer(event) {

    this.timer.hasStarted = false;
    this.timer.runTimer = false;

    this.dataAnswers.push(this.questionForm.value.questionsSend);
    this.indexNum += 1;
    if((this.indexNum) < this.numTotalQuestions){
```

```
      switch (this.questions[this.indexNum].type) {
          case 'textArea':
            this.navController.setRoot(QuestionnaireTextAreaPage,            {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          case 'image':
            this.navController.setRoot(QuestionnaireImage1Page,              {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          default:
            break;
        }

    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
      ((value:                              Student)                           =>
this.navController.setRoot(CompletedQuestionnaire1Page,   {   student:   value,
myQuestionnaire:           this.myQuestionnaire,           numTotalQuestions:
this.numTotalQuestions,        numAnswerCorrect:        this.numAnswerCorrect,
numAnswerNoCorrect:   this.numAnswerNoCorrect,   finalNote:   this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
      }

      event.preventDefault();
    }

    /**
   *  This  method  manages  the  call  to  the  service  for  performing  a
doSubmitEmptyAnswer
   */
    public doSubmitEmptyAnswer() {

    this.dataAnswers.push('empty');
    this.indexNum += 1;

    if((this.indexNum) < this.numTotalQuestions){

      switch (this.questions[this.indexNum].type) {
          case 'textArea':
            this.navController.setRoot(QuestionnaireTextAreaPage,            {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          case 'image':
            this.navController.setRoot(QuestionnaireImage1Page,              {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          default:
            break;
        }
```

```
    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
      ((value:                              Student)                            =>
this.navController.setRoot(CompletedQuestionnaire1Page,   {   student:   value,
myQuestionnaire:           this.myQuestionnaire,           numTotalQuestions:
this.numTotalQuestions,        numAnswerCorrect:        this.numAnswerCorrect,
numAnswerNoCorrect:   this.numAnswerNoCorrect,   finalNote:   this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));

      }
    }
}
```

7.4.3. Classe Qüestionari Text Area

```
export class QuestionnaireTextAreaPage {

  public myQuestionnaire: Questionnaire;
  public questions: Array<Question>;
  public questionsAnswers: Array<Question>
  public myCredentials: Credentials;
  public indexNum: number;
  public numTotalQuestions: number;
  public numQuestions: number;
  public numAnswerCorrect: number;
  public numAnswerNoCorrect: number;
  public finalNote: number = 0;
  private questionsSend: Question;
  private resultQuestionnaire: ResultQuestionnaire;
  public dataAnswers  = [];

  questionForm;

  public timeInSeconds: number;
  public timer: PTimer;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public utilsService: UtilsService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {

    this.questionForm = new FormGroup({
      "questionsSend":  new  FormControl({value:  this.questionsSend,  disabled:
false})
    });

    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.questions = this.navParams.data.questions;
    this.indexNum = this.navParams.data.indexNum;
    this.numTotalQuestions = this.questions.length;
    this.questionsSend = this.questions[this.indexNum];
    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
```

```
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.myCredentials = this.navParams.data.myCredentials;
  }

  /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {

    this.ionicService.removeLoading();
  }

  /**
   * This method returns the questions list of the
   * current questionnaire
   * @param {Refresher} Refresher element
   */
  private getQuestions(refresher?: Refresher): void {


this.questionnaireService.getMyQuestionnaireQuestions(this.myCredentials).fina
lly(() => {
      refresher ? refresher.complete() : null;
    }).subscribe(
      ((value: Array<Question>) => this.questions = value),
      error                                                                 =>
this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }


 /**
   * This method manages the call to the service for performing a doSubmitAnswer
   * against the public services
   */
  public doSubmitAnswer(event) {

    this.timer.hasStarted = false;
    this.timer.runTimer = false;

    this.dataAnswers.push(this.questionForm.value.questionsSend);

    if((this.indexNum) < this.numTotalQuestions){

        switch (this.questions[this.indexNum].type) {
          case 'textArea':
            this.navController.setRoot(QuestionnaireTextArea1Page,            {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          case 'image':
            this.navController.setRoot(QuestionnaireImagePage,                {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          default:
            break;
        }

    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
```

```
      ((value:                              Student)                              =>
this.navController.setRoot(CompletedQuestionnaire1Page,   {   student:   value,
myQuestionnaire:           this.myQuestionnaire,           numTotalQuestions:
this.numTotalQuestions,        numAnswerCorrect:        this.numAnswerCorrect,
numAnswerNoCorrect:   this.numAnswerNoCorrect,   finalNote:   this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
      }

      event.preventDefault();
    }

   /**
   *  This  method  manages  the  call  to  the  service  for  performing  a
doSubmitEmptyAnswer
   */
    public doSubmitEmptyAnswer() {

    this.dataAnswers.push('empty');
    this.indexNum += 1;

    if((this.indexNum) < this.numTotalQuestions){

        switch (this.questions[this.indexNum].type) {
          case 'textArea':
            this.navController.setRoot(QuestionnaireTextArea1Page,           {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          case 'image':
            this.navController.setRoot(QuestionnaireImagePage,               {
myQuestionnaire:   this.myQuestionnaire,   myCredentials:   this.myCredentials,
questions:   this.questions,   indexNum:   this.indexNum,   numAnswerCorrect:
this.numAnswerCorrect,       numAnswerNoCorrect:       this.numAnswerNoCorrect,
dataAnswers: this.dataAnswers });
            break;
          default:
            break;
        }

    }else{

this.questionnaireService.getMyStudent(this.utilsService.currentUser.userId).s
ubscribe(
      ((value:                              Student)                              =>
this.navController.setRoot(CompletedQuestionnaire1Page,   {   student:   value,
myQuestionnaire:           this.myQuestionnaire,           numTotalQuestions:
this.numTotalQuestions,        numAnswerCorrect:        this.numAnswerCorrect,
numAnswerNoCorrect:   this.numAnswerNoCorrect,   finalNote:   this.finalNote,
dataAnswers:  this.dataAnswers,  myQuestions:  this.questions,  myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
      }
}
```

7.4.4. Classe Resultats Qüestionari

```
export class ResultQuestionnairePage {

  public myResults: ResultQuestionnaire;
  public student: Student;
  public result: ResultQuestionnaire;
  public myQuestions: Array<Question>;
  public myQuestionnaire: Questionnaire;
  public myQuestionsCorrectAnswers: Array<Question>;
  public myCredentials: Credentials;
  public myAnswers: Array<string>;
  public dataAnswers: Array<string>;

  public numTotalQuestions: number = 0;
  public numAnswerCorrect: number = 0;
  public numAnswerNoCorrect: number = 0;
  public finalNote: number = 0;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {
    this.myQuestionsCorrectAnswers=
    this.navParams.data.myQuestionsCorrectAnswers;
    this.student = this.navParams.data.student;
    this.myQuestions = this.navParams.data.myQuestions;
    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.myCredentials = this.navParams.data.myCredentials;
    this.numTotalQuestions = this.navParams.data.numTotalQuestions;
    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
    this.finalNote = this.navParams.data.finalNote;
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.getCorrectionQuestionnaire();
  }

  /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {
    this.ionicService.removeLoading();
  }

  /**
   * Method for displaying the MenuPage page
   */
  public outQuestionnaire(event) {
    this.navController.setRoot(MenuPage);
  }

  /**
  * Method to correct the results of the questionnaire
  */
  public getCorrectionQuestionnaire(): void {

    for(var i = 0; i < this.numTotalQuestions; i++){
      if(this.dataAnswers[i]===
this.myQuestionsCorrectAnswers[i].correctAnswer[0].name){
        this.numAnswerCorrect += 1;
      }
      else{
        this.numAnswerNoCorrect += 1;
```

```
      }
    }

    this.finalNote = this.numAnswerCorrect - this.numAnswerNoCorrect;

      this.questionnaireService.saveResults(this.student,
this.myQuestionnaire,    this.myQuestionnaire.name,    this.myQuestionnaire.id,
this.numTotalQuestions,     this.numAnswerCorrect,     this.numAnswerNoCorrect,
this.finalNote, this.dataAnswers).subscribe(
      ((value: ResultQuestionnaire) => this.result = value),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }
}
```

7.4.5. Classe Get Qüestionari

