TecnoCampus
Escola Superior Politècnica

Centre adscrit a la
upf. Universitat Pompeu Fabra Barcelona

Grau en Enginyeria Informàtica de Gestió i Sistemes d'Informació

APP EDUCATIVA DE PROGRAMACIÓ HTML5

Memòria

Sergi Cels Santiago

Tutor: Catalina Juan Nadal

4t curs

TecnoCampus
Mataró-Maresme

Abstract

This final degree project shows the creation and development of an Android application to learn HTML5 language programming with the main goal of gaining a basic knowledge of this language, providing accessibility to M-learning and promoting the growth of computer science. For this purpose, the official integrated development environment called Android Studio in Java has been used together with the remote database system MongoDB. An agile software development methodology has been followed and throughout the document, the tools, research, guidelines and techniques that have been used to carry out the application are observed.

Resum

Aquest treball de fi de grau mostra la creació i desenvolupament d'una aplicació Android per aprendre a programar en llenguatge HTML5, amb els objectius d'obtenir un coneixement base sobre aquest llenguatge, donar accessibilitat a l'aprenentatge M-learning i fomentar el creixement de la informàtica. Per fer-ho, s'ha utilitzat el entorn de desenvolupament integrat oficial anomenat Android Studio en Java juntament amb el sistema de base de dades remot MongoDB. S'ha seguit una metodologia de desenvolupament de software àgil i al llarg del document, s'observen les eines, investigació, guies i tècniques que s'han utilitzat per duu a terme la aplicació.

Resumen

Este trabajo de fin de grado muestra la creación y desarrollo de una aplicación Android para aprender a programar en lenguaje HTML5, con los objetivos de obtener un conocimiento base sobre este lenguaje, dar accesibilidad al aprendizaje M-learning y fomentar el crecimiento de la informática. Para ello, se ha utilizado el entorno de desarrollo integrado oficial llamado Android Studio en Java junto con el sistema de base de datos remoto MongoDB. Se ha seguido una metodología de desarrollo de software ágil y a lo largo del documento, se observan las herramientas, investigación, guías y técnicas que se han utilizado para llevar a cabo la aplicación.

Índex

Índex de figures

Índex de taules

Glossari de termes

M-learning	Aprenentatge electrònic via mòbil
SDK	Software Development Kid
TFG	Treball Final de Grau
Merge	Se li diu a la unió de dues branques
Back-End	Part lògica i interna de la aplicació.
Front-End	Part visual que veu l'usuari
RAM	Memòria d'emmagatzemat a curt termini per guardar dades temporalment
CPU	Unitat Central de Processament
GUI	Interfície d'usuari gràfica
Tree B +	Tipus d'estructura de dades en arbre
FK	Clau forana (Foreign Key)

1. Objecte del projecte

Es tracta de desenvolupar una aplicació mòbil educativa per aprendre a programar. El "target" són usuaris que no tenen cap noció de programació i que volen aprendre HTML5, llenguatge de marques que té les seves normes i estructures i que serveix tant per definir l'estructura com el contingut d'una web. L'aplicació ha d'oferir una petita explicació de les parts teòriques, sempre que sigui possible amb animacions que ajudin a comprendre el funcionament de què s'està explicant, una llista d'exemples i altra d'exercicis per fer. Sobretot, fent ús de recursos visuals juntament amb animacions per complementar les explicacions. El contingut estarà distribuït en cursos/temaris ordenats per dificultat on a cada un hi haurà una sèrie de nivells o subapartats. A cada nivell de cada tema es realitzen diversos exercicis que decidirà si passa al següent nivell o curs. Finalment, en acabar tots els cursos l'aplicació dota a l'usuari d'un certificat en HTML5.

2. Context, antecedents i necessitats d'informació

2.1. Context

El món està en desenvolupament constant i a un ritme de creixement tecnològic exponencial que cada cop va més de pressa. La tecnologia està cada dia més present i influent a la vida de les persones a causa de la utilitat que proporcionen i és normal que a mesura que avança el temps, els joves vulguin i necessitin més coneixements tecnològics.

En aquest sentit, la informàtica o més específicament la programació, està vista pel món com una cosa per experts i professionals i que només es pot aprendre qui es dedica explícitament això i no com una eina extra o un simple divertiment d'aprenentatge. Quan en realitat hi ha un gamma de dificultat de llenguatges molt extensa, on a dintre de cada llenguatge hi ha diferents nivells de coneixement. En el cas de l'HTML és un llenguatge amb conceptes bàsics molt senzills, però que pot arribar a ser complex quan es té un nivell avançat i es combina amb altres llenguatges. És per això que la gran majoria de persones interessades en aquest món comença amb llenguatges d'aquest caire.

Actualment, l'aprenentatge en línia i l'autodidactisme s'està potenciant gràcies als forts avantatges respecte a les metodologies tradicionals, i quina millor manera d'arribar a la joventut que a través d'aplicacions mòbils, que és el dispositiu personal que més hores es consumeix.

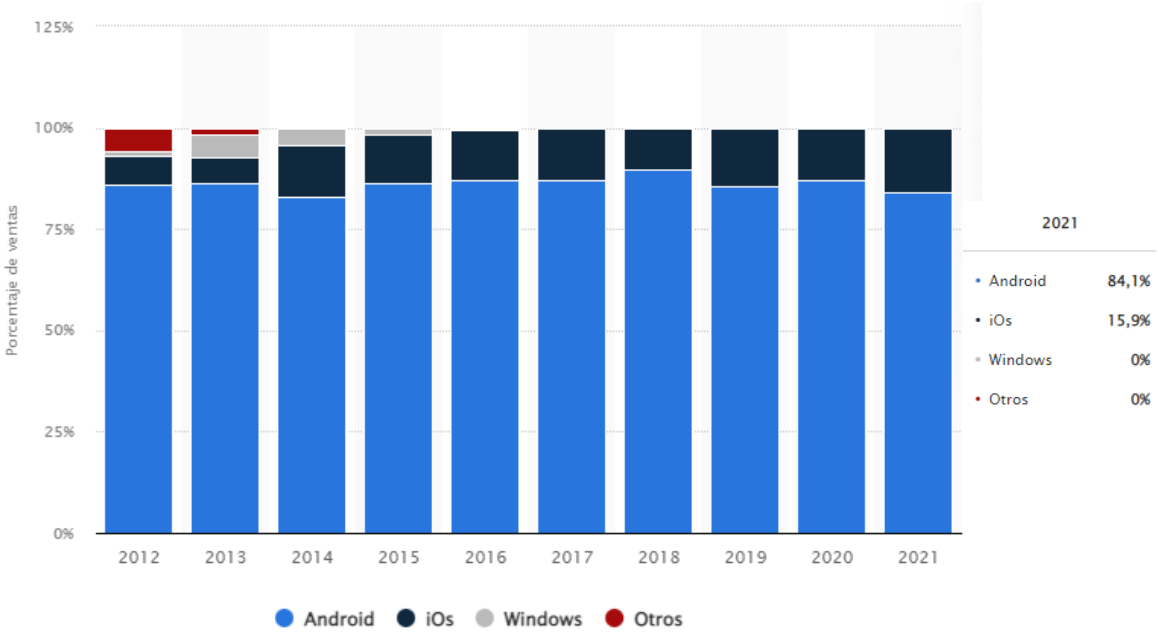
El M-learning presenta una sèrie d'avantatges que cada cop fa decantar més la balança cap al seu costat, aquestes són: l'alta flexibilitat d'ús que presenta, ja que es pot aprendre quan i a on es desitgi i el temps que es vulgui, inclús sense connexió a internet. Presenta una reducció del cost respecte a l'ensenyança tradicional molt elevada, només requereix un dispositiu mòbil que gairebé qualsevol persona del primer món disposa i el preu, si el té, del curs o l'aplicació. Evitant així el cost de qualsevol desplaçament o material físic. Segons un article que va treure la revista Forbes que parla sobre l'enfocament de la capacitat d'aprenentatge dels empleats confirma que l'Institut d'Investigació d'Amèrica (INC) conclou en què l'aprenentatge electrònic augmenta les taxes de retenció entre un 25% i 60% en comparació amb les taxes de vuit i déu per cent de les tradicionals, vist que els empleats

o alumnes tenen més control sobre el procés d'aprenentatge i poden revisar els cops que sigui necessari adaptant-se així al ritme de cada usuari. El fet que es tingui més retenció provoca, per tant, menys temps d'aprenentatge i dedicació. Per acabar, el fet de fer-ho remotament fa que aquest mètode sigui amigable amb el medi ambient.

Un punt molt important que té i ha tingut molt impacte en el M-learning és la recent pandèmia que hi ha hagut a escala mundial, que ha provocat que milions de persones es passin al bàndol digital.

Un altre motiu de pes és l'accessibilitat a dispositius mòbils per part dels joves i la seva adquisició a primerenca edat. Això també provoca que cada cop hi hagi gent més petita utilitzant el mòbil.

Existeixen diferents tipus de mòbils, però els que interessen en aquest cas i que qualsevol persona corrent té són els Smartphones o telèfons intel·ligents que són els més moderns, amb connexió a internet i pantalla tàctil. Dintre d'aquesta categoria es diferencien entre ells pel sistema operatiu que porten, tot i que predominen dos grans grups: Android (Google) i IOS (Apple). Tal com es veu a la imatge "I Fig. 2.1.1" s'observa una gràfica feta a través de les dades recollides de la consultora de tecnologia IDC (International Data Corportion).



I Fig. 2.1.1 Gràfic del percentatge de vendes global segons el sistema operatiu, Statista 2021

Cada un té les seves característiques, però no es pot dir quin és millor que l'altre, passa a ser més un gust de preferències. Pel cas del projecte s'ha escollit el que ofereix més facilitats com desenvolupador, a la Taula 2.1.1 es pot veure.

	Android	**IOS**
Desenvolupador	Google	Apple, Inc.
Sistema operatiu	Linux	OS X, UNIX
Entorn de desenvolupament	Android Studio	XCode
Penetració en el mercat	España (85%) USA (51%)	España (15%) USA (49%)
Costos en recursos per el desenvolupament	Llicència per públic l'app a Google Play	Quota de desenvolupador més recursos MAC.
Terminals	Molta variabilitat	Un sol fabricant
Codi	Plataforma de codi obert	Plataforma de codi tancat
Seguretat	Més fàcil de trobar vulnerabilitats al ser codi obert	Difícil trobar vulnerabilitats
Llenguatge	Java o Kotlin	Objective-C

I *Taula 2.1.1 Comparativa Android i IOS*

Si s'observen els avantatges i desavantatges de cada sistema operatiu s'arriba a la conclusió que el millor per aquest projecte és en Android per la gran variabilitat de terminals, el llenguatge Java i el codi obert, a més a més es pot fer sempre de manera gratuïta i amb iOS no.

L'HTML (HyperText Markup Language) és un llenguatge de baix nivell i és el codi base de qualsevol interfície web a internet. Aquest utilitza un sistema de marques per etiquetar diferents tipus d'elements, com textos, imatges, contenidors d'elements entre d'altres el qual permet formar una estructura en forma d'arbre on el fill es mostrarà dintre del pare. Normalment, aquest es combina amb altres tipus de llenguatge com CSS que aporta estils o JavaScript que aporta funcionalitat.

2.2. Antecedents

Per entendre millor com funciona el món de les apps educatives i quina estructura de disseny i interfície triomfa més, s'ha de cercar els antecedents que hi ha i com són les apps que més han destacat en aquest sector.

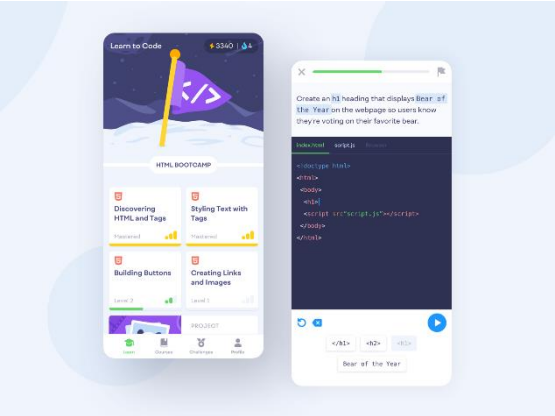
Una de les que té millor valoració dintre de Google Play és "**Mimo Code**" que ofereix diferents tipus de llenguatge. És un molt bon exemple del que es vol dur a terme vist que ofereix un mecanisme molt interactiu amb dissenys molt colorits i un sistema d'aprenentatge senzill per interioritzar els processos de codificació. Està enfocat per a nens i joves, i parteixen d'una base elemental fins a un nivell mitjà-baix. La idea d'enfocar l'aprenentatge com un joc es pot veure reflectit en aquesta app, ja que fa ús d'un sistema de puntuació que permet modificar el personatge i la interfície en si, que es pot aconseguir fent exercicis. És una manera de motivar als més petits a continuar aprenent.

Aquesta app ha ajudat a desenvolupar i donar idees de la interfície o Front-End, a més a més de tenir una estructura basada en cursos i nivells força similar.

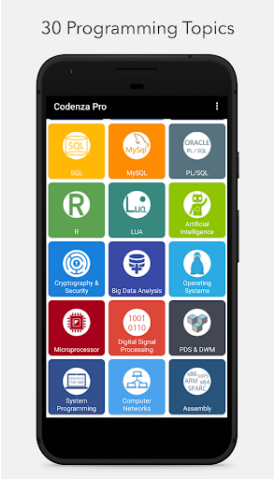
D'aquesta app també s'ha extret la idea d'enfocar els exercicis com un joc per tal de motivar més als usuaris i tinguin ganes de millorar. Alhora el sistema de vides que hi ha muntat permet limitar les hores d'ús.

En segon lloc, està l'app "**Codenza**" que és per nivells més avançats i està dotada d'una gran varietat d'exercicis, proves, llibres, vídeos i projectes. És una app més enfocada per posar a prova les habilitats sobre el llenguatge escollit. D'aquesta s'ha tret profit de la gran biblioteca de recursos que conté juntament amb la quantitat de proves i documentació sobre el llenguatge.