```
export class GetQuestionnairePage {

  public credentials: Credentials = new Credentials();
  public questions: Array<Question>;
  public answers: Array<Answer>;
  public myQuestionnaire: Questionnaire;
  public numAnswerCorrect: number = 0;
  public numAnswerNoCorrect: number = 0;
  public indexNum: number = 0;
  public dataAnswers  = [];

  constructor(
    public navController: NavController,
    public questionnaireService: QuestionnaireService,
    public ionicService: IonicService,
    public utilsService: UtilsService,
    public translateService: TranslateService,
    public menuController: MenuController) {

    // TODO: remove this
    switch (utilsService.role) {
      case Role.STUDENT:
        this.credentials.username = 'student-1';
        this.credentials.id = this.credentials.id;
        break;
      case Role.TEACHER:
        this.credentials.username = 'teacher-1';
        this.credentials.id = this.credentials.id;
        break;
      case Role.SCHOOLADMIN:
        this.credentials.username = 'school-admin-1';
        this.credentials.id = this.credentials.id;
        break;
      default:
        break;
    }

  }

    /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {
```

```
    this.menuController.enable(true);
  }

  /**
   *  This  method  manages  the  call  to  the  service  for  performing  a
getQuestionnaire
   * against the public services
   */
  public getQuestionnaire(): void {

    this.questionnaireService.getMyQuestionnaire(this.credentials).subscribe(
      ((value: Questionnaire) => this.myQuestionnaire = value),
    error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));

this.questionnaireService.getMyQuestionnaireQuestions(this.credentials).subscr
ibe(
      ((value: Array<Question>) => {

        switch (value[0].type) {
          case 'test':
            this.navController.setRoot(QuestionnairePage, { questions: value,
myCredentials:   this.credentials,   myQuestionnaire:   this.myQuestionnaire,
indexNum:     this.indexNum,     numAnswerCorrect:     this.numAnswerCorrect,
numAnswerNoCorrect: this.numAnswerNoCorrect, dataAnswers: this.dataAnswers });
            break;
          case 'textArea':
            this.navController.setRoot(QuestionnaireTextAreaPage, { questions:
value, myCredentials: this.credentials, myQuestionnaire: this.myQuestionnaire,
indexNum:     this.indexNum,     numAnswerCorrect:     this.numAnswerCorrect,
numAnswerNoCorrect: this.numAnswerNoCorrect, dataAnswers: this.dataAnswers });
            break;
          case 'image':
            this.navController.setRoot(QuestionnaireImagePage,   {   questions:
value, myCredentials: this.credentials, myQuestionnaire: this.myQuestionnaire,
indexNum:     this.indexNum,     numAnswerCorrect:     this.numAnswerCorrect,
numAnswerNoCorrect: this.numAnswerNoCorrect, dataAnswers: this.dataAnswers });
            break;
          default:
            break;
        }
      }),
    error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));

  }
}
```

7.4.6. Classe Qüestionari complert

```
export class CompletedQuestionnairePage {

  public student: Student;
  public numTotalQuestions: number;
  public numAnswerCorrect: number;
  public numAnswerNoCorrect: number;
  public finalNote: number;
  public myQuestions: Array<Question>;
  public dataAnswers  = [];
  public myQuestionnaire: Questionnaire;
```

```
  public myCredentials: Credentials;

  constructor(
    public navParams: NavParams,
    public navController: NavController,
    public ionicService: IonicService,
    public questionnaireService: QuestionnaireService,
    public translateService: TranslateService) {

    this.student = this.navParams.data.student;
    this.numTotalQuestions = this.navParams.data.numTotalQuestions;
    this.numAnswerCorrect = this.navParams.data.numAnswerCorrect;
    this.numAnswerNoCorrect = this.navParams.data.numAnswerNoCorrect;
    this.finalNote = this.navParams.data.finalNote;
    this.dataAnswers = this.navParams.data.dataAnswers;
    this.myQuestionnaire = this.navParams.data.myQuestionnaire;
    this.myQuestions = this.navParams.data.myQuestions;
    this.myCredentials = this.navParams.data.myCredentials;

  }

  /**
   * Fires when the page appears on the screen.
   * Used to get all the data needed in page
   */
  public ionViewDidEnter(): void {
    this.ionicService.removeLoading();
  }

  /**
   * This method returns the questions list of the
   * current questionnaire
   * @param Array<Question>
   */
  public getResults(event) {

this.questionnaireService.getQuestionsAnswersCorrectAnswers(this.myCredentials
).subscribe(
      ((value:                        Array<Question>)                        =>
this.navController.setRoot(ResultQuestionnairePage,                          {
myQuestionsCorrectAnswers:  value,  student:  this.student,  myQuestionnaire:
this.myQuestionnaire,        numTotalQuestions:        this.numTotalQuestions,
numAnswerCorrect:          this.numAnswerCorrect,          numAnswerNoCorrect:
this.numAnswerNoCorrect,      finalNote:      this.finalNote,      dataAnswers:
this.dataAnswers,      myQuestions:      this.myQuestions,      myCredentials:
this.myCredentials })),
      error =>