Una idea interessant que es pot intentar a aplicar a l'app que no està present en la gran majoria d'antecedents i que la pot fer destacar per sobre de les altres és incorporar un editor HTML integrat en temps real per tal de portar la pràctica a un nivell més, d'aquesta manera els estudiants podran veure en temps real si el que estan fent és el que se'ls demana o necessiten.



II Fig. 2.2.1 Pantalla inicial i d'exercici de l'app Mimo Code, Dribble 2021



III Fig. 2.2.2 Pantalla inici de l'app Codenza, Google Play 2022

2.3. Necessitats d'informació

Per dur a terme el treball es necessita certa documentació i aprenentatge sobre el funcionament de les diferents plataformes que es fa ús.

Primerament, s'ha de conèixer l'entorn Android Studio i la seva programació a través de Java i integració a través de Github. Tota la part que té a veure amb el codi es pot consultar a través de la web Android Developers que és la principal font de coneixement del codi de Android Studio, igual que la part d'integració.

En segon lloc, es necessita la informació per tractar amb el Back-End i base de dades de MongoDB que es pot fer a través de la documentació molt extensa que disposen a la web. Dintre d'aquest bloc s'inclou des de la inserció de les dades i creació dels models de dades fins a la integració i sincronització amb les aplicacions client.

Es necessita conèixer el funcionament del codi SDK per configurar Realm (base de dades local) en mode sincronització.

Es necessita un nivell de HTML avançat per tal de poder crear els temes, els nivells i els exercicis.

3. Objectius i abast

3.1. Objectius principals

A nivell acadèmic:

- Desenvolupar una app educativa amb Java per aprendre a programar pel sistema operatiu Android.
- Desenvolupar un producte utilitzant metodologies de desenvolupament de software agile.
- Integrar el projecte amb la plataforma Github.
- Implementar activadors automàtics davant d'events a la base de dades.
- Descobrir i implementar plugins nous útils per a l'app.
- Conèixer els límits de l'Android Studio i les aplicacions mòbils.

A nivell de producte:

- Crear un entorn d'aprenentatge personalitzat adaptat als coneixements de l'usuari, partint que no té un alt nivell en programació HTML5.
- Desenvolupar una interfície d'usuari atractiva pel client objectiu.
- Fer que els usuaris paguin pels avantatges de l'app (dependrà del model de negoci).
- Crear una estructura d'aplicació basada en cursos (de menys a més difícil) amb nivells/temaris dintre de cada curs.
- Mantenir els usuaris actius a l'app. Es comprovarà utilitzant "MAU" usuaris actius de forma mensual, KPI.
- Aconseguir el màxim de descàrregues al Google Play.
- Obtenir bones valoracions i ressenyes al Google Play.
- Garantir la seguretat de les dades dintre de l'app.
- Desenvolupar explicacions teòriques i visuals del temari.
- Dotar d'un certificat en coneixement HTML5 en acabar tots els cursos.
- Dotar l'app de funcionament offline.
- Poder variar l'idioma de l'aplicació.
- Integrar l'app amb una base de dades local que es sincronitzi amb una remota.

3.2. Objectius secundaris

- Dotar l'app de funcionament offline.
- Crear l'app en versió per iOS.
- Implementar un editor HTML en temps real dintre l'aplicació
- Incorporar l'aprenentatge de diferents tipus de llenguatge.

3.3. Públic Target

El públic objectiu d'aplicacions enfocades a l'ensenyament de tecnologies modernes, com és el cas de la programació, són normalment gent jove amb interessos d'aprenentatge informàtic.

En tractar-se de llenguatge HTML també interessa a persones emprenedores que vulguin crear la seva pròpia web i necessitin formació bàsica sobre el llenguatge.

3.4. Abast

Per definir l'abast de l'app s'ha de tenir en compte els objectius i qüestionar diversos punts per marcar els límits de l'app.

Si es comença per la part del Back-End el primer que s'ha de veure és la capacitat que haurà de tenir el servidor per poder gestionar tota la informació. Es pot dir que l'app serà capaç de suportar entre cinquanta i cent usuaris amb capacitat d'ampliació.

En segon punt, l'aplicació que té cadascú dels clients, quina pila de pantalles pot tenir, és a dir, fins a quantes pantalles és capaç l'aplicació de guardar en temps d'execució. Es determina un màxim de cinc pantalles de profunditat. En Android es parla d'activitats tot i que cada un es pot fragmentar en diverses pantalles.

S'ha de conèixer el pressupost que es disposa, especificat en l'estudi de viabilitat del projecte, juntament amb la manera que l'aplicació obtindrà benefici escrit a l'apartat de model de negoci.

S'ha d'especificar el temps que es disposa, en aquest cas uns quatre mesos de desenvolupament.

L'aplicació funciona en un principi només en el sistema operatiu d'Android, tot i que no es descarta una possible ampliació per a IOS.

El fet que es vulguin aplicar diferents idiomes fa que es multipliquin les dades. La idea inicial de l'app és oferir tres tipus d'idioma: català, castellà i anglès.

S'ha de determinar el tipus de registre per si s'ha d'integrar amb alguna plataforma externa d'autenticació. Inicialment, es fa a través de correu i contrasenya.

Un punt que marca molt la dimensió és si és una app sincronitzada amb una pàgina web, o només existeix en format mòbil. De moment es llençarà només en format d'aplicació mòbil.

S'ha de determinar quin sistema d'emmagatzematge de dades s'utilitza, si es guarda tot en local o remotament alliberant càrrega d'espai al dispositiu. Per a l'aplicació s'usa base de dades remota de MongoDB Atlas i App Services per integrar-ho.

Requereix connectivitat per sincronitzar les dades, però es podrà treballar sense connexió temporalment.

4. Metodologia

4.1. Anàlisis de metodologies de desenvolupament de software

Pel desenvolupament del software existeixen diferents metodologies, les més adequades per aquest projecte són: Cascada (waterfall), Incremental, Espiral o una àgil. Per escollir una de les quatre s'ha fet cerca d'informació i investigació per determinar la més adequada al projecte.

La primera d'elles és en cascada o "waterfall", que té aquest nom a causa de l'organització del treball i el seu flux que és totalment vertical i amb una execució de les dades seqüencial. És un mètode que treballa sobre segur estalvia temps. Les fases principals són: anàlisis de requisits, disseny del sistema, disseny de programa, modificacions, "testing", codificació i manteniment. És poc flexible al canvi a causa del funcionament en seqüència i que no treballa amb retroalimentació entre les diferents fases. Tot el procés està fixat amb dates d'entrega i pressupostos, però un error en qualsevol part del procés pot endarrerir molt les dates.

En segon lloc, l'incremental, que com diu el seu nom treballa amb etapes incrementals on a cada una s'afegeix una nova funcionalitat. Cada una d'aquestes etapes es caracteritza en ser curtes i aconseguir objectius específics per tal d'observar millores en el producte final cada cop que finalitza una. El fet de millorar constantment el producte final provoca una participació constant per part del client. És necessari definir prioritats dintre dels requeriments per escollir quins requeriments complir a cada etapa. Aquesta metodologia permet molta flexibilitat i es pot començar a utilitzar el producte prèviament a acabar-lo. També és la base de moltes metodologies àgils.

En tercer lloc, està en espiral, que venen a ser quatre fases: planificació, anàlisis de risc, enginyeria (codificació, "testing" i desplegament) i avaluació del client, però aquest cop es processen en espiral. Es comença de més enfora i es va anant cap a dins, com més a dintre més avançat està el projecte. L'usuari és participat en tots els processos de cicle de vida del

disseny. És molt útil sobretot amb projectes de llarga durada i normalment requereixen un prototip. Com a punt negatiu és molt poc flexible i s'ha de seguir estrictament el procés.

Aquestes tres metodologies formen part del grup de les tradicionals i són les que porten més temps existint. Tot i això, les empreses més grans del món estan apostant des de fa pocs anys per noves metodologies de desenvolupament de software basades en "agile", que permeten una alta flexibilitat, agilitat i entregues a curt termini funcionals. D'alguna manera aquestes tenen una base de la metodologia incremental, en què a cada cicle del desenvolupament s'agreguen noves funcionalitats a l'aplicació final. Aquestes fases són cicles curts i ràpids en els quals s'afegeixen canvis petits. Les principals són: Scrum, KanBan i la programació extrema.

Scrum itera sobre blocs de temps curts i fixes per aconseguir un resultat complert a cada iteració. Cada una d'aquestes iteracions es diuen "Sprints" i consta de quatre parts: la planificació de la iteració (Planning sprint), la execució, la reunió diària (daily Meeting) i la demostració (sprint review). Gràcies a les reunions i revisions diàries es possible portar les entregues i el seguiment al dia exacte. Per iniciar aquesta metodologia es necessari crear un "backlog" (llista prioritzada de funcionalitats que ha de tenir el producte) com a magatzem de futurs "sprints". Al final de cada un dels sprints s'escullen les funcionalitats a lliurar pel pròxim.

El següent és KanBan, aquest divideix les tasques en petites porcions per organitzar-les en una taula de treball dividit en les següents columnes: "To Do", "In Curse", "Done". Amb aquestes columnes s'aconsegueix crear un flux de treball molt visual basat en prioritats que va incrementant el valor del producte.

Per últim, està la programació extrema que es caracteritza pel feedback constant per part del client, la integració continua, programació en petits grups de treball i entregues setmanals. Està orientat sobretot a projectes de curta durada.

4.2. Tria i conclusió de la metodologia

L'inconvenient que es té amb les metodologies àgils amb aquest projecte és que tenen una gran utilitat, en part, si es treballa en grups o equips de persones reduïts, ja que busca comunicacions setmanals, equips motivats amb bon ambient de treball i reunions setmanals per posar en comú el progrés de la feina, i això no es pot dur a terme en un TFG individual.

D'altra banda, aporten molta més flexibilitat, bona resposta al canvi i s'observa el progrés de l'aplicació al final de cada "sprint", a part de portar les entregues sempre al dia i amb desviacions de temps molt petites.

A l'hora de fer la planificació de les tasques resulta molt més adaptable que les tradicionals vist que aquestes tenen un procés més controlat, amb normes i entregues fixes que a l'hora de desenvolupar costaran de complir.

L'aspecte de relació amb el client és molt més bona opció les àgils també. El fet que hi hagi un feedback constant amb el client fa que quedi una aplicació molt més gustosa i el programador tingui més facilitat a l'hora de fer-ho.

IV Fig. 4.2.1 Gràfic del procés de la metodologia tradicional i agile. STIC Noticias, 2018

En conclusió la metodologia que s'utilitza és la "agile" amb una base incremental, però sense haver de fer els "daily meetings" ni reunions de grup, ja que és un projecte individual. Però si amb les revisions de sprint cada dues setmanes. En aquest cas no s'escull una en concret sinó que s'agafa un part de cadascuna per adaptar-se millor al projecte. A la part de planificació s'explica com es fa.

5. Definició de requeriments funcionals i tecnològics

Requeriments funcionals:

- Permetre funcionar en qualsevol tipus de Smartphone Android.
- S'ha de poder descarregar a Google Play.
- Controlar l'accés i permetre'l als usuaris que s'identifiquin (login).
- Permetre registrar una nova compta associada a un correu electrònic.
- Permetre autenticar-se amb Google o Facebook.
- Verificar el correu electrònic en crear el compte.
- Permetre tancar la sessió
- Tancar sessió en tancar l'aplicació després d'un temps específic.
- Permetre actualitzar les dades del perfil de l'usuari o canviar el nom d'usuari/contrasenya.
- Permetre activar/desactivar la música de fons.
- Permetre activar/desactivar el so d'animacions.
- Permetre eliminar el compte mitjançant una confirmació amb el correu associat.
- Mostrar l'aplicació horitzontalment (responsive)
- Mostrar el curs pel qual va l'usuari.
- Mostra una barra del progrés del curs.
- Mostrar el nivell dintre del curs pel qual va el usuari.
- Mostrar la mitja global de qualificació de les proves realitzades fins al moment.
- Mostrar el llistat dels cursos disponibles i els no disponibles
- Mostrar una explicació teòrica dintre de cada nivell.
- Realitzar prova al final de cada curs.
- Dotar d'exercicis cada curs.
- Permetre entrar als cursos ja realitzats.
- Denegar l'accés als cursos més avançats del que està cursant.
- Permetre l'accés al següent curs un cop s'han superat els nivells del curs anterior.
- Permetre tenir amics agregats.
- Permetre utilitzar l'app sense connexió.

- Permetre realitzar pagaments per aconseguir l'app amb avantatges.
- Permetre integrar anuncis amb l'app gratuïta.

Requeriments tecnològics:

- Treballar amb un portàtil amb capacitat de processament per desenvolupar una app.
- Fes ús d'una base de dades remota/clúster servidor.
- Treballar amb l'entorn Android Studio i llenguatge Java.
- Integrar l'app amb Github per usar la board (KanBan) i controlar les iteracions dels sprints.
- Utilitzar una eina per a la creació del BackLog del producte.

6. Desenvolupament

Es realitza el desenvolupament de codi mitjançant iteracions de funcionalitats. La idea dels primers sprints és crear i implementar les bases del que serà l'app: la creació de la base de dades, integració amb l'app, la estructura de pantalles i "stack", dissenys de les pantalles principals, inici de sessió, etc. Els darrers sprints són per anar implementant funcionalitats addicionals que no són tan prioritàries a l'aplicació com el so, animacions entre pantalles, opcions de personalització, etc.

6.1. Creació de l'app i integració amb Github

Per desenvolupar l'aplicació mòbil s'utilitza l'entorn integrat oficial de la plataforma Android, Android Studio. Tot i que fins llavors s'havia fet servir Eclipse, els professionals i el mateix sistema operatiu recomanen aquest entorn. A més a més integra diferents emuladors de mòbil per testejar l'aplicació.

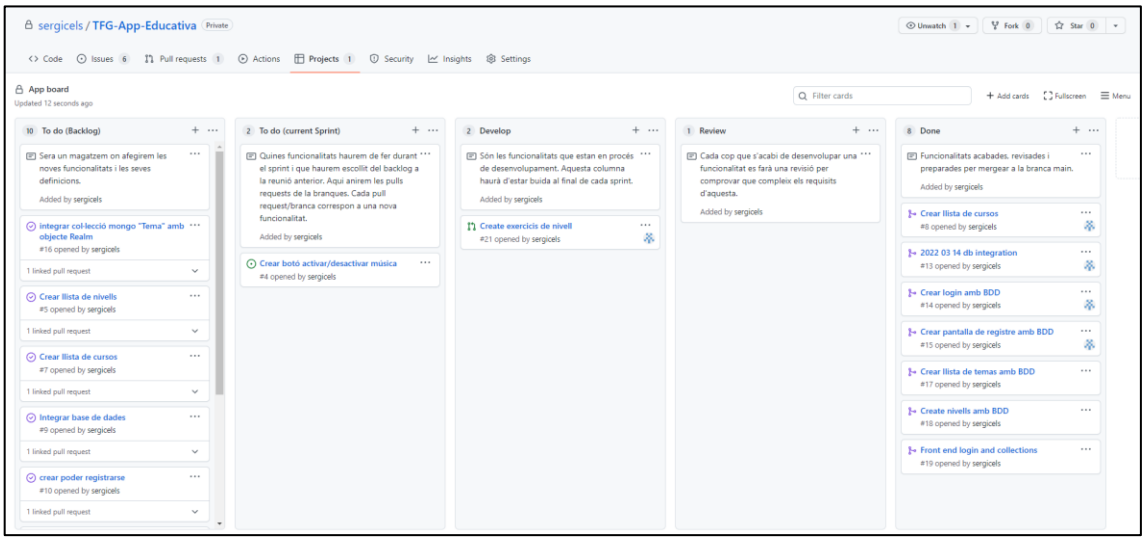
Es crea un projecte/app que pot estar en llenguatge Kotlin, o com es el cas, en Java que està penjat i connectat a un repositori de Github privat que integra la pissarra KanBan on es mostra el BackLog, i les funcionalitats que estan desenvolupades, en procés o que encara no s'ha realitzat amb les seves respectives notes, prioritats i dates.
Per a cada una de les funcionalitats noves es crea una branca nova sobre la que treballar, la qual serà una "pull request" dintre de la pissarra del Github.

Quan aquesta funcionalitat està finalitzada i funcional al cent per cent s'integra a la branca main (sempre que no hi hagi conflictes) per poder actualitzar les altres branques fent "pull". D'aquesta manera queda el projecte penjat al núvol, ben organitzat, estructurat i amb un sistema de branques que fa més flexible els canvis, correcció d'errors i possibilitat de treballar en paral·lel amb diferents funcionalitats. Així mateix, veure de manera global el progrés de l'app i el compliment dels diferents sprints.

El projecte està estructurat amb un sistema de carpetes per defecte, tot i que es poden crear subdirectoris dintre de cada un però no aglomerar tots els fitxers i sigui més llegible. A l'apartat de funcionament de l'aplicació es defineix com s'han estructurat els fitxers i la funció de cada un d'ells.

Tal com es veu a la Fig. 6.1.1 hi ha la pissarra KanBan del repositori on està penjada l'aplicació.

Està dividida en diferents columnes: " *To do (Backlog)* " on hi consten totes les funcionalitats a integrar a l'aplicació que donen resposta a tots els requeriments. S'hi poden afegir de noves durant el transcurs del projecte però complint les entregues dels sprints. " *To do (current Sprint)* " on estan les funcionalitats que han d'estar fetes quan finalitzi el sprint en que es trobi, " *Develop* " on figuren les funcionalitats que estan en fase de desenvolupament, " *Review* " són les funcionalitats que estan acabades però pendents de verificació (s'han de testejar), i un cop passen a " *Done* " significa que ja estan llestes per *mergear* la funcionalitat/branca a la main. Per dir-ho d'alguna manera cada branca ve a ser com l'aplicació que està en preproducció d'una nova funcionalitat que es pot afegir al producte en producció quan supera tota aquesta cadena de columnes.



V Fig. 6.1.1 Pissarra KanBan Github

6.2. MongoDB Atlas

Per a la base de dades remota de l'app s'utilitza MongoDB, un sistema de base dades NoSQL que està orientat a documents i és de codi obert. Algunes de les característiques d'aquest sistema són: emmagatzema dades en documents flexibles similars a JSON (BSON)on cada un d'ells s'assigna a un objecte del codi de l'aplicació, és una base de dades distribuïda en el nucli que aporta alta disponibilitat i escalabilitat horitzontal, és

d'ús gratuït, però limitat i suporta diferents llenguatges de programació com C, C++, Java, Python, PHP i JS.

Per aquest projecte en concret s'utilitzen els serveis MongoDB Atlas i MongoDB App Services que tenen una configuració adaptada a Realm i Android Studio de forma gratuïta. A més a més MongoDB disposa d'una precisa i densa documentació a la pàgina web oficial amb tota mena de guies, explicacions i configuracions específiques.

MongoDB Atlas és una potent plataforma que pot complir amb tots els requisits que tindria un Back-End però basat en un servidor en clúster. Aquest tipus de servidor és la unió de varis sistemes informàtics que funcionen com si fos un sol. Això significa que comparteixen recursos hardware i software amb els fins d'oferir més velocitat i alta disponibilitat. Existeixen dos tipus de clústers depenent de l'arquitectura: en núvol i en servidor dedicat.

En núvol proporciona serveis de clustering que incorporen balanceig de càrrega i replicació basat en núvol i tenen una implementació més ràpida. En canvi, ofereix menys personalització per adaptar-ho a l'aplicació específica.

En servidor dedicat, al ser propi, és totalment adaptable i personalitzable a les necessitats, tot i que això comporta més volum de càrrega, per tant, menys rapidesa i lleugeresa.

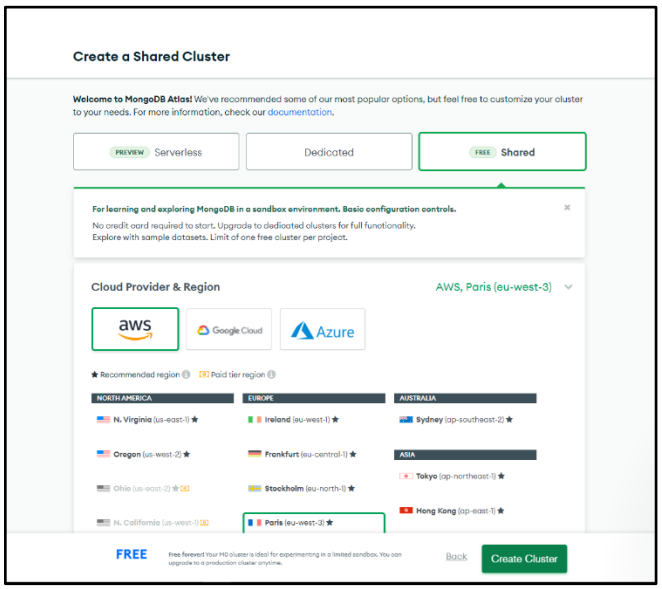
MongoDB Atlas ofereix els dos tipus de clústers i per a cada tipus diferents gammes. Per a l'aplicació s'ha escollit un clúster en cloud compartit gratuït amb un màxim de 512MB d'emmagatzematge, amb RAM i CPU compartida. Aquest tipus de clústers amb capacitats tan petites s'utilitzen normalment amb aplicacions que estan en preproducció o s'està començant a construir, com és el cas de l'app. Tot i ser de franc aquest clúster ofereix monitorització i alertes a través de disparadors, eines de diagnòstic i optimització del rendiment (real-time statistics), autentificació (ip access-lists, user athentication and autorization, roles), encriptació End-to-End TLS 1.2+, encriptació a nivell de dades de documents al sector del client i integracions API.

Per poder contractar un clúster amb MongoDB és necessari crear un compte a través de la pàgina web o registrar-se com a usuari de Google. Seguit d'això es contracta el clúster i s'especifica la seva configuració, en el cas del projecte quedaria de la següent manera: servidor compartit, 512MB d'emmagatzematge (gratis) amb opció d'ampliació, el proveïdor

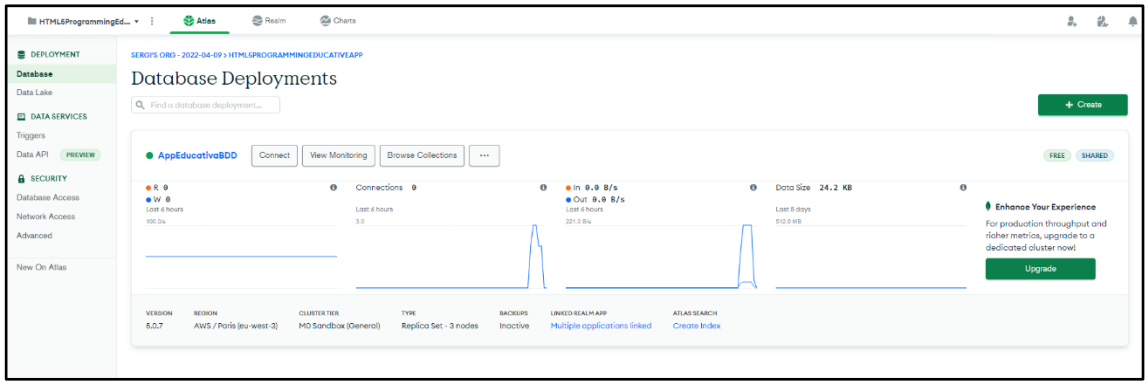
cloud serà AWS (Amazon Web Services) que és una col·lecció de serveis de computació al núvol i s'ha escollit la regió de Paris que és la més recomanable.

En tercer lloc es configura un "ip access-list" on es determina a través de la Ip, quins dispositius poden connectar-se al clúster. Per poder accedir, s'ha introduït a aquesta llista la Ip del ordinador on s'utilitza l'Android Studio amb l'emulador.

En darrer lloc, es crea un usuari de base de dades amb el que s'accedeix al clúster. Serveix per afegir una capa més de seguretat a les dades, d'aquesta manera també es té un registre de l'activitat al clúster i les últimes modificacions.



VI Fig. 6.2.1 Creació i configuració del clúster de MongoDB. Pròpia, 2022

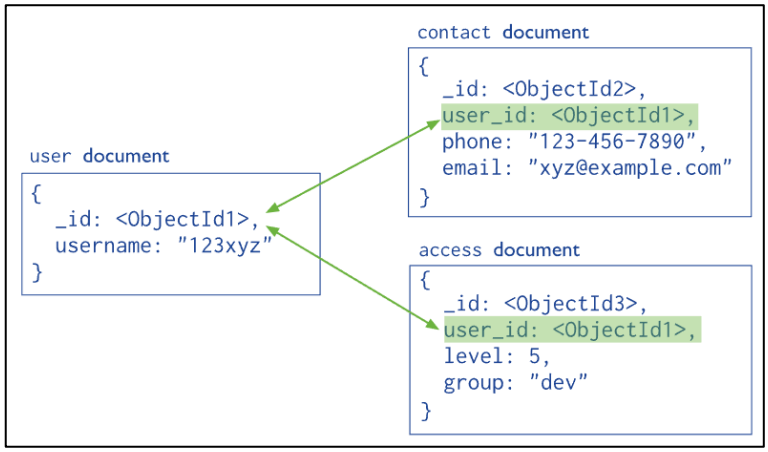


VII Fig. 6.2.2 Panell del back-end del clúster de MongoDB. Pròpia, 2022

6.2.1. Base de dades, documents i col·leccions

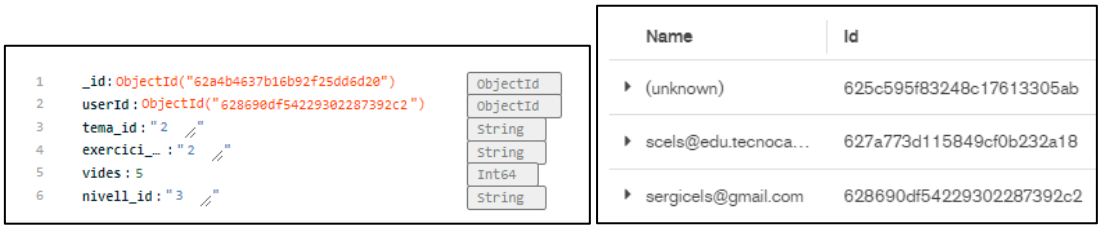
MongoDB Atlas guarda els registres de dades com a documents BSON (key-value) que estan emmagatzemats dintre de col·leccions. Les col·leccions són tractades com si fossin taules en el llenguatge SQL en el que cada una correspon a un objecte. Cada base de dades conté una o més col·leccions.

Per a cada registre key-value d'un document s'ha d'especificar quin tipus de BSON és, que poden ser els més elementals com int, boolean, String o fer referència a altres documents o matrius de documents per relacionar-los.



VIII Fig. 6.2.1.1 Relació de camps de documents. MongoDB, 2020

A l'aplicació s'utilitzen variables referencials per accedir a documents únics d'usuaris com per exemple el document "*Result*" que és únic per usuari i guarda el punt exacte on l'usuari es trobava i les vides que té de cada un d'ells a través de l'identificador.