this.ionicService.showAlert(this.translateService.instant('APP.ERROR'),
error));
  }
}
```

7.5. Imatges de l'aplicació mòbil

En aquest subcapítol es mostren les imatges de l'Aplicació Mòbil del projecte.

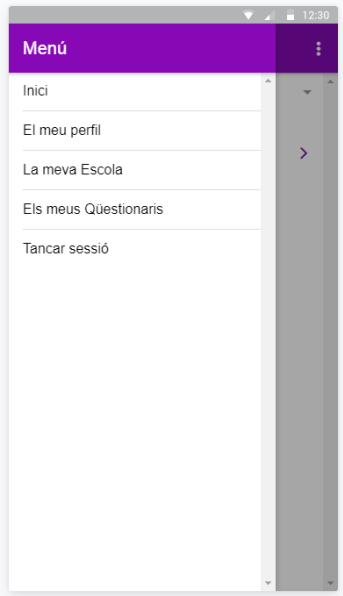


Figura 7.1 Mòbil (Menú)

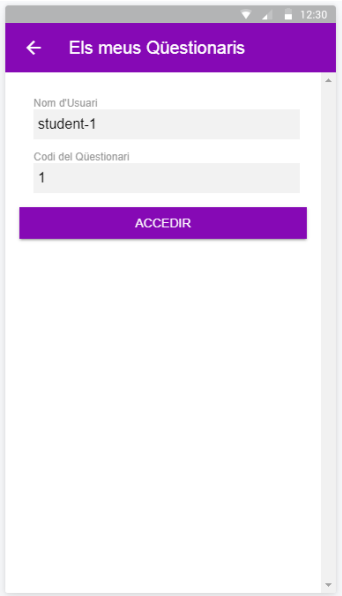


Figura 7.2 Mòbil (Get Questionnaire)

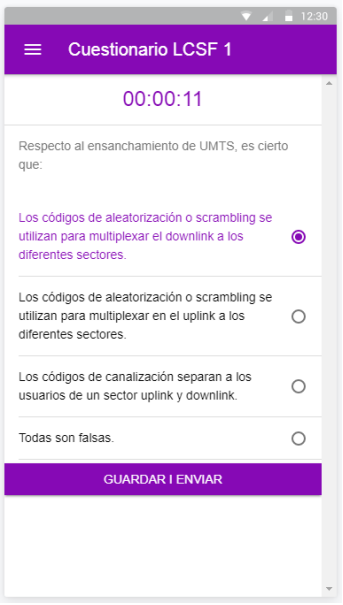


Figura 7.3 Mòbil (Question Test)

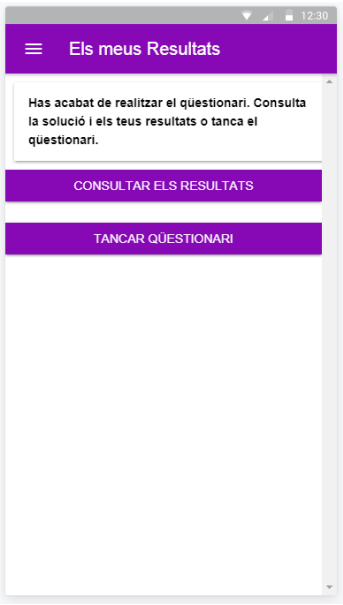


Figura 7.4 Mòbil (End Questionnaire Test)

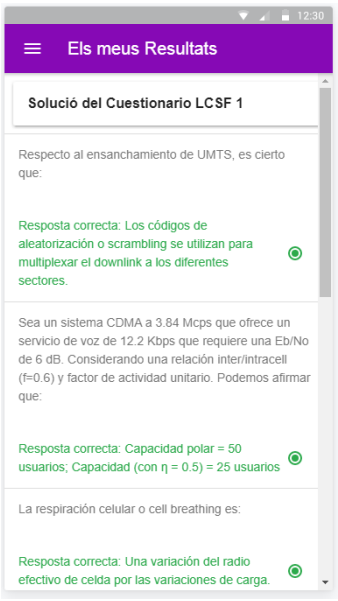


Figura 7.5 Mòbil
(Results Questionnaire Test)

Figura 7.6 Mòbil
(Results Questionnaire Test)

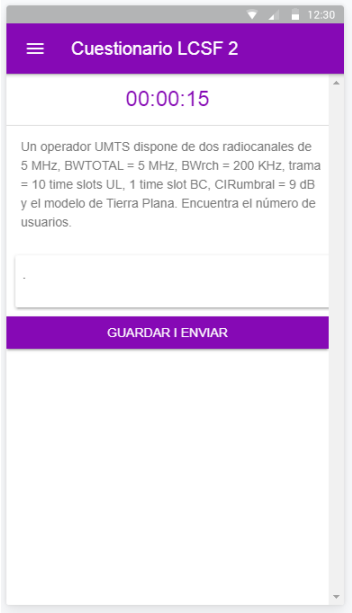


Figura 7.7 Mòbil (Question TextArea)

Figura 7.8 Mòbil (Question Image 1)

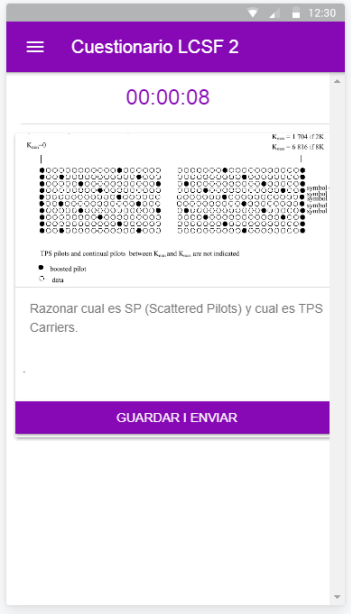


Figura 7.9 Mòbil
(Question Image 2)

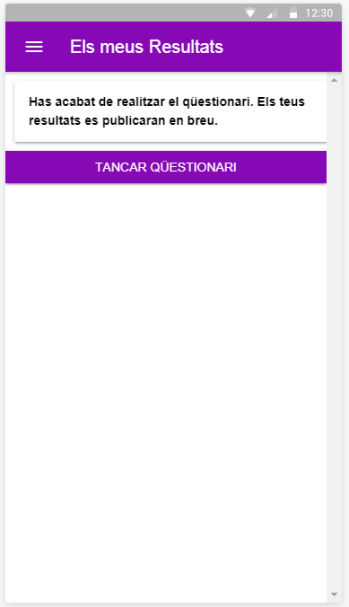


Figura 7.10 Mòbil
(End Questionnaire TextArea)

7.6. JSONs Panell Administració multiidioma

En aquest subcapítol es mostren els JSONs del projecte que fan possible que el Panell d'Administració sigui multiidioma.

7.6.1. Json idioma Català

```
{
  "APP": {
    "NAME": "Classpip",
    "CANCEL": "Cancel·lar",
    "LOGOUT": "Tancar sessió",
    "LEGAL": "Avís Legal",
    "PRIVACY": "Privacitat",
    "MARK": "Classpip ©2016-2017. Codi amb llicència sota una llicència Apache-
2"
  },
  "HOME": {
    "TITLE": "Inici",
    "TWITTER": "TWITTER",
    "FACEBOOK": "FACEBOOK",
    "WEBSITE": "WEBSITE",
    "LANGUAGE": "Escull l'idioma"
  },
  "LOGIN": {
    "USERNAME": "Nom d'usuari",
    "PASSWORD": "Contrasenya",
    "ROLE": "Rol",
    "LOGIN": "Entrar",
    "FORGOT": "He oblidat la contrasenya",
    "REGISTER": "Necessito crear un nou compte",
    "TEACHER": "Docent",
    "STUDENT": "Estudiant",
```

```
    "SCHOOLADMIN": "Director d'escola"
  },
  "GROUPS": {
    "TITLE": "Grups"
  },
  "QUESTIONNAIRES": {
    "TITLE": "Qüestionaris",
    "ID": "ID del Qüestionari:",
    "DELETE": "Eliminar Qüestionaris",
    "CREATE": "Crear nou Qüestionaris"
  },
  "QUESTIONNAIRE": {
    "RESULTS": "Resultats Qüestionaris",
    "ID_STUDENT": "ID Estudiant",
    "STUDENT": "Estudiant",
    "CORRECT_ANSWER": "Preguntes encertades",
    "NO_CORRECT_ANSWER": "Preguntes incorrectes",
    "RESULT": "Resultat",
    "QUESTIONS": "Preguntes",
    "ANSWERS": "Respostes",
    "DELETE1": "Segur que vols eliminar el Qüestionari",
    "DELETE2": "Eliminar",
    "CANCEL": "Cancel·lar",
    "QUESTION": "Pregunta",
    "ANSWER1": "Resposta 1",
    "ANSWER2": "Resposta 2",
    "ANSWER3": "Resposta 3",
    "ANSWER4": "Resposta 4",
    "CORRECTANSWER": "Resposta Correcta",
    "NEXT": "Següent",
    "BACK": "Enrere",
    "ANSWER_OPEN": "Pregunta amb resposta oberta",
    "QUESTION_IMAGE": "Pregunta amb imatge",
    "TEXT1": "Aquí està la fletxa desplegable ^",
    "TEXT2": "Tipus de pregunta",
    "URL": "URL de la imagen",
    "TEXT3": "Max 40 caràcters",
    "TEXT4": "Nom del Qüestionari",
    "TEXT5": "En segons",
    "TEXT6": "Temps total del Qüestionari",
    "TEXT7": "Max 100 preguntes",
    "TEXT8": "Nombre total de preguntes del Qüestionari",
    "TEXT9": "Tipus de Qüestionari",
    "TEXT10": "Resposta oberta",
    "TEXT11": "Resposta múltiple",
    "TEXT12": "Introduir la data"
  },
  "ERROR": {
    "LOGIN_FAILED": "El nom d'usuari i/o contrasenya són invàlides.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "El teu correu no ha estat verificat
encara, si us plau, revisa el teu correu.",
    "INTERNAL_ERROR": "Hi ha hagut un error intern al servidor. Si us plau
contacta amb el teu administrador. "
  }
}
```


7.6.2. Json idioma Castellà


```
{
  "APP": {
    "NAME": "Classpip",
    "CANCEL": "Cancelar",
    "LOGOUT": "Cerrar Sesión",
```