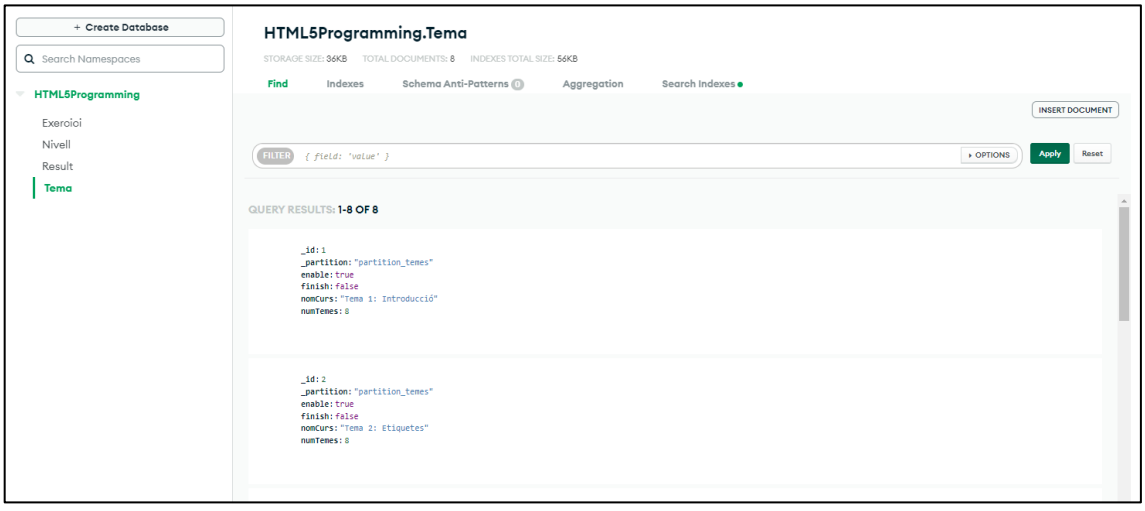
IX Fig. 6.2.1.2 Relació de un usuari amb un document. Pròpia, 2022

Els noms de camp (key) han de ser cadenes i han de complir unes restriccions: sempre ha d'haver-hi un camp "_id" que actua d'identificador o clau principal del document (únic en tota la col·lecció i de tipus int, String, ObjectId o UUID), cap valor pot ser null i han d'estar tots els camps ordenats. No és necessari que els documents d'una mateixa col·lecció tinguin els mateixos camps i el tipus de dada d'un camp pot ser diferent del mateix camp d'un altre document.

Cada operació que es realitza sobre un document és atòmica, inclús si aquesta modifica diferents documents incrustats en un. Si l'operació modifica diferents documents independents, la modificació de cada un serà atòmica però l'operació en el seu conjunt no. Degut això, és possible que s'intercalin les operacions.

La mida màxima de cada document es de setze megabytes per tal de garantir un correcte funcionament de la RAM i que un sol document no utilitzi gran part de l'amplada de banda.

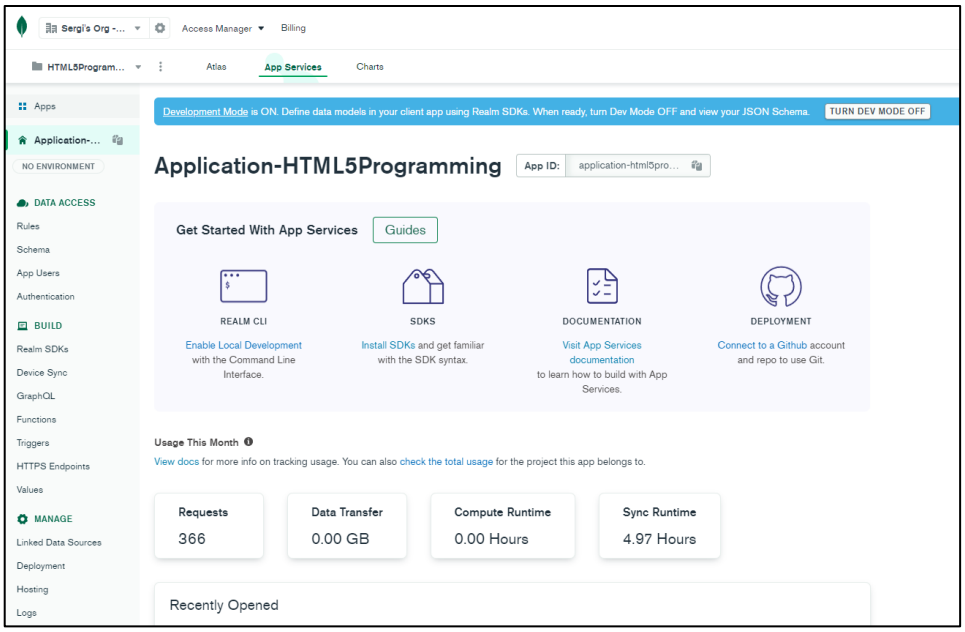
A la Fig. 6.2.1.3 s'observen quatre col·leccions: "Exercici", "Nivell", "Result", "Tema" dintre de la base de dades HTML5 Programming. Dins de cada col·lecció estan els diferents documents de la col·lecció, és a dir, dintre de Tema hi ha x documents on cada un d'ells correspon a un tema de la matèria. Aquesta col·lecció es pot dir que és de tipus estàtica, iguals i accessibles per a tots els usuaris. En canvi, la col·lecció "Result", és única per usuari com s'ha dit anteriorment.



X Fig. 6.2.1.3 Apartat de les col·leccions i dades dintre del clúster. Pròpia, 2022

Aquest clúster serà el Back-End de l'app la qual s'haurà de connectar/integrar amb l'aplicació local del client. Existeixen tres formes diferents de connectar el clúster: mitjançant una interfície interactiva JavaScript anomenat MongoDB Shell que s'instal·la al ordinador (poc pràctic), connectar a una aplicació Atlas App Services que s'integra i sincronitza amb RealmDatabase o per últim fent ús de l'eina Compass de Mongo que consisteix en una GUI que permet explorar, modificar i visualitzar les dades emmagatzemades.

Per a una app mòbil el més convenient, eficient, útil i que proporciona més avantatges, és fer-ho amb Atlas App Services. Que a més a més és una eina pròpia de la mateixa empresa que permet accedir directament des del panell del navegador on es configura el clúster.



XI Fig. 6.2.1.4 Interfície web de Mongo App Services. Pròpia, 2022

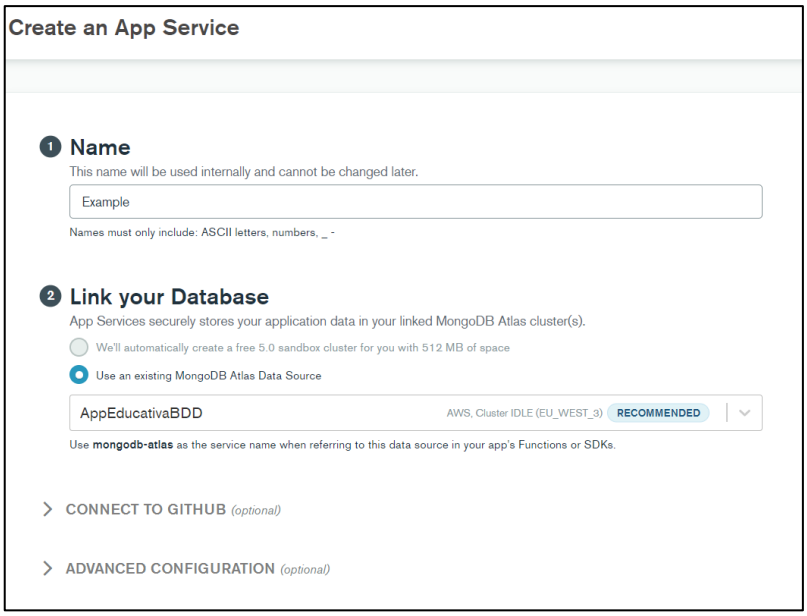
6.3. MongoDB App Services

Aquesta eina o extensió de Atlas es pot definir com un conjunt de serveis al núvol administrats, funcions i regles d'accés a les dades. Serveix per allotjar una aplicació completa a un entorn del núvol administrat i enllaçat amb el clúster Atlas. Tenen la característica de reaccionar davant de canvis a les dades del servidor automàticament. Es pot accedir directament a través del panell d'administrador de Mongo Atlas i només cal crear el App Service i enllaçar-lo amb el servidor.

Qualsevol infraestructura Back-End d'aplicacions requereix temps, diners i experiència per construir-lo, administrar-lo i mantenir-lo. En canvi, aquesta eina administra els usuaris sol permetent al desenvolupador centrar-se en les funcionalitats noves. També disposa de registres i còpies de seguretat incrementals.

Té una gran escalabilitat en treballar en núvol i permet pagar la quantitat exacta de computació que es fa servir, ja que els preus estan basats en l'ús.

Permet controlar tots els usuaris i els rols de cadascú, a més a més, gràcies a la lògica de validació de dades a través dels esquemes d'objectes la integritat de les dades queda totalment reforçada.



XII Fig. 6.3.1 Enllaç clúster Atlas amb App Services. Pròpia, 2022

6.3.1. Atlas Device Sync

Per aquest projecte un dels serveis més importants que s'utilitza és el servei sincronitzat Atlas Device Sync, que permet sincronitzar les dades emmagatzemades entre els diferents dispositius, sense necessitat de connexió immediata, amb el Back-End a través dels SDK de la base de dades local. Les aplicacions client utilitzen RealmDatabase per guardar les dades en local i quan el dispositiu disposa de connexió de xarxa, Sync es connecta al Back-End a través dels SDK de Realm per enviar les dades i actualitzar-les. De la mateixa manera els SDK de Realm rep els canvis del servidor i els integra en el domini local. Aquest mètode de

treball permet que en àmbit local sempre es pugui treballar i no s'hagi de preocupar per la connexió a la xarxa.

Aquest servei ofereix un sistema de resolució de conflictes per tal de manejar els diferents usuaris que puguin actualitzar les mateixes dades alhora i mantenir la lògica d'aquestes. Per fer-ho, es basa en unes regles de resolució de conflictes com per exemple que les eliminacions o actualitzacions sempre guanyen sobre altres canvis. De totes maneres es poden personalitzar aquestes regles al gust de l'administrador i aplicació.

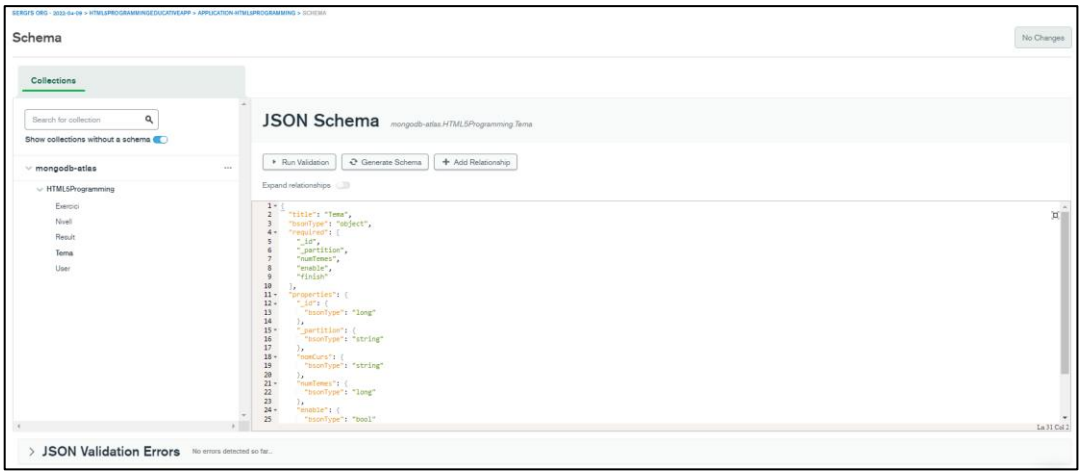
El servidor processa les peticions fora d'ordre, ja que el conjunt de canvis pot ser que no arribi ordenadament, i és per això que el servei manté un historial de transaccions per domini per poder ordenar els missatges, dit d'una altra manera, l'última escriptura és la que té prioritat.

Quan el dispositiu té connexió, l'app treballa amb objectes vius, és a dir, pot actualitzar i mostrar les últimes dades emmagatzemades de la base de dades remota mentre s'està utilitzant. Així pot actualitzar en temps real la interfície de l'usuari.

Per garantir la consistència de les dades aquest servei aplica el seu model de dades de dues maneres: un en format BSON que interpreta la banda del servidor/clúster i defineix les dades com a documents MongoDB per validar-los i sincronitzar-los i la banda de App Services que defineix les dades com objectes natius de l'aplicació del mòbil perquè sigui llegible per la base de dades local.

El model del costat dels serveis se'l descriu com esquema d'objectes, on es defineixen cada un dels camps, les propietats i les relacions amb l'objecte Realm en format JSON.

Es important entendre que es genera un esquema per a cada un dels objectes/col·lecció de l'aplicació perquè pugui llegir l'estructura del que s'envia i es rep. A la Fig. 6.3.1.1 es veu l'exemple de l'esquema de la col·lecció "Tema" el qual es defineix el nom de l'objecte, les propietats i tipus de dada que té i en aquest cas es marquen cinc que són obligatòries.



XIII Fig. 6.3.1.1 Esquema d'objectes d'App Services. Pròpia, 2022.

Existeixen diferents maneres de generar aquest esquema: es pot crear a partir de les dades que s'hagin introduït al clúster en format de documents i col·lecció, on el mateix servei integra una opció "Generate Schema" que crea aquest esquema a partir de les dades del clúster. L'avantatge d'això és que posteriorment a generar l'esquema, a l'apartat de SDK s'indica com ha d'estar escrit l'objecte en el codi i si hi ha algun error o falta per definir algun camp obligatori.

Com es veu a la Fig. 6.3.1.2 s'utilitza la classe abstracta RealmObject per declarar-lo com un objecte de la base de dades. Els objectes que implementen aquesta classe abstracta han de contenir un constructor buit per quan se'n creen de nous.

XIV Fig. 6.3.1.2 Model de codi d'objecte Realm. Pròpia, 2022.

L'altra opció és fer-ho en sentit invers, introduir tota la informació a través de mètodes SDK utilitzant els models d'objectes del domini i la funció *createObject(class)*. Normalment, s'utilitza aquesta manera quan ja es disposa de la informació i les dades en local i es volen guardar remotament en el servidor.

En el cas de l'aplicació educativa, s'ha introduït tota la informació a través del clúster i amb l'eina del App Services s'han creat els esquemes excepte els documents privats d'usuari, que aquests es creen cada cop que un usuari nou es registra.

Si es volen fer canvis a l'esquema, ja sigui a través de l'aplicació client o a través del clúster s'ha d'anar amb compte, ja que pot causar una ruptura a l'altre model. L'esquema accepta canvis d'una certa dimensió, si es volgués fer canvis majors o reestructurar l'esquema s'hauria de fer una migració al dispositiu o eliminar la base de dades local per poder tornar a importar els esquemes.

Per activar la sincronització s'ha de seleccionar la base de dades que es vol connectar, activar el mode de desenvolupador per poder testejar i editar (en petita mesura) l'esquema en qualsevol moment i escollir un dels dos mètodes de sincronització. Aquest mètode determina la manera en què els usuaris poden accedir a les dades de MongoDB a través de App Services utilitzant els SDK de Realm. Existeixen aquests dos tipus:

- Sincronització basada en partició
- Sincronització flexible

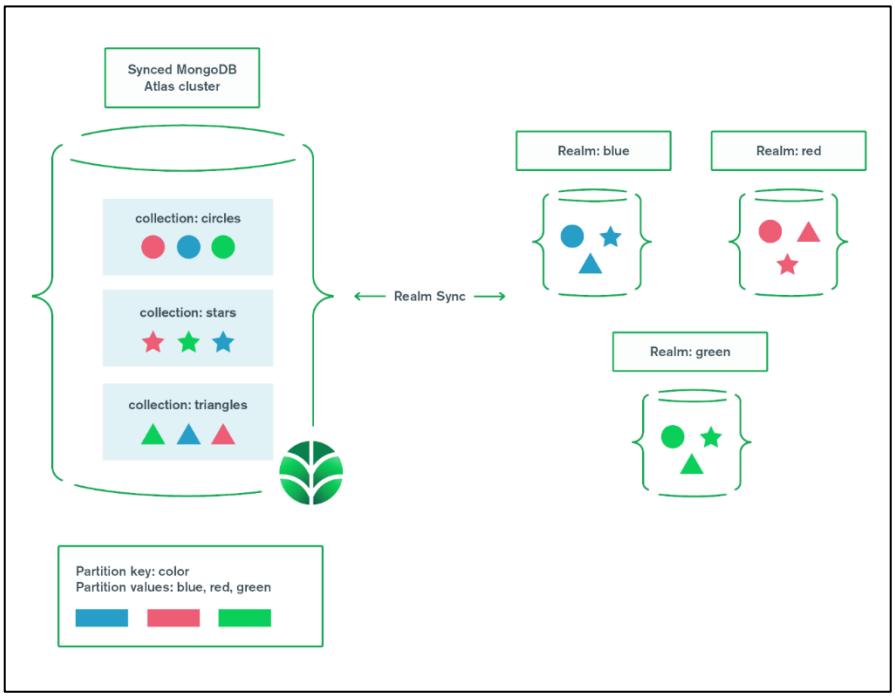
No es poden fer servir les dues simultàniament i per canviar d'opció s'ha de finalitzar i tornar a habilitar la sincronització causant un restabliment del client.

6.3.1.1. Sincronització basada en partició

Les particions són l'abstracció entre el model d'objectes Realm "*RealmObject*" i el model de documents del server Atlas anomenats col·leccions. El clúster Atlas treballa amb diferents servidors remots que proporcionen emmagatzematge per a les dades sincronitzades. Cada una d'aquestes col·leccions s'assigna a un tipus d'objecte Realm diferent de l'app. L'app conté el que es diu un Realm sincronitzat que és un arxiu local que conté cada dispositiu. Aquest arxiu pot contenir alguns o tots els objectes importants per l'usuari. Podria ser que una aplicació client utilitzés més d'un Realm sincronitzat per accedir a diferents objectes que necessita.

La funció d'aquestes particions llavors és enllaçar els objectes Realm amb les col·leccions de la següent manera: un dels paràmetres del arxiu sincronitzat és un valor de partició llavors quan l'aplicació crea objectes en el Realm sincronitzat i es sincronitzen amb el clúster el valor de la partició es converteix en un camp en les col·leccions. Això vol dir que tots els documents que pertanyen a una mateixa partició tenen els mateixos permisos de lectura/escriptura per a un usuari determinat i Realm assigna per a cada partició un arxiu sincronitzat individual. D'aquesta manera es pot distribuir i assignar a cada usuari les dades que necessita i les que pot veure o modificar.

Està pensada per aplicacions en les quals un subconjunt de documents (partició) del clúster sincronitzat estan relacionats entre ells i tenen els mateixos permisos de lectura i escriptura per a cada un dels usuaris. La Fig. 6.3.1.1.1 mostra el comportament d'aquest tipus de sincronització.



XV Fig. 6.3.1.1.1 Model de funcionament de particions. MongoDB, 2022.

6.3.1.1.1. Clau de partició

Es tracta d'un camp que s'especifica quan es configura el Sync que s'utilitza per determinar quina partició conté cada document determinat. Aquesta clau pot ser: un camp més que forma part del document i ajuda a dividir la lògica de les dades. Com per exemple guardar

un valor ID per usuari que s'utilitza de clau de partició i d'aquesta manera cada document és privat per a cada usuari específic o pot ser una partició sintètica per partir les dades.

Es pot escollir que la clau de partició sigui obligatòria o opcional, però Realm assignarà a qualsevol objecte una clau de partició nul·la predeterminada si es crea des de l'aplicació.

Aquestes claus de partició poden ser variables de 4 tipus: String, ObjectID, Long o UUID i els usuaris de Realm mai poden canviar aquest valor ja que significa editar els privilegis. Les claus de partició utilitzaran sempre el mateix nom de camp amb tots els documents sincronitzats i aquests no han de col·lisionar ni causar incongruències amb el nom dels camps en el model de cap objecte.

6.3.1.1.2. Valor de partició

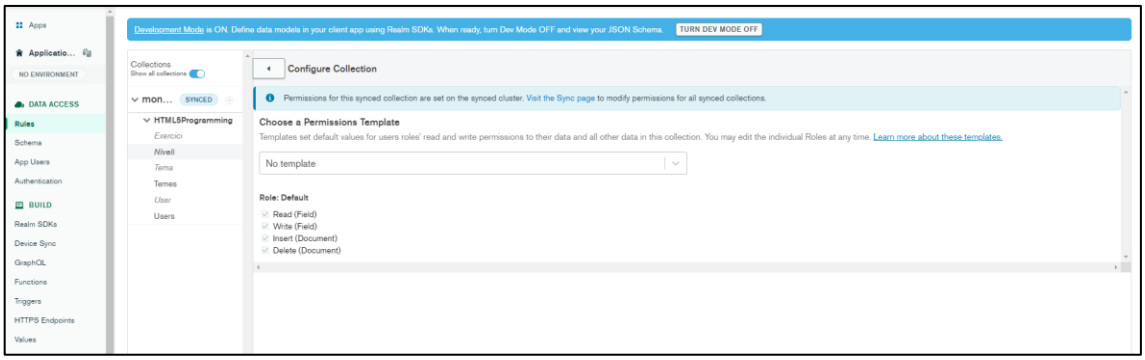
És el valor de la clau de partició d'un document sincronitzat. Els documents que tinguin aquest valor igual vol dir que pertanyen a la mateixa partició i que es sincronitzen amb el mateix arxiu de domini i comparteixen els privilegis d'accés a les dades (a nivell d'usuari). Se'l pot reconèixer com l'identificador del Realm sincronitzat que si es canvia el seu valor es perdran els canvis no sincronitzats que estan emmagatzemats localment en l'aplicació del client. Com s'ha dit anteriorment en l'app s'utilitza la partició per diferenciar les dades dels diferents usuaris i cada cop que es registri un usuari, l'aplicació crearà una nova col·lecció amb la partició de l'identificador del nou usuari.

6.3.1.2. Sincronització flexible

Amb aquesta sincronització es poden definir consultes en l'aplicació del client i sincronitzar només els objectes de la consulta. Cada cop que el client fa una consulta App Services busca el conjunt de dades al costat del servidor que coincideixin i mitjançant el motor de sincronització aplica un conjunt de regles per determinar si l'usuari té privilegis per accedir. Només retornarà les dades si l'usuari té permisos. Aquestes regles i permisos són personalitzables i es poden editar per document o col·lecció sencera utilitzant els rols d'usuari. Per aquesta sincronització també és necessari tenir definit un esquema d'objectes i utilitzar SDK a l'aplicació. Tot i que aquest mètode és molt dinàmic i té molta projecció encara està en fase experimental, ja que ha sortit fa molt poc.

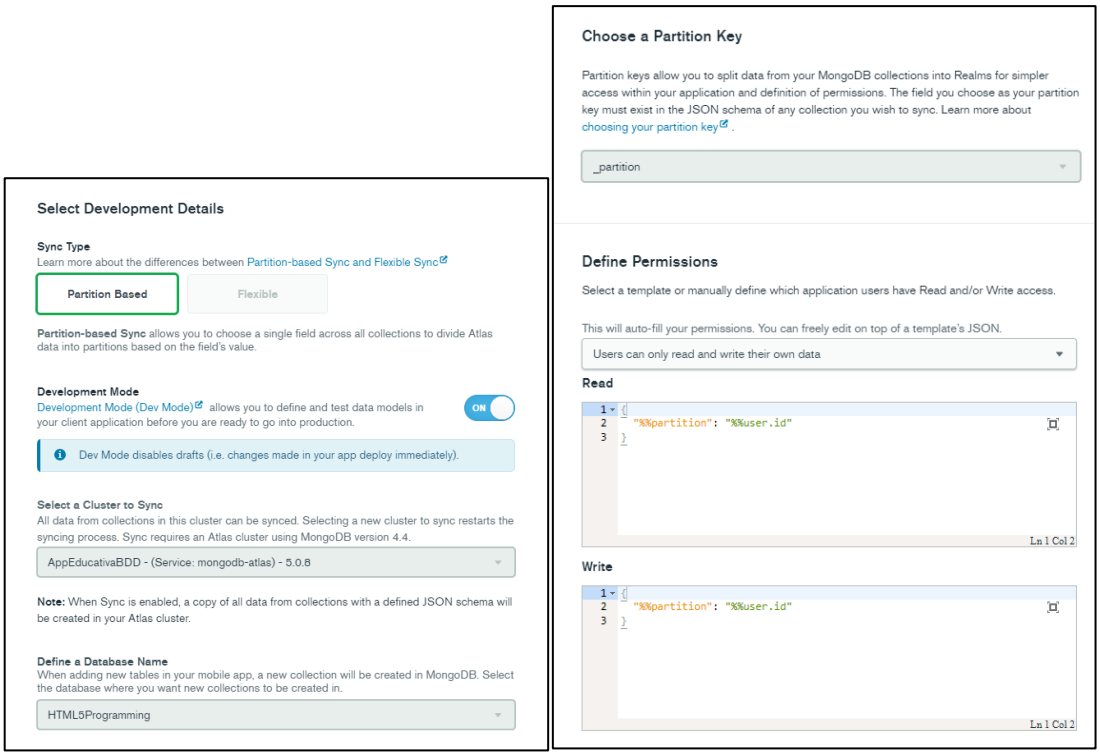
6.3.2. Sincronització de la aplicació

Per l'aplicació s'utilitza l'opció de la partició perquè permet separar tota informació perfectament per usuaris a més d'integrar-se perfectament amb els SDK de Realm. A més a més, segons la partició s'especifica quin privilegi tenen amb el servei de "Normas" sobre cada objecte. És el mètode més utilitzat, amb més documentació i sense "bugs" que encara no estan solucionats.



XVI Fig 6.3.2.1 Permisos de l'usuari sobre els objectes. Pròpia, 2022.

Per habilitar la sincronització s'ha de fer a través de la interfície d'administrador de l'App Services on s'indiquen tot els passos. S'ha d'anar amb compte perquè no és el mateix iniciar-ho per primer cop que rehabilitar perquè s'ha finalitzat la sincronització. Reiniciar-ho comporta una sèrie de passos addicionals per tal d'evitar qualsevol conflicte entre els esquemes sobretot si en el moment de parar la sincronització hi havia algun client connectat, doncs aquest haurà de realitzar un restabliment manual del client per poder-se tornar a connectar remotament. A causa d'això quan el producte surti a producció s'implementarà un sistema de manteniment on els clients no puguin connectar-se de cap de les maneres mentre s'està pausat la sincronització i s'anul·larà un cop estigui rehabilitada.



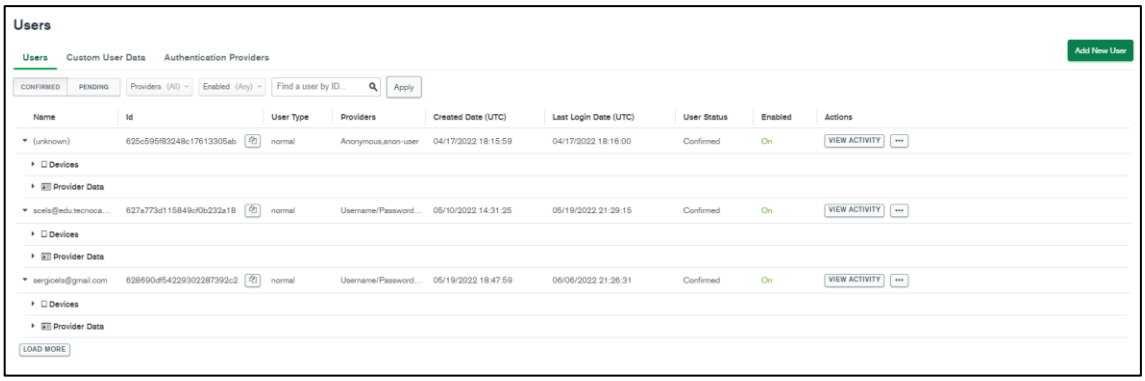
XVII Fig. 6.3.2.2 Activació de la sincronització. Pròpia, 2022.

A la Fig. 6.3.2.2 es pot veure la configuració feta per l'aplicació, on s'afegeix el user.id com a partició per llegir i escriure. Per tant, els documents privats per usuari, han d'incorporar aquest camp en l'esquema amb el valor de l'identificador de l'usuari.