```
    "LEGAL": "Aviso Legal",
    "PRIVACY": "Privacidad",
    "MARK": "Classpip ©2016-2017. Código licenciado bajo una licencia Apache-2"
  },
  "HOME": {
    "TITLE": "Inicio",
    "TWITTER": "TWITTER",
    "FACEBOOK": "FACEBOOK",
    "WEBSITE": "WEBSITE",
    "LANGUAGE": "Escoge el idioma"
  },
  "LOGIN": {
    "USERNAME": "Nombre de usuario",
    "PASSWORD": "Contraseña",
    "ROLE": "Rol",
    "LOGIN": "Entrar",
    "FORGOT": "Olvidé mi contraseña",
    "REGISTER": "Necesito crear una nueva cuenta",
    "TEACHER": "Docente",
    "STUDENT": "Estudiante",
    "SCHOOLADMIN": "Director de Escuela"
  },
  "GROUPS": {
    "TITLE": "Grupos"
  },
  "QUESTIONNAIRES": {
    "TITLE": "Cuestionarios",
    "ID": "ID del Cuestionario:",
    "DELETE": "Eliminar Cuestionario",
    "CREATE": "Crear Nuevo Cuestionario"
  },
  "QUESTIONNAIRE": {
    "RESULTS": "Resultados Cuestionario",
    "ID_STUDENT": "ID Estudiante",
    "STUDENT": "Estudiante",
    "CORRECT_ANSWER": "Preguntas acertadas",
    "NO_CORRECT_ANSWER": "Preguntas incorrectas",
    "RESULT": "Resultado",
    "QUESTIONS": "Preguntas",
    "ANSWERS": "Respuestas",
    "DELETE1": "¿Seguro que quieres eliminar el Cuestionario",
    "DELETE2": "Eliminar",
    "CANCEL": "Cancelar",
    "QUESTION": "Pregunta",
    "ANSWER1": "Respuesta 1",
    "ANSWER2": "Respuesta 2",
    "ANSWER3": "Respuesta 3",
    "ANSWER4": "Respuesta 4",
    "CORRECTANSWER": "Respuesta Correcta",
    "NEXT": "Siguiente",
    "BACK": "Atrás",
    "ANSWER_OPEN": "Pregunta con respuesta abierta",
    "QUESTION_IMAGE": "Pregunta con imagen",
    "TEXT1": "Aquí está la flecha desplegable ^",
    "TEXT2": "Tipo de Pregunta",
    "URL": "URL de la imagen",
    "TEXT3": "Max 40 caracteres",
    "TEXT4": "Nombre del Cuestionario",
    "TEXT5": "En segundos",
    "TEXT6": "Tiempo total del Cuestionario",
    "TEXT7": "Max 100 preguntas",
    "TEXT8": "Número total de preguntas del Cuestionario",
    "TEXT9": "Tipo de Cuestionario",
    "TEXT10": "Respuesta abierta",
    "TEXT11": "Respuesta múltiple",
    "TEXT12": "Introducir la fecha"
  },
  "ERROR": {
```

```
    "LOGIN_FAILED": "El nombre de usuario y/o la contraseña son inválidas.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "Tu correo no ha sido verificado aún, por
favor, revisa tu correo.",
    "INTERNAL_ERROR": "Ha ocurrido un error interno en el servidor. Por favor
contacta con tu administrador. "
  }
}
```

7.6.2. Json idioma Anglès

```
{
  "APP": {
    "NAME": "Classpip",
    "CANCEL": "Cancel",
    "LOGOUT": "Logout",
    "LEGAL": "Legal warning",
    "PRIVACY": "Privacy",
    "MARK": "Classpip ©2016-2017. Code licensed under an Apache-2 License"
  },
  "HOME": {
    "TITLE": "Home",
    "TWITTER": "TWITTER",
    "FACEBOOK": "FACEBOOK",
    "WEBSITE": "WEBSITE",
    "LANGUAGE": "Choose the language"
  },
  "LOGIN": {
    "USERNAME": "Username",
    "PASSWORD": "Password",
    "ROLE": "Rol",
    "LOGIN": "Get in",
    "FORGOT": "I forgot my password",
    "REGISTER": "I need to create a new account",
    "TEACHER": "Teacher",
    "STUDENT": "Student",
    "SCHOOLADMIN": "School principal"
  },
  "GROUPS": {
    "TITLE": "Groups"
  },
  "QUESTIONNAIRES": {
    "TITLE": "Questionnaires",
    "ID": "Questionnaire ID:",
    "DELETE": "Delete Questionnaire",
    "CREATE": "Create New Questionnaire"
  },
  "QUESTIONNAIRE": {
    "RESULTS": "Results Questionnaire",
    "ID_STUDENT": "ID Student",
    "STUDENT": "Student",
    "CORRECT_ANSWER": "Successful questions",
    "NO_CORRECT_ANSWER": "Incorrect questions",
    "RESULT": "Result",
    "QUESTIONS": "Questions",
    "ANSWERS": "Answers",
    "DELETE1": "Are you sure you want to delete the Questionnaire",
    "DELETE2": "Delete",
    "CANCEL": "Cancel",
    "QUESTION": "Question",
    "ANSWER1": "Answer 1",
    "ANSWER2": "Answer 2",
    "ANSWER3": "Answer 3",
    "ANSWER4": "Answer 4",
    "CORRECTANSWER": "Correct answer",
    "NEXT": "Next",
```

```
    "BACK": "Back",
    "ANSWER_OPEN": "Question with open answer",
    "QUESTION_IMAGE": "Question with image",
    "TEXT1": "Here is the drop-down arrow ^",
    "TEXT2": "Type of Question",
    "URL": "Image URL",
    "TEXT3": "Max 40 characters",
    "TEXT4": "Questionnaire Name",
    "TEXT5": "In seconds",
    "TEXT6": "Total time of the Questionnaire",
    "TEXT7": "Max 100 Questions",
    "TEXT8": "Total number of questions in the Questionnaire",
    "TEXT9": "Type of Questionnaire",
    "TEXT10": "Open answer",
    "TEXT11": "Multiple answer",
    "TEXT12": "Enter date"
  },
  "ERROR": {
    "LOGIN_FAILED": "The username and/or password are invalid.",
    "LOGIN_FAILED_EMAIL_NOT_VERIFIED": "Your email has not been verified yet,
please check your email.",
    "INTERNAL_ERROR": "An internal error has occurred on the server. Please
contact your administrator."
  }
}
```

7.7. Classes Panell Administració

En aquest subcapítol es mostren les Classes de l'arquitectura del Panell d'Administració del projecte.

7.7.1. Classe Qüestionari

```
export class QuestionnaireComponent implements OnInit {

  public myQuestionnaire: Questionnaire;
  public myQuestions: Array<Question>;
  public myAnswers1: Array<Answer>;
  public myAnswers2: Array<Answer>;
  public myAnswers3: Array<Answer>;
  public myAnswers4: Array<Answer>;
  private returnUrl: string;
  public questionnaireId: string;
  private sub: any;
  public items: string[] = [];


  constructor(
    public route: ActivatedRoute,
    public router: Router,
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog) {

    this.utilsService.currentUser                                          =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
```

```
  }

 public ngOnInit(): void {

  this.returnUrl     =     this.route.snapshot.queryParams['returnUrl']     ||
'/questionnaireResults';

  this.sub = this.route.params.subscribe(params => {
      this.questionnaireId = params['id'];
    });

    if (this.utilsService.role === Role.TEACHER) {


this.questionnaireService.getMyQuestionnaire(this.questionnaireId).subscribe(
        ((questionnaire: Questionnaire) => {
          this.myQuestionnaire = questionnaire;
          this.loadingService.hide();
        }),
        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));


this.questionnaireService.getMyQuestionnaireQuestions(this.questionnaireId).su
bscribe(
        ((questions: Array<Question>) => {
          this.myQuestions = questions;
          this.loadingService.hide();
        }),
        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

  public goToResultQuestionnaire(): void {

    this.router.navigate([this.returnUrl, this.questionnaireId]);
  }
}
```

7.7.2. Classe Qüestionaris

```
export class QuestionnairesComponent implements OnInit {

  public questionnaires: Array<Questionnaire>;
  private returnUrl: string;

  animal: number;
  name: number;
  resultCreate: number;

  constructor(
    public route: ActivatedRoute,
    public router: Router,
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog) {
```

```
    this.utilsService.currentUser                                                    =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
  }

 public ngOnInit(): void {

    this.returnUrl     =     this.route.snapshot.queryParams['returnUrl']     ||
'/questionnaire';

    if (this.utilsService.role === Role.TEACHER) {

      this.loadingService.show();

      this.questionnaireService.getQuestionnaires().subscribe(
        ((questionnaires: Array<Questionnaire>) => {
          this.questionnaires = questionnaires;
          this.loadingService.hide();
        }),

        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

 public openDialog(): void {

    let dialogRef = this.dialog.open(DeleteQuestionnaireComponent, {
      height: '400px',
      width: '600px',
      data: { name: this.name, animal: this.animal }
    });

    dialogRef.afterClosed().subscribe(result => {

      this.animal = result;
      this.ngOnInit();
    });
  }

 public refresh(): void {

    window.location.reload();
  }

 public createQuestionnaire() {

    const dialogRef = this.dialog.open(CreateQuestionnaireComponent, {
      height: '600px',
      width: '700px',
    });

    dialogRef.afterClosed().subscribe(result => {
      this.resultCreate = result;
      this.ngOnInit();
    });
  }

  public goToQuestionnaireDetail(questionnaire: Questionnaire): void {

    this.router.navigate([this.returnUrl, questionnaire.id]);
  }
}
```

7.7.3. Classe Crear Qüestionari

```
export class CreateQuestionnaireComponent implements OnInit {

  public questionnaires: Array<Questionnaire>;
  public myQuestionnaire: Questionnaire;
  public stringData = [];
  public numberData = [];

  public name: string;
  public time: number;
  public number: number;
  public date: string;
  public selected: string;
  public result: string;
  public num = 0;

  constructor(
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog,
    public dialogRef: MatDialogRef<CreateQuestionnaireComponent>,
    @Inject(MAT_DIALOG_DATA) public data: any) {

    this.utilsService.currentUser                                          =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
  }

  ngOnInit(): void {

    if (this.utilsService.role === Role.TEACHER) {

      this.loadingService.show();
      this.questionnaireService.getQuestionnaires().subscribe(
        ((questionnaires: Array<Questionnaire>) => {
          this.questionnaires = questionnaires;
          this.loadingService.hide();
        }),
        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

  cancel(): void {
    this.dialogRef.close();
  }

  createQuestionnaire(): void {

    switch (this.selected) {
          case 'optionRespuestaMultiple':
            this.stringData.push(this.name);
            this.stringData.push(this.date);
            this.numberData.push(this.time);
            this.numberData.push(this.number);

            /*Save new Questionnaire*/

this.questionnaireService.saveQuestionnaire(this.stringData).subscribe(
              ((value: Questionnaire) => this.myQuestionnaire = value),
            ((error: Questionnaire) => {
              this.loadingService.hide();
```

```
              this.alertService.show(error.toString());
            }));

            let                          dialogRef1                          =
this.dialog.open(CreateQuestionnaireTest1Component, {
            height: '600px',
            width: '700px',
            data:  {stringData:  this.stringData,  numberData:  this.numberData,
num: this.num}
            });

            dialogRef1.afterClosed().subscribe(result => {
              this.result = result;
              this.ngOnInit();
            });
            this.cancel();
            break;
          case 'optionRespuestaAbierta':
            this.stringData.push(this.name);
            this.stringData.push(this.date);
            this.numberData.push(this.time);
            this.numberData.push(this.number);

            /*Save new Questionnaire*/

this.questionnaireService.saveQuestionnaire(this.stringData).subscribe(
              ((value: Questionnaire) => this.myQuestionnaire = value),
            ((error: Questionnaire) => {
              this.loadingService.hide();
              this.alertService.show(error.toString());
            }));
            let                          dialogRef2                          =
this.dialog.open(CreateQuestionnaireTextArea1Component, {
              height: '600px',
              width: '700px',
              data: {stringData: this.stringData, numberData: this.numberData,
num: this.num}
            });

            dialogRef2.afterClosed().subscribe(result => {
              this.result = result;
              this.ngOnInit();
            });
            this.cancel();
            break;
          default:
            break;
        }
  }
}
```

7.7.4. Classe Crear Qüestionari Test

```
export class CreateQuestionnaireTest1Component implements OnInit {

  public questionnaires: Array<Questionnaire>;
  public myQuestionnaire: Questionnaire;
  public myQuestions: Array<Question>;
  public stringData = [];
  public numberData = [];
  public questionData = [];
  public numberQuestions: number;
  public num: number;
```

```
  public result: string;
  public question: string;
  public answer1: string;
  public answer2: string;
  public answer3: string;
  public answer4: string;
  public correctAnswer: string;

  constructor(
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog,
    public dialogRef: MatDialogRef<CreateQuestionnaireTest1Component>,
    @Inject(MAT_DIALOG_DATA) public data: any) {

    this.utilsService.currentUser                                          =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
    this.stringData.push(data.stringData[0]);
    this.stringData.push(data.stringData[1]);
    this.numberData.push(data.numberData[0]);
    this.numberData.push(data.numberData[1]);
    this.numberQuestions = data.numberData[1];
    this.num = data.num;
  }

  ngOnInit(): void {

    if (this.utilsService.role === Role.TEACHER) {

      this.questionnaireService.getQuestionnaires().subscribe(
        ((questionnaires: Array<Questionnaire>) => {
          this.questionnaires = questionnaires;
          this.loadingService.hide();
        }),
        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

  cancel(): void {
    this.dialogRef.close();
  }

  createQuestionnaire(): void {

    this.num += 1;
    this.questionData.push(this.question);
    this.questionData.push(this.answer1);
    this.questionData.push(this.answer2);
    this.questionData.push(this.answer3);
    this.questionData.push(this.answer4);
    this.questionData.push(this.correctAnswer);
    this.questionData.push('test');

    if ( (this.num) < this.numberQuestions) {


this.questionnaireService.saveQuestionAnswersCorrectAnswer(this.numberData,
this.questionData).subscribe(
        ((value: Array<Question>) => this.myQuestions = value),
      ((error: Question) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
```

```
        }));

      let dialogRef = this.dialog.open(CreateQuestionnaireTest2Component, {
            height: '600px',
            width: '700px',
            data: {stringData: this.stringData, numberData: this.numberData,
questionData: this.questionData, num: this.num}
            });

      this.cancel();

    } else {


this.questionnaireService.saveQuestionAnswersCorrectAnswer(this.numberData,
this.questionData).subscribe(
        ((value: Array<Question>) => this.myQuestions = value),
      ((error: Question) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
      }));

      this.dialogRef.afterClosed().subscribe(result => {
      this.result = result;
      this.ngOnInit();
      });
        this.cancel();
    }
  }
}
```

7.7.5. Classe Crear Qüestionari Text Area

```
export class CreateQuestionnaireTextArea1Component implements OnInit {

  public questionnaires: Array<Questionnaire>;
  public myQuestionnaire: Questionnaire;
  public myQuestion: Question;
  public stringData = [];
  public numberData = [];
  public questionData = [];
  public numberQuestions: number;
  public num: number;
  public result: string;
  public selected: string;
  public question: string;
  public answer1: string;
  public answer2: string;
  public answer3: string;
  public answer4: string;
  public correctAnswer: string;
  public urlImage: string;

  constructor(
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog,
    public dialogRef: MatDialogRef<CreateQuestionnaireTextArea1Component>,
    @Inject(MAT_DIALOG_DATA) public data: any) {

    this.utilsService.currentUser                                    =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
```

```
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
    this.stringData.push(data.stringData[0]);
    this.stringData.push(data.stringData[1]);
    this.numberData.push(data.numberData[0]);
    this.numberData.push(data.numberData[1]);
    this.numberQuestions = data.numberData[1];
    this.num = data.num;
  }

  ngOnInit(): void {

    if (this.utilsService.role === Role.TEACHER) {
      this.questionnaireService.getQuestionnaires().subscribe(
        ((questionnaires: Array<Questionnaire>) => {
          this.questionnaires = questionnaires;
          this.loadingService.hide();
        }),
        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

  cancel(): void {
    this.dialogRef.close();
  }

  createQuestionnaire(): void {
    this.num += 1;
    this.questionData.push(this.question);
    this.questionData.push(this.answer1);
    this.questionData.push(this.answer2);
    this.questionData.push(this.answer3);
    this.questionData.push(this.answer4);
    this.questionData.push(this.correctAnswer);
    this.questionData.push(this.selected);
    this.questionData.push(this.urlImage);

    if ( (this.num) < this.numberQuestions) {

      this.questionnaireService.postQuestionnaireQuestions(this.numberData,
this.questionData).subscribe(
        ((value: Question) => this.myQuestion = value),
      ((error: Question) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));

      let dialogRef = this.dialog.open(CreateQuestionnaireTextArea2Component, {
            height: '600px',
            width: '700px',
            data: {stringData: this.stringData, numberData: this.numberData,
questionData: this.questionData, num: this.num}
            });
      this.cancel();
    } else {
      this.questionnaireService.postQuestionnaireQuestions(this.numberData,
this.questionData).subscribe(
        ((value: Question) => this.myQuestion = value),
      ((error: Question) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
      }));

      this.dialogRef.afterClosed().subscribe(result => {
      this.result = result;
      this.ngOnInit();
```