6.3.3. Authentication provider

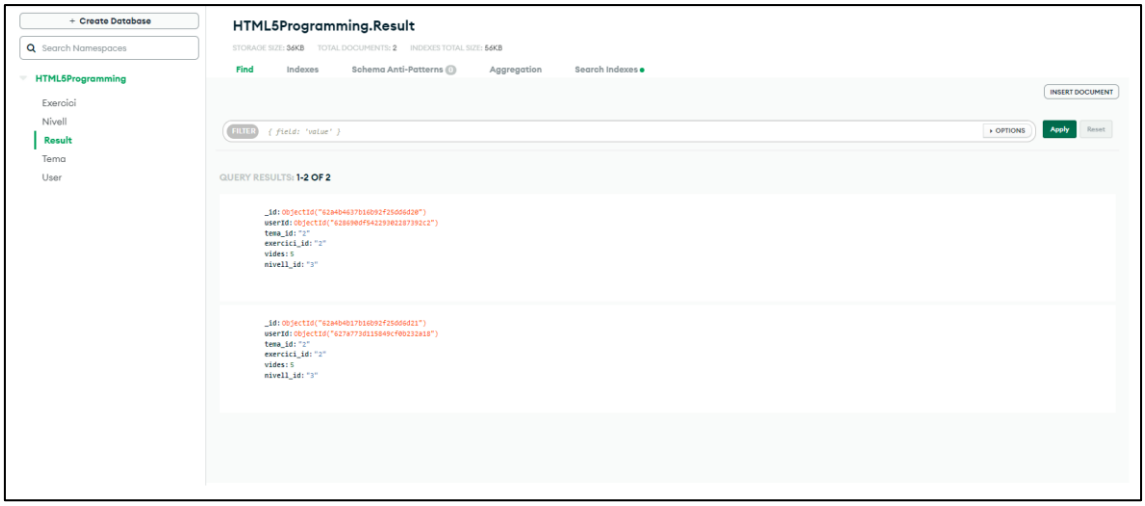
Atlas App Services ofereix més serveis a part del de sincronització, com la validació de dades, sistema d'autenticació integrat, disparadors, entre d'altres.

En l'aplicació s'utilitza el servei d'autenticació per poder accedir i registrar els usuaris. D'aquesta manera es té accés només a les dades de l'usuari a partir del seu identificador. A la vegada incorpora un registre d'activitat a l'aplicació, on a través de la interfície es pot veure qui ha accedit i registrat, historial de logs, peticions a la base de dades, dispositiu amb el qual s'ha connectat, entre d'altres.



XVIII Fig. 6.3.3.1 Registre d'inicis de sessió. Pròpia, 2022

Es poden crear usuaris a través de la interfície o directament des de l'aplicació si es registren a través de comandes Realm SDK. Cada cop que es registra un usuari, l'aplicació crea els objectes únics d'usuari que es guardaran a la base de dades com documents. Un dels objectes únics és "Result" on es guarden dades com la puntuació i progressió total, el nivell per el que va, i les vides.



XIX Fig. 6.3.3.2 Dades i col·lecció Result. Pròpia, 2022.

Aquest servei ofereix moltes opcions disponibles per a registrar-se, encara que estan totes desactivades per defecte quan es crea el App Services. En el cas de l'aplicació s'ha habilitat el registre per correu i contrasenya. Tot i que per realitzar les proves i el desenvolupament també s'ha habilitat l'autenticació anònima. És una mesura temporal per tal de fer les proves d'aplicació més ràpides i no haver d'iniciar sessió cada vegada que es vol provar l'aplicació. En el moment que passi a producció es deshabilitarà aquesta opció. Per afegir extra de

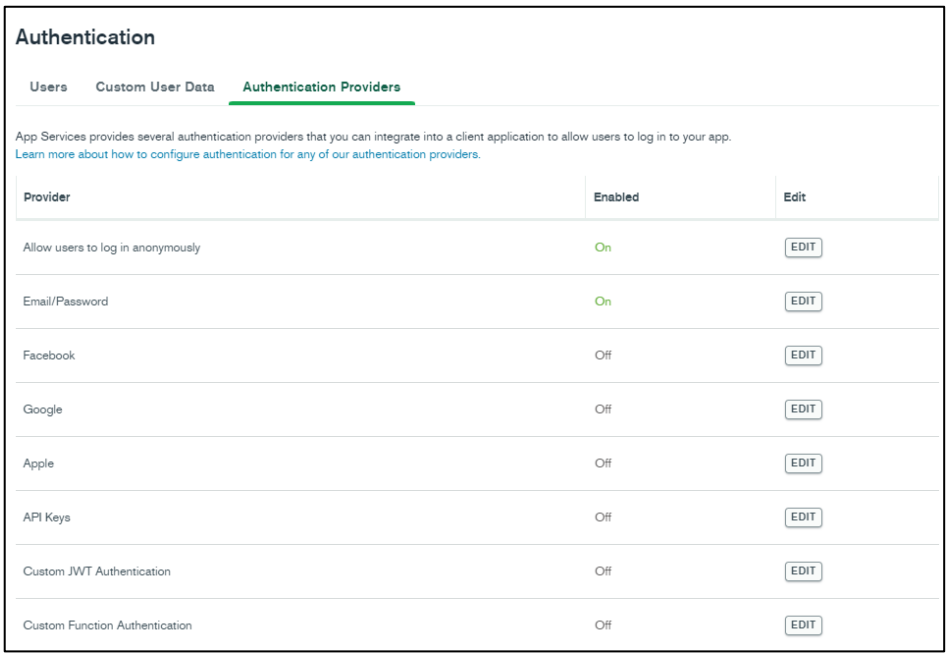
seguretat a les peticions de l'aplicació es pot configurar i activar claus API o utilitzar tokens JSON firmats per un sistema extern (JWT). Per últim, es pot integrar proveïdors d'autenticació externs com Google, Facebook o Apple ID.

Tot i que l'aplicació incorpora el registre de correu i contrasenya, App Services té un sistema de confirmació de correus electrònics en el que envia un URL al compte i l'usuari disposa de trenta minuts per visitar i confirmar. Així i tot, per a la fase de preproducció s'utilitza l'opció de confirmar els usuaris automàticament. El restabliment de contrasenya segueix el mateix patró que el registre, a través de un URL al correu del compte.

Els SDK locals permeten iniciar sessions de diferents usuaris en el mateix dispositiu però, només un compte pot estar actiu al mateix instant.

Quan l'usuari ingressa a la sessió aquest ja no cal que torni a introduir la contrasenya per tractar les dades, ja que Atlas App Services administra aquestes sessions amb tokens d'accés i d'actualització. Els tokens d'actualització tenen una caducitat màxima de 60 dies excepte el d'usuari anònim que són il·limitats. De totes maneres es poden eliminar els inactius quan l'administrador vulgui o crear disparadors que ho facin automàticament.

El token de sessió es cancel·la quan s'invoca el SDK de tancament de sessió. El qual elimina tota informació de la sessió guardat en local i els dos tipus de tokens. En el Back-End de l'App Services s'elimina el token d'actualització.



XX Fig. 6.3.3.3 Configuració de les opcions de autenticació. Pròpia, 2022

Per acabar amb aquest servei, ofereix una opció anomenada “Custom User Data” que permet guardar dades personalitzades al clúster en forma de camp a través dels objectes Users com per exemple el sexe i l’edat.

El que fa el servei és buscar el document personalitzat de l’usuari i l’inclou en el token d’accés quan s’inicia la sessió. És per això que aquestes dades han de ser el menor possible pel fet que la capçalera HTTP ha de mantenir una càrrega petita de dades. En el cas que hi hagués diferents documents que tinguessin el mateix camp d’usuari el sistema li dona preferència al que va entra primer. L’aplicació recull aquestes dades a partir de l’objecte Realm “User”.

Així com es poden registrar usuaris, també es poden eliminar o deshabilitar. A través de funcions SDK de Realm poden eliminar el seu propi compte o deshabilitar-la temporalment. Actualment, a l’app educativa només hi consta el mètode d’eliminació total del compte sense opció de recuperació.

Aquests són els serveis que s’han implementat a l’aplicació, però Atlas App Services ofereix molts més gratuïtament i que podrien ser noves ampliacions de cara al futur. Entre elles estan: GraphQL API, disparadors, funcions personalitzades, allotjament estàtic.

6.4. Realm Database

Per a la base de dades local s’utilitza Realm que tot i ser la tecnologia puntera en el sector de dispositius mòbils permet treballar sense connexió i sincronitzar-se amb la base de dades/clúster de Back-End un cop es connecti utilitzant els SDK. Els SDK són kits de desenvolupament de software que s’han d’importar a la nostra aplicació/projecte per fer servir les variables i mètodes pertinents. En el cas de Realm té uns propis que s’han d’instal·lar al projecte a través del fitxer build.gradle en format de plugin i dependències. A part, també s’ha d’habilitar la funcionalitat de sincronització a “true”.

Un altre avantatge que proporciona és que és utilitzable també amb IOS per si es volgués crear l’app per a sistema operatius Apple.

Realm Database utilitza un motor de base de dades, un format d’arxiu i un disseny completament únic i diferent de qualsevol model de dades semblant al SQL. Enlloc de construir-se sobre un motor de base de dades subjacent com SQLite utilitza les capes

d'emmagatzemat subjacents de Realm Database Tree B + per organitzar els objectes. Les dades es guarden en els Realms que són col·leccions d'objectes i cada un d'ells equival un registre/fila d'una taula SQL o un document Mongo del clúster. Els Realms guarden les dades a arxius que queden emmagatzemats als dispositius i n'hi ha de tres tipus: els arxius de Realm on es guarden les dades dels objectes, els arxius de bloqueig per fer un seguiment de les dades actives per tal de no malgastar espai, arxius de notes per enviar notificacions a través de processos i subprocessos i els arxius de gestió.

Aquest garanteix que les transaccions siguin ACID, és a dir, tinguin atomicitat, consistència, aïllament i durabilitat. Així es confia que les operacions d'escriptura són validades i els clients no visualitzaran estats transitoris si es bloqueja l'aplicació.

Com s'ha dit anteriorment, RealmDatabase s'encarrega de l'emmagatzematge local que s'executa directament en els dispositius de cada un dels clients i s'hi pot accedir fent ús del llenguatge de consulta natiu de la plataforma. Realitzar aquest emmagatzematge, accés i actualitzacions de les dades és senzill i lleuger i sempre es fa, en primer lloc, en aquesta base de dades. Quan Realm Sync està activat, es sincronitzen les dades amb MongoDB a través de la xarxa en un fil de fons. La sincronització incorpora un sistema de resolució de conflictes de manera consistent a cada client i en el clúster vinculat de MongoDB Atlas. Els objectes de l'aplicació sempre mostraran les dades que es guarden a la base de dades local podent subscriure als canvis per mantenir la interfície sempre actualitzada. Però per poder fer ús dels serveis integrats de App Services com és el cas de l'autenticació i sincronització són necessàries diverses coses.

Prèviament a utilitzar la base de dades s'ha de declarar la seva configuració al moment de crear-la. La configuració serà diferent si l'app vol que tingui només emmagatzematge local al fet que el tingui remot i sincronitzat tot i que de les dues maneres sempre tindrà una local. Com el cas de l'aplicació és el segon, s'ha d'utilitzar la configuració SDK explícita per a poder sincronitzar que requereix més paràmetres que la normal.

Abans de fer ús de les dades del servidor, s'ha d'inicialitzar el que serà el Realm, diferent per cada dispositiu. Un cop s'hagi inicialitzat, amb la configuració de l'aplicació i autenticat un usuari es pot obrir el Realm sincronitzat. Aquest s'inicialitza un cop cada vegada que s'executa l'aplicació i guardarà les dades específiques de l'usuari en local. Cada cop que el client vulgui consultar o editar les dades es farà sobre aquest Realm sincronitzat. El que fa l'aplicació quan té connexió és sincronitzar-lo amb la base de dades remota per comparar i

veure si s'han generat canvis. Gràcies al sistema de resolució de conflictes que incorpora i les regles de prioritat de canvis s'actualitzen les dades un cop la sincronització és correcte. Si el dispositiu té connexió, la sincronització és instantània. La part bona de treballar amb aquest Realm sincronitzat es que es pot treballar sense connexió i aconseguir una aplicació molt més lleugera en contenir només les dades necessàries.

El fet d'utilitzar els serveis integrats d'App Services al moment de crear l'app en el codi s'ha de determinar la Id de l'aplicació. En aquesta imatge s'està inicialitzant el Realm i construint l'objecte app des d'on es faran les crides.

```
Realm.init( context: this);
String appID = "application-html5programming-aohrd";
app = new App(new AppConfiguration.Builder(appID).build());
```

XXI fig. 6.4.1 Codi de creació de Realm. Pròpia, 2022.

El fet d'haver introduït les dades a través del servidor i generar l'esquema a través de la interfície d'Atlas App Services facilita molt escriure el codi dels objectes Realm, ja que el mateix servei que crea l'esquema a partir de les dades indica com han d'estar escrits. També és possible fer-ho a la inversa, però no és el cas de l'aplicació, aquí només es creen objectes únics de col·leccions que ja existeixen, però mai es crea una col·lecció directament des del codi.

Per obrir el Realm sincronitzat s'utilitza la configuració SyncConfiguration en lloc de RealmConfiguration que és el cas de si només es treballa en local i seria un simple Realm. Per obrir el sincronitzat és necessari un usuari autentificat i la clau de partició. Aquesta clau de partició indica quins objectes/documents pot carregar al Realm sincronitzat del clúster. A partir d'aquesta configuració es pot generar el que serà la instància Realm sobre el que es treballa, la qual aplicarà sempre els canvis al Realm sincronitzat que si pot es connectarà amb App Services.

Per realitzar les consultes des de la instància i modificar-la, s'utilitzen transaccions d'escriptura.

6.4.1. SDK asíncron i síncron

Els SDK de Realm permeten l'accés a les dades i recursos de dues maneres: asíncronament i síncronament. Amb les síncrones l'execució es bloqueja fins que es retorna èxit o falla, mentre que les asíncrones assignen una "callback" i segueixen la execució. Quan arriba la sol·licitud el "callback" s'executa per processar el resultat implementant el mètode *isSuccess()* i *onError()* (opcional).