```
      });
        this.cancel();
    }
  }
}
```

7.7.6. Classe Eliminar Qüestionari

```
export class DeleteQuestionnaireComponent implements OnInit {

  public questionnaires: Array<Questionnaire>;
  animal: string;
  name: number;
  result: string;

  constructor(

    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialogRef: MatDialogRef<DeleteQuestionnaireComponent>,
    @Inject(MAT_DIALOG_DATA) public data: any) {

    this.utilsService.currentUser                                          =
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
  }

  ngOnInit(): void {
    if (this.utilsService.role === Role.TEACHER) {

      this.loadingService.show();
      this.questionnaireService.getQuestionnaires().subscribe(
        ((questionnaires: Array<Questionnaire>) => {
          this.questionnaires = questionnaires;
          this.loadingService.hide();
        }),

        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }

  cancel(): void {
    this.dialogRef.close();
  }

  deleteQuestionnaire(): void {
    if (this.utilsService.role === Role.TEACHER) {

      this.loadingService.show();
      this.questionnaireService.deleteQuestionnaire(this.data.name).subscribe(
        result => this.result
      );
      this.cancel();
    }
  }
}
```

7.7.7. Classe Resultats Qüestionari

```
export class QuestionnaireResultsComponent implements OnInit {

  public resultsQuestionnaire: Array<ResultQuestionnaire>;
  public myQuestions: Array<Question>;
  public myAnswers1: Array<Answer>;
  public myAnswers2: Array<Answer>;
  public myAnswers3: Array<Answer>;
  public myAnswers4: Array<Answer>;

  private returnUrl: string;
  public questionnaireId: string;
  private sub: any;
  public items: string[] = [];

  constructor(
    public route: ActivatedRoute,
    public router: Router,
    public alertService: AlertService,
    public utilsService: UtilsService,
    public loadingService: LoadingService,
    public questionnaireService: QuestionnaireService,
    public dialog: MatDialog) {

    this.utilsService.currentUser=
Login.toObject(localStorage.getItem(AppConfig.LS_USER));
    this.utilsService.role = Number(localStorage.getItem(AppConfig.LS_ROLE));
  }

 public ngOnInit(): void {

  this.sub = this.route.params.subscribe(params => {

      this.questionnaireId = params['id'];
    });

    if (this.utilsService.role === Role.TEACHER) {


this.questionnaireService.getResultsQuestionnaire(this.questionnaireId).subscr
ibe(
        ((resultsQuestionnaire: Array<ResultQuestionnaire>) => {
          this.resultsQuestionnaire = resultsQuestionnaire;
          this.loadingService.hide();
        }),

        ((error: Response) => {
          this.loadingService.hide();
          this.alertService.show(error.toString());
        }));
    }
  }
}
```

7.8. Imatges del Panell Administració

En aquest subcapítol es mostren les imatges del Panell d'Administarció del projecte.

7.8.1. Flux per a crear un qüestionari amb multiresposta

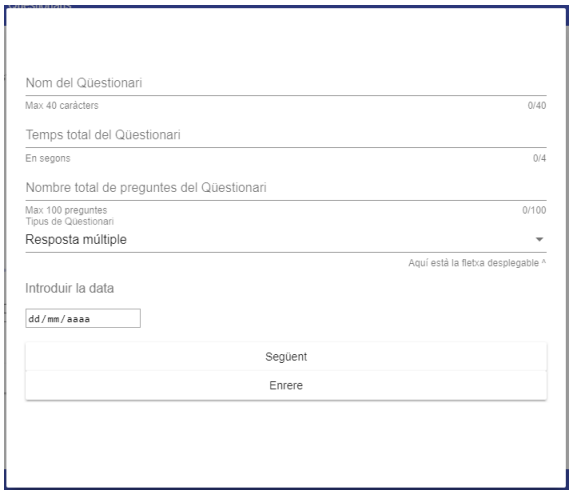


Figura 7.11 Panell Administració (Create Questionnaire 1)

Figura 7.12 Panell Administració (Create Question 1)

7.8.2. Flux per a crear un qüestionari amb resposta oberta

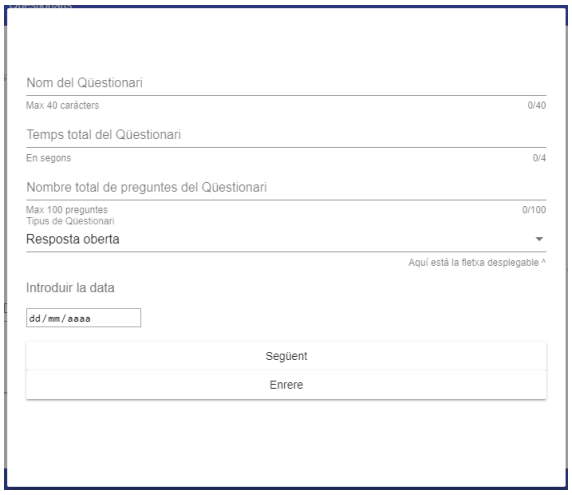


Figura 7.13 Panell Administració (Create Questionnaire 2)

Figura 7.14 Panell Administració (Create Question 2)

7.8.3. Flux per a crear un qüestionari amb imatge

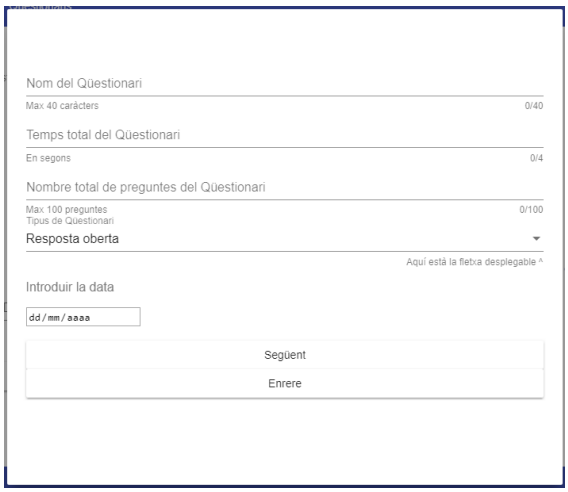


Figura 7.15 Panell Administració (Create Questionnaire 3)

Figura 7.16 Panell Administració (Create Question 3)

7.9. Models i Relacions

En aquest subcapítol es mostren les definicions de Models i Relacions dels models.

7.9.1. Definició qüestionari (questionnaire.json)

```
{
  "name": "Questionnaire",
  "plural": "questionnaires",
  "base": "PersistedModel",
  "idInjection": true,
  "options": {
    "validateUpsert": true
  },
  "properties": {
    "name": {
      "type": "string",
      "required": true
    }
  },
  "validations": [],
  "relations": {
    "teacher": {
      "type": "belongsTo",
      "model": "Teacher",
      "foreignKey": "teacherId"
    },
    "student": {
      "type": "belongsTo",
      "model": "Student",
      "foreignKey": "studentId"
```

```
    },
    "questions": {
      "type": "hasAndBelongsToMany",
      "model": "Question",
      "foreignKey": "questionId"
    }
  },
  "acls": [{
      "accessType": "*",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "DENY"
    },
    {
      "accessType": "READ",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "ALLOW"
    },
    {
      "accessType": "WRITE",
      "principalType": "ROLE",
      "principalId": "$owner",
      "permission": "ALLOW"
    }
  ],
  "methods": {}
}
```

7.9.2. Definició pregunta (question.json)

```
{
  "name": "Question",
  "plural": "questions",
  "base": "PersistedModel",
  "idInjection": true,
  "options": {
    "validateUpsert": true
  },
  "properties": {
    "name": {
      "type": "string",
      "required": true
    },
    "type": {
      "type": "string",
      "required": true
    },
    "image": {
      "type": "string",
      "required": false
    },
    "time": {
      "type": "number",
      "required": true
    }
  },
  "validations": [],
  "relations": {
    "questionnaire": {
      "type": "belongsTo",
      "model": "Questionnaire",
      "foreignKey": "questionnaireId"
    },
```

```
    "answers": {
      "type": "hasAndBelongsToMany",
      "model": "Answer",
      "foreignKey": "answerId"
    },
    "correctAnswers": {
      "type": "hasAndBelongsToMany",
      "model": "CorrectAnswer",
      "foreignKey": "correctAnswerId"
    }
  },
  "acls": [{
      "accessType": "*",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "DENY"
    },
    {
      "accessType": "READ",
      "principalType": "ROLE",
      "principalId": "$authenticated",
      "permission": "ALLOW"
    },
    {
      "accessType": "WRITE",
      "principalType": "ROLE",
      "principalId": "$owner",
      "permission": "ALLOW"
    }
  ],
  "methods": {}
}
```

7.9.3. Definició resposta (answer.json)

```
{
  "name": "Answer",
  "plural": "answers",
  "base": "PersistedModel",
  "idInjection": true,
  "options": {
    "validateUpsert": true
  },
  "properties": {
    "name": {
      "type": "string",
      "required": true
    }
  },
  "validations": [],
  "relations": {
    "question": {
      "type": "belongsTo",
      "model": "Question",
      "foreignKey": "questionId"
    }
  },
  "acls": [{
      "accessType": "*",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "DENY"
    },
    {
```

```
      "accessType": "READ",
      "principalType": "ROLE",
      "principalId": "$authenticated",
      "permission": "ALLOW"
    },
    {
      "accessType": "WRITE",
      "principalType": "ROLE",
      "principalId": "SYS-ADMIN",
      "permission": "ALLOW"
    }
  ],
  "methods": {}
}
```

7.9.4. Definició resposta correcta (CorrectAnswer.json)

```
{
  "name": "CorrectAnswer",
  "plural": "correctAnswers",
  "base": "PersistedModel",
  "idInjection": true,
  "options": {
    "validateUpsert": true
  },
  "properties": {
    "name": {
      "type": "string",
      "required": true
    }
  },
  "validations": [],
  "relations": {
    "question": {
      "type": "belongsTo",
      "model": "Question",
      "foreignKey": "questionId"
    }
  },
  "acls": [{
      "accessType": "*",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "DENY"
    },
    {
      "accessType": "READ",
      "principalType": "ROLE",
      "principalId": "$authenticated",
      "permission": "ALLOW"
    },
    {
      "accessType": "WRITE",
      "principalType": "ROLE",
      "principalId": "SYS-ADMIN",
      "permission": "ALLOW"
    }
  ],
  "methods": {}
}
```

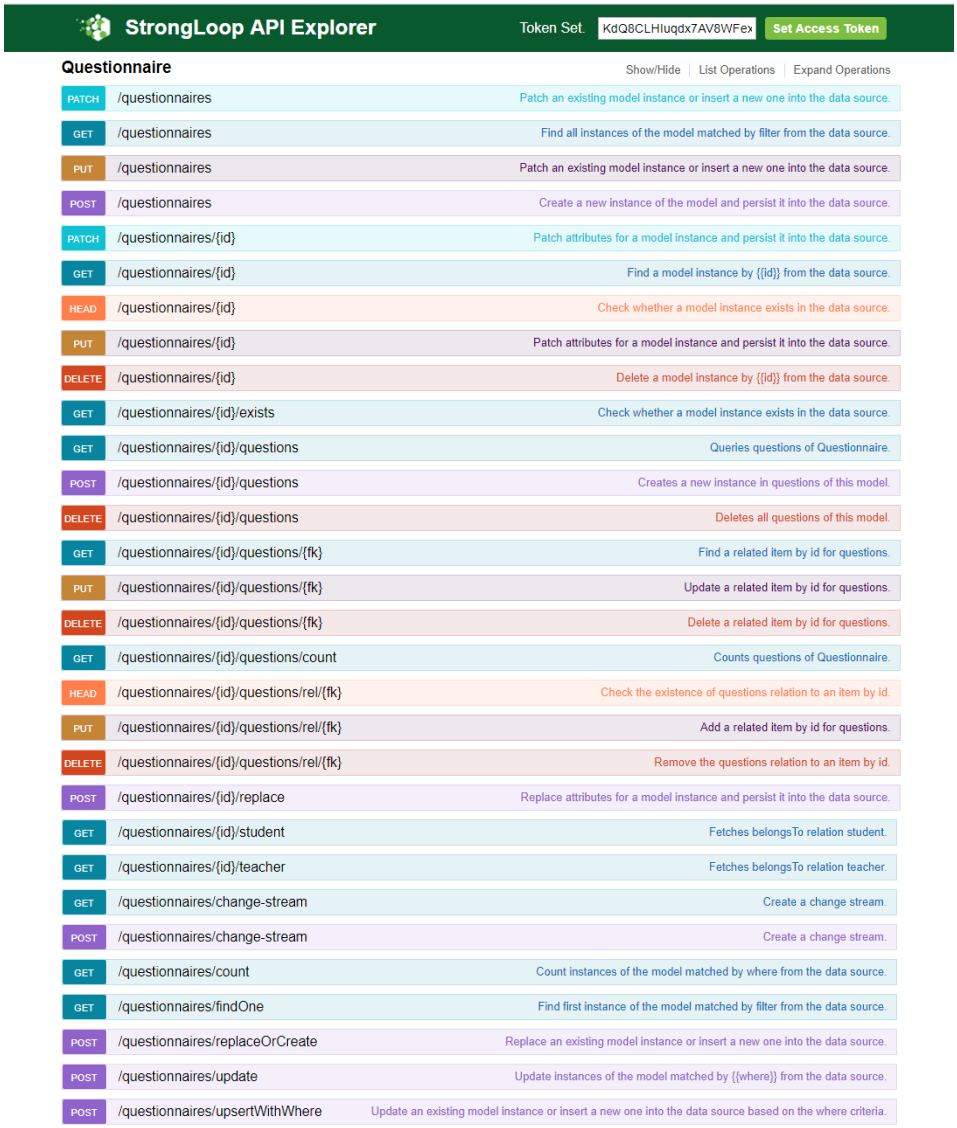
7.9.5. Definició resultat qüestionari (ResultQuestionnaire.json)