Per defecte SyncConfiguration no permet transaccions síncrones per evitar bloquejos en l'execució o interaccions lentes amb l'usuari, però al moment de crear la configuració es pot afegir que les permeti.

Es per això que per a tots els mètodes de Realm existeix un mètode síncron i un mètode asíncron, que tenen el mateix nom, simplement que els asíncrons acaben amb el sufix "Async".

6.4.2. Transaccions d'escriptura

Un cop s'ha obert el Realm sincronitzat es poden modificar les dades en un bloc d'escriptura. Normalment, s'utilitza per executar sentències en bloc, ja que la transacció només retorna èxit si totes les operacions fetes han sigut correctes.

Només es pot executar una transacció alhora i fins que no s'acaba no pot començar una altra. Per invocar el mètode és necessari la instància de Realm i utilitzar el mètode síncron *executeTransaction()* o asíncron *executeTransactionAsync()*. Dintre d'aquesta funció es

poden inserir i eliminar objectes, realitzar una petició get d'un objecte en concret utilitzant les eines de filtre per a modificar-lo, entre d'altres.

Quan la transacció finalitza aquesta es confirma i escriu tots els canvis a l'arxiu Realm i posa en cua els canvis per sincronitzar o es cancel·la descartant tots els canvis de dintre la transacció.

Si es volen consultar o modificar llistes d'objectes aquest tornaran en forma de *RealmResults<RealmObject>* que és una llista d'objectes Realm. Un exemple de codi a l'aplicació seria la següent:

```
realm.executeTransaction(r -> {
    // Instantiate the class using the factory function.
    RealmResults <Tema> listaTemes = r.where(Tema.class).findAll().sort( fieldName: "_id", Sort.ASCENDING);
    listaTemes.first().setNomCurs("CURS 1");
    listaTemes.first().setNumTemes(6);
});
```

XXII Fig. 6.4.2.1 Transacció d'objectes "Tema". Pròpia, 2022.

Aquesta sentència de l'aplicació el que fa és guardar en una variable RealmResults d'objectes Tema tots els objectes Tema ordenats pel seu identificador. Seguidament, agafa el primer Tema i li modifica el nom del curs i el nombre de temes. Si la transacció és exitosa es guardaran els canvis i es posaran a la cua per a sincronitzar. En aquest cas s'utilitza el mètode *getDefaultInstance()* ja que prèviament quan l'usuari s'autentifica, l'aplicació guarda la configuració com per defecte. Això ens permet poder cridar cada cop que es vulgui utilitzar Realm amb la configuració i no haver de crear una instància global. A la Fig. 6.4.2.2 s'observa el codi del mètode d'inici de sessió on s'autentica l'usuari, es guarda la configuració del Realm sincronitzat i a més a més s'habilita perquè es puguin fer crides síncrones. S'ha implementat una alerta per si l'inici de sessió no fos exitós.

```
public void login(String email, String password) {
    AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
    Credentials credentials = Credentials.emailPassword(email, password);
    app.loginAsync(credentials, new App.Callback<User>() {
        @Override
        public void onResult(App.Result<User> result) {
            if (result.isSuccess()) {
                User user = app.currentUser();
                String partitionValue = user.getId();

                SyncConfiguration config = new SyncConfiguration.Builder(user, partitionValue)
                        .allowQueriesOnUiThread(true)
                        .allowWritesOnUiThread(true)
                        .waitForInitialRemoteData( timeout: 500, TimeUnit.MILLISECONDS)
                        .compactOnLaunch()
                        .build();
                Realm.setDefaultConfiguration(config);
                showCollection();
            } else {
                builder.setTitle("Login");
                builder.setMessage("Credencials incorrectes");
                builder.setPositiveButton( text: "Aceptar", listener: null);
                AlertDialog dialog = builder.create();
                dialog.show();
            }
        }
```

XXIII Fig. 6.4.2.2 Mètode de inici de sessió. Pròpia, 2022.

Realm també disposa de relacions, en el que en SQL serien les claus foranes FK. Aquestes permeten relacionar valors d'objectes amb valors d'altres objectes. Aquestes relacions han de quedar especificades en l'esquema. En el cas de l'aplicació existeixen diferents relacions entre objectes que es poden veure en el esquema de propietat de l'app. Existeixen les relacions un a un en el que només actuen dos objectes, les relacions un a molts en què el valor d'un objecte està relacionat a molts altres i la relació inversa. En l'aplicació s'apliquen les relacions un a un i la d'un a molts com per exemple un grup de nivells està relacionat a un tema en concret.

Com s'ha comentat anteriorment l'esquema del projecte s'ha fet a través del backend i no a través del codi.

Cada objecte Realm ha de tenir obligatòriament una clau primària que s'indica en el codi. Després qualsevol camp d'un objecte que figuri al servidor ha d'aparèixer en el codi com a camps obligatoris, usant la sentència @Required.

Finalment, es poden tractar col·leccions o llistes utilitzant objectes de tipus RealmSet, RealmDictionary, RealmList i RealmResuts.

6.5. Android Studio i Java

Com s'ha comentat anteriorment Android Studio és un IDE oficial pel desenvolupament d'apps Android amb llenguatge Java o Kotlin. Aquest ofereix, un sistema de compilació basta en Gradle, diferents tipus d'emuladors per testejar l'aplicació abans de ser llençada o actualitzada (si està habilitada l'opció "developer" a un mòbil Android també es pot carregar), integració amb Github, entre d'altres.

L'estructura del projecte està dividit en 2 mòduls per defecte i és convenient partir d'aquesta estructura. El primer mòdul és l'aplicació en si, on resideixen tres subdirectoris: manifests, java i res. En el "manifests" consta només un arxiu "AndroidManifest.xml" que és fonamental per descriure la informació essencial de l'aplicació. Aquest arxiu a de declarar principalment: el nom del paquet de l'aplicació i les eines de compilació, els components de l'aplicació, els permisos i les funcions hardware i software que requereix. Quan es crea un nou projecte amb Android Studio, aquest arxiu es crea automàticament amb les configuracions essencials i cada cop que és necessari afegir alguna cosa nova s'afegeix automàticament quan es compila l'aplicació.

El segon subdirectori es diu "java" i conté tots els arxius de codi i test. En aquesta carpeta és on s'afegiran totes les classes, interfícies, JUnit, etc. En el cas del projecte s'ha dividit aquesta carpeta amb més subdirectoris per tal d'ordenar les classes i tenir una estructura llegible en els paquets de "domain", "fragments", "views" i "bdd".

Per últim, està el subdirectori "res" on es guarden tots els recursos sense codi, és a dir, els layouts, les imatges, els sons, etc.

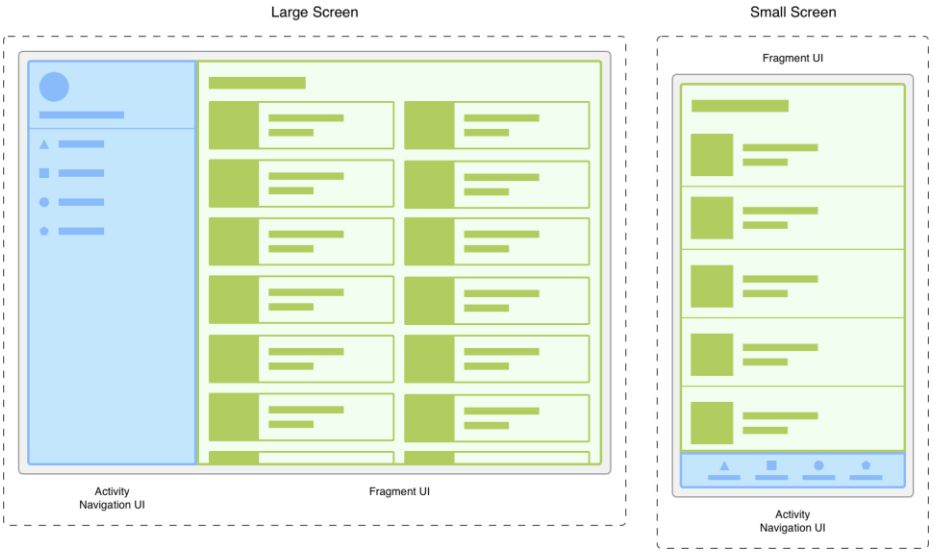
Per defecte l'aplicació té moltes plantilles d'exemple, tant de codi com de layouts en format xml. S'han utilitzat patrons d'aquestes plantilles per desenvolupar pantalles de l'aplicació.

L'emulador permet testejar l'app en molts tipus de dispositius i de diferents resolucions, des de mòbils o tauletes fins a televisors intel·ligents.

El codi es basa principalment en la classe "Activity" que vindria a ser com la classe Main, però amb la diferència que es poden crear més d'una. Per declarar una classe activitat aquesta ha de tenir la "extends AppCompatActivity" i implementar diferents mètodes. Android Studio inicia el codi a través d'una instància de "Activity" que retorna els mètodes que corresponen a cada etapa de vida l'activitat, per exemple: el primer mètode que s'invoca és

el onCreate() que s'executa només a l'invocar l'activitat o el onDestroy() que és just abans que s'acabi. L'activitat proporciona la finestra on l'app dibuixarà la interfície. Es podria dir que cada pantalla és una activitat i quan es vol canviar s'ha de crear una de nova tot i que sempre hi ha una activitat principal. Però això no és un mètode eficaç ni òptim amb aplicacions de moltes pantalles, ja que tenir diferents activitats obertes consumeix molta més RAM i CPU i la comunicació entre activitats és molt limitada. Cada una de les activitats han d'estar declarades a l'arxiu manifests, que es posen automàticament quan es compila l'aplicació.

En lloc de crear una activitat per pantalla s'ha utilitzat la classe "Fragment" que representa una part reutilitzable de la interfície. Per declarar un fragment s'ha de crear una classe i tenir "extends Fragment". Cada un defineix i administra el seu propi disseny i té el seu cicle de vida propi. No poden existir per si sols, han d'estar allotjats en alguna activitat, normalment la principal. Tot i que habitualment es compleix que cada fragment correspon a una pantalla també es pot configurar que una mostri dos fragments. Això és interessant en el cas de la pantalla en horitzontal.



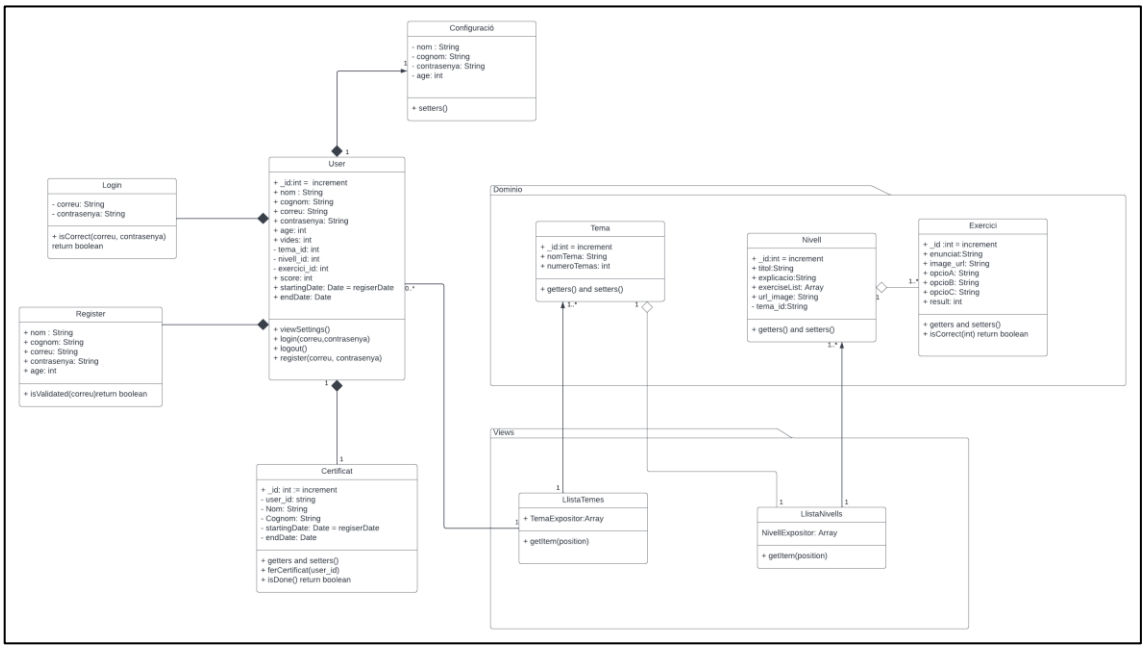
XXIV Fig. 6.5.1 Exemple de pantalles amb la classe Fragment. Android Developer, 2018.

A l'aplicació cada classe fragment li correspon un layout xml, llavors el joc està en què l'activitat pot manipular els diferents fragments per mostrar una pantalla o una altra en temps d'execució. Per a la creació de fragments s'ha utilitzat el patró Singleton, d'aquesta manera es restringeix la creació de fragments cada vegada que s'accedeix a la pantalla que el mostra.

L'aplicació està dotada de top tipus d'ítems i elements com Button, TextView, EditText, RecyclerView (contenidor d'elements), DataSet i tots els tipus de RealmObject creats.

6.6. Funcionament i estructura de l'aplicació

Per entendre com funciona l'aplicació, els seus components i les classes és necessari dissenyar un digrama de classes UML. Això permet veure quines classes es necessiten, com es relacionen i quines variables té cadascuna.



XXV Fig. 6.6.1 Diagrama de classes. Pròpia, 2022.

Primerament, apareix la pantalla d'inici de sessió i es mostra el fragment d'aquest. En ell apareix el camp de correu i contrasenya per accedir i un botó amb l'opció de registrar-se el qual porta al fragment de registre. Un cop s'ha accedit amb un usuari, mostra un llistat de temes en forma de RecyclerView<Tema> amb el títol de cada un d'ells. Si és un usuari nou apareixen tots els temes bloquejats excepte el primer, si no és la primera vegada que s'accedeix, deixa continuar a partir d'on es va quedar l'última vegada.