```
{
  "name": "ResultQuestionnaire",
  "base": "PersistedModel",
  "idInjection": true,
  "options": {
    "validateUpsert": true
  },
  "properties": {
    "questionnaireName": {
      "type": "string",
      "required": true
    },
    "numTotalQuestions": {
      "type": "number",
      "required": true
    },
    "numAnswerCorrect": {
      "type": "number",
      "required": true
    },
    "numAnswerNoCorrect": {
      "type": "number",
      "required": true
    },
    "finalNote": {
      "type": "number",
      "required": true
    },
    "dataAnswers": {
      "type": "string",
      "required": true
    }
  },
  "validations": [],
  "relations": {
    "questionnaire": {
      "type": "belongsTo",
      "model": "Questionnaire",
      "foreignKey": "questionnaireId"
    },
    "student": {
      "type": "belongsTo",
      "model": "Student",
      "foreignKey": "studentId"
    }
  },
  "acls": [{
      "accessType": "*",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "DENY"
    },
    {
      "accessType": "READ",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "ALLOW"
    },
    {
      "accessType": "WRITE",
      "principalType": "ROLE",
      "principalId": "$everyone",
      "permission": "ALLOW"
    }
  "methods": {} }
```

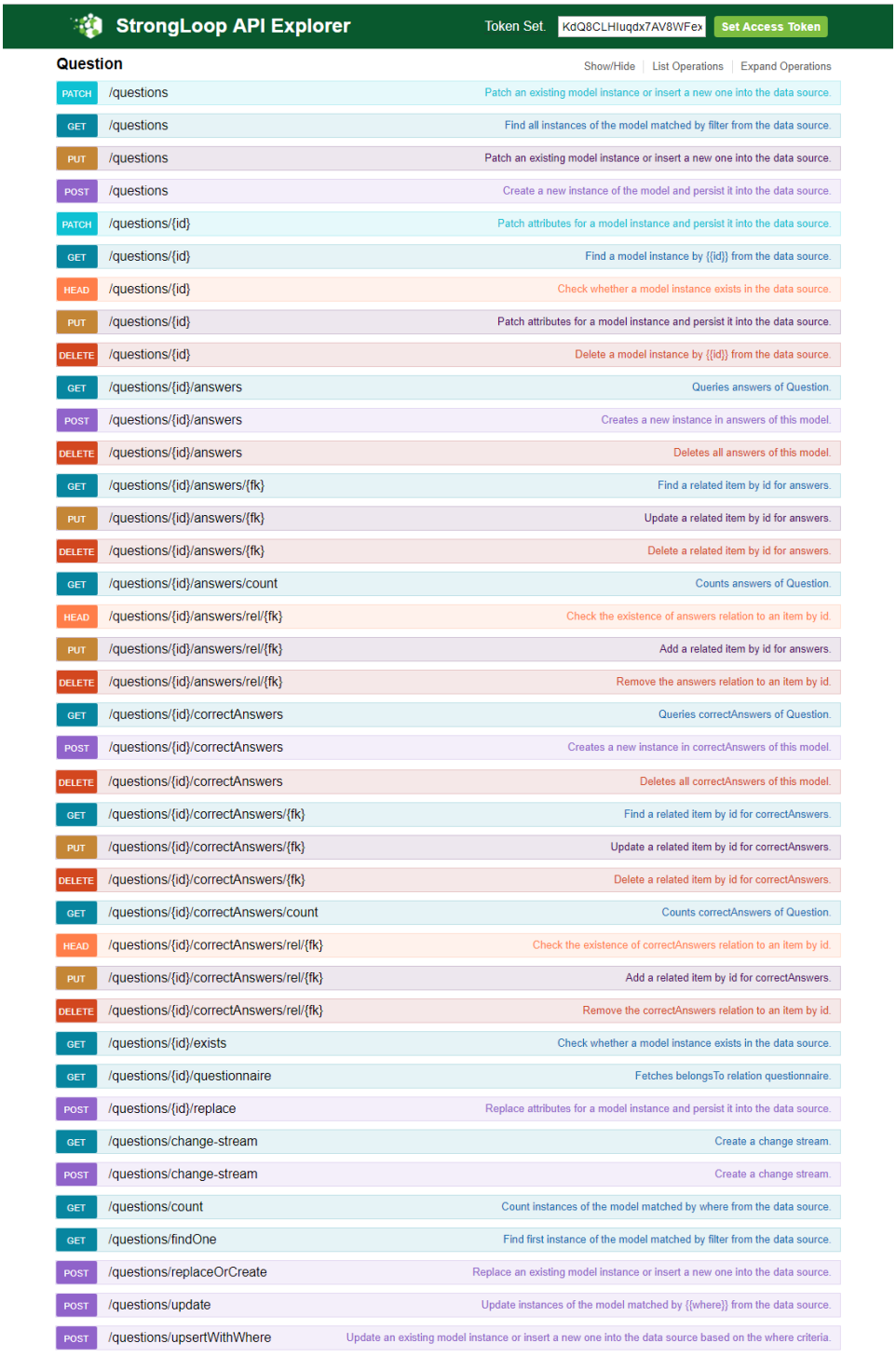
7.10. Serveis locals de l'explorador d'API de compilació

En aquest subcapítol es mostren els serveis de l'explorador de compilació.

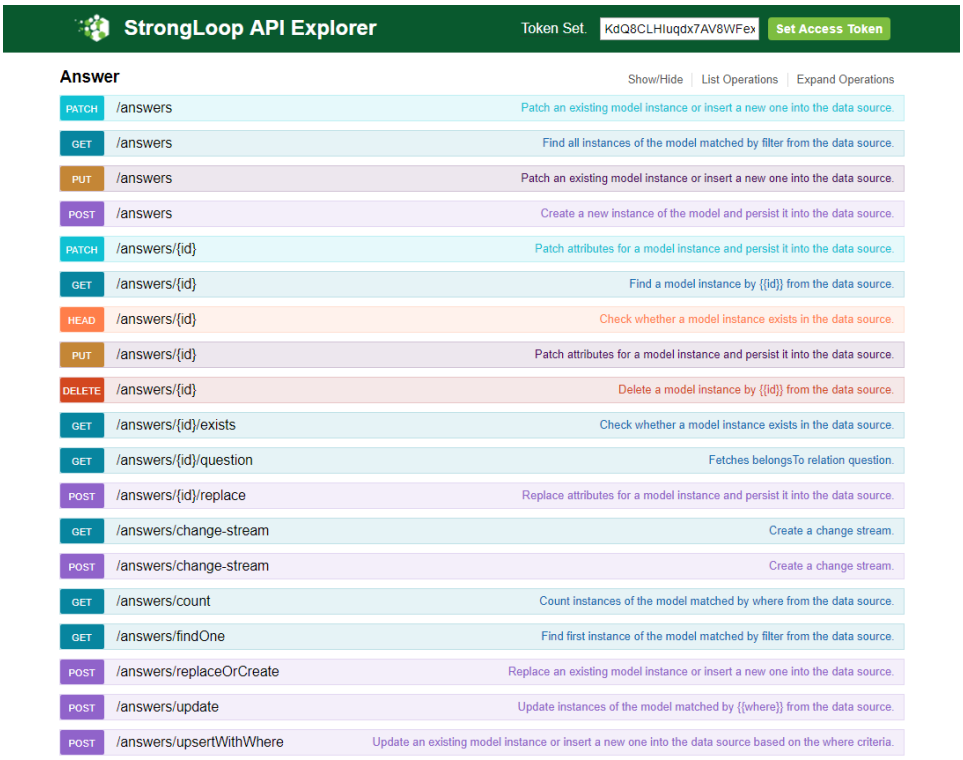
7.10.1. Interfície swagger del Qüestionari



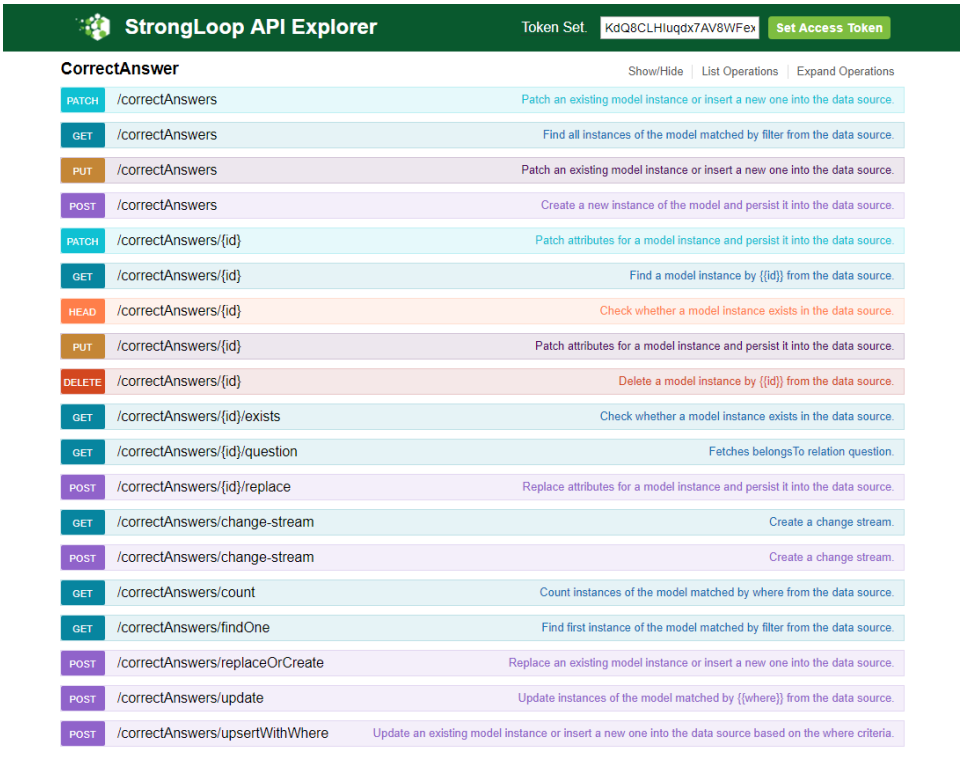
7.10.2. Interfície swagger de la Pregunta



7.10.3. Interfície swagger de la Resposta



7.10.4. Interfície swagger de la Resposta Correcta



7.10.5. Interfície swagger del Resultat del Qüestionari