A dins de cada un dels temes hi ha diferents nivells, i passa el mateix que amb els temes, estan bloquejats i només es pot accedir al nivell que estava o anteriors. Això es fa perquè l'usuari segueixi un fil de dificultat i comenci amb el més bàsic i acabi amb el més complex.

A dintre de cada un dels nivells apareix una petita explicació teòrica amb imatges depenent del nivell i deu exercicis de tipus test per a respondre. Cada cop que es falla un pregunta, es resta una vida del total de 5 que disposa l'usuari. Cada cert temps es genera una nova vida automàticament. Quan s'està a la pantalla de llista de temes hi ha dos botons: un per tancar la sessió i l'altre per accedir a la configuració on es pot canviar algunes dades l'usuari, activar/desactivar el so de l'aplicació i canviar l'idioma.

Quan s'acaben tots els temes i nivells, apareix una opció per descarregar un certificat en format PDF indicant que has superat el curs bàsic de HTML5. El PDF es genera a través de la classe PdfDocument.

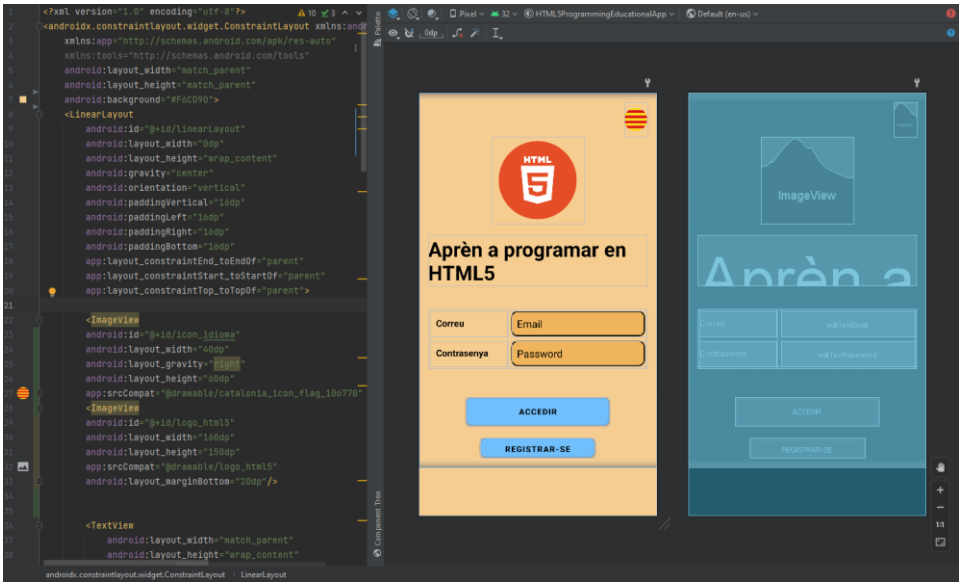
Tant els temes com els nivells com els exercicis es guarden estàticament a la base de dades, per tant, tots els usuaris consultaran els mateixos documents. El document/objecte únic que hi haurà per usuari és "Result" que guardarà el tema, el nivell, l'exercici i les vides en què es troba. Els temes, nivells i exercicis estaran relacionats de la següent manera: cada nivell tindrà un identificador propi i després un camp que es dirà "tema_id" que estarà relacionat amb el identificador del tema que li correspon. El mateix amb l'exercici, a part de l'identificador propi tindrà un camp anomenant "nivell_id" que fa referència a l'identificador del nivell que li correspon. Gràcies a aquestes relacions i els valors únics de las variables de "Result" es pot saber exactament en quin punt d'aprenentatge està l'usuari. El certificat també és un document únic que es farà a través de les dades de l'usuari.

6.7. Disseny de l'aplicació

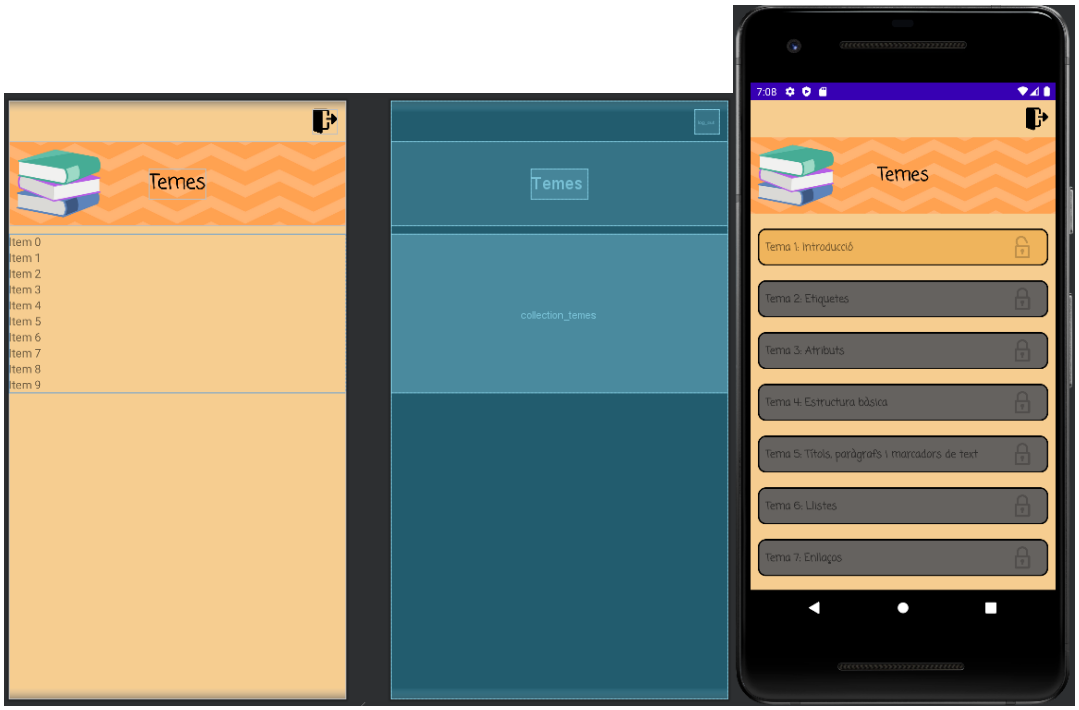
En Android Studio els dissenys es diuen layouts i són fitxers .xml semblant al HTML5 perquè també funciona amb etiquetes. Tot i que es poden dissenyar els layouts a través de codi també es poden fer a través d'una barra d'eines i disseny. Aquesta barra és útil per determinar colors o formes però per afegir elements, configurar "constraints" o definir les estructures de les parts de la pantalla s'ha d'escriure directament amb codi. Un punt a favor que tenen aquests layouts és que es pot veure en viu com està quedant la pantalla mentre s'edita. Existeixen multitud d'elements com "Button", "TextView", "EditText", "ImageView" entre molts altres i després està el que es diuen els grups de vistes que ajuden a estructurar aquests elements com per exemple els "LinearLayout" que apilen els elements

un darrer de l'altre amb l'opció de decidir l'orientació, els "GridLayouts" que col·loca els elements en quadricules, etc.

En els layouts de l'aplicació s'ha utilitzat principalment Linears i Grid layouts, i els FrameLayouts que el que fan és bloquejar una àrea de la pantalla pels elements que hi ha dintre. A la següent imatge es veu el layout del login on consten diferents TextViews, ImageViews, EditTexts i Buttons. Posteriorment al disseny de cada una de les pantalles es procedirà a través de classes Java a programar la funcionalitat dels elements de la pantalla. Aquest layout correspon a l'orientació del mòbil en vertical, però Android Studio permet crear un layout de la mateixa pantalla, però en format horitzontal i tindrà el mateix nom de fitxer simplement que al final entre parèntesis apareixerà un "Landscape". Per defecte si s'han organitzat els elements amb els grups de vistes no existirà cap problema quan es giri el telèfon però aquest mètode és molt útil per exemple en el cas que en vertical es vulgui mostrar un fragment i en horitzontal es visualitzin dos.



XXVI Fig. 6.7.1 Disseny de la pantalla d'inici de sessió. Pròpia, 2022.



XXVII Fig. 6.7.2 Disseny de la pantalla de col·lecció de temes. Pròpia, 2022.

7. Conclusions i possibles ampliacions

7.1. Conclusions del treball

Múltiples són els objectius que s'han plantejat a l'inici del treball i ha arribat el moment d'analitzar si s'han complert tots o la majoria.

Cal fer un incís del punt de partida del treball i del desenvolupador, ja que aquest partia amb uns nivells bàsics d'Android Studio i desconeixement total dels sistemes MongoDB Atlas i Realm Database. Per tant, en primer lloc, es pot dir que aquest treball ha permès conèixer i profunditzar en aquestes tecnologies de manera autodidàctica tant a nivell teòric com a nivell pràctic, que era el principal objectiu. Ha permès conèixer les diferents formes de funcionament, plataformes i capes de les aplicacions mòbils i les limitacions de les tecnologies fetes servir.

Primerament, s'han hagut d'integrar totes les bases teòriques sobre els diferents conceptes i sistemes que ha provocat un endarreriment general de les diferents entregues a les dates inicialment proposades. Això es deu a la magnitud de la documentació de MongoDB i les infinites possibilitats que ofereix. És una plataforma gegant amb infinites funcionalitats, configuracions, implementacions i aplicacions dels serveis que apareixen, que per escollir la millor opció pel treball s'ha revisat i avaluat cada una d'elles.

Haver de crear un projecte de desenvolupament de zero ha permès recuperar conceptes d'enginyeria del software estudiats al llarg de la carrera, i només això, si no saber en cada moment què i com toca fer-ho i aplicar-ho al projecte.

Una conclusió que extreta de l'experiència realitzant el projecte és que per molt moderna, antiga, complexa o senzilla que sigui la metodologia de treball o de desenvolupament de software, sempre poden existir desviacions i marges d'error. És cert que cada projecte s'adapta a una o altre millor i pot ser més o menys concisa, però cap d'elles garanteix un projecte lineal sense entrebancs. Sempre s'ha de deixar un marge d'error per aquest que poden provocar no arribar a temps a les dates d'entrega.

A nivell legal s'ha conegut els límits que ha de complir una aplicació i que en el món real sembla que no existeixin però hi són. Són termes molt importants que la gent passa per alt i cada cop se li dona més importància degut a l'avanç de les tecnologies. La gran majoria d'aquests límits tenen a veure amb la privacitat de les persones i les seves dades. Passar per

alt qualsevol d'aquests aspectes pot provocar la caiguda del producte, per tant, del projecte i pèrdues considerables a l'empresa que el desenvolupi.

Per acabar, i el més important, s'ha treballat com si fos una empresa de veritat amb un projecte real de client, amb les seves respectives entregues, seguiments i tancament de projecte. Sobretot, fent servir una eina de compartició d'arxius principalment de codi com Github i estructurant-ho com una empresa professional.

7.2. Conclusions del producte

Qualsevol aplicació, sigui de la temàtica que sigui, sempre té marge de millora o funcionalitats extres que es poden afegir. En el cas de l'app del treball compleix amb les bases que s'havien fixat ens els objectius i les funcionalitats prioritàries descrites en el backlog del projecte, però com a professional se'l definiria com un prototip o primera versió, no com un producte final. Amb més hores de desenvolupament pur i amb el coneixement après al llarg del treball sobre Android Studio i sobretot MongoDB juntament amb RealmDatabase és segur que el producte milloraria considerablement els pròxims mesos.

Es creu que els objectius secundaris i les funcionalitat afegides amb poca prioritat del producte plantejats a l'inici no contemplaven la complexitat del sistema de base de dades i el seu aprenentatge. Malgrat això, el producte compleix els objectius principals i les bases que es van escriure als requeriments, simplement que hi ha funcionalitats que no s'han arribat a desenvolupar, però es podrien fer.

L'objectiu principal és que l'usuari aprengui HTML5 de manera dinàmica amb l'aplicació, com això es possible es pot dir que s'ha aconseguit un producte exitós.

7.3. Possibles ampliacions del producte

Les primeres ampliacions que es podrien afegir en el producte són, en primer lloc, les que no s'han pogut implementar durant el TFG a causa de les desviacions del projecte.

Una d'elles és permetre autenticar-se a través de comptes de Google i Facebook. Tot i que el sistema d'autenticació que incorpora App Services té la opció de fer-ho directament, falta programar-ho. Era una funcionalitat de prioritat baixa.

Un altre punt que és interessant és el fet de poder agregar a altres usuaris a la teva llista d'amics i poder fer competicions a través de rondes de preguntes. Aquesta funcionalitat implica haver d'ampliar el banc d'exercicis considerablement per no repetir-ne.

Integrar una passarel·la de pagament que permet recarregar les vides i obtenir una aplicació sense anuncis, els anuncis s'afegeixen quan es llença l'aplicació a la plataforma de GooglePlay.

Aquestes han sigut funcionalitats que es volien inicialment i no s'han pogut realitzar. Durant el projecte, mentre s'aprenien i descobrien les tecnologies de Mongo i Realm també han sortit de noves que poden ser molt interessants de cara al futur com per exemple: implementar en el servidor Mongo un sistema de rendiment i activitat dels usuaris per realitzar informes sobre el funcionament i estudiar els gustos dels usuaris, fer ús dels "triggers" per automatitzar accions tant a nivell de servidor com d'aplicació o dintre de la mateixa app habilitar l'aprenentatge de diferents tipus de llenguatge.

Per acabar, seria molt atractiu i es desmarcaria al mercat de les altres aplicacions semblants integrar una mena de compilador per poder escriure i programar codi directament a l'aplicació.

8. Bibliografia

[1] "Top de apps para aprender a programar", Andro4All, 2022. [En línea] Disponible a: https://andro4all.com/listas/apps-android/las-mejores-apps-para-aprender-a-programar [Accedit: 15-01-2022]

[2] "Alcance de una app", Workana, 2020. [En línea] Disponible a: https://blog.workana.com/emprendimiento/como-crear-una-app-alcance-del-proyecto/ [Accedit: 20-01-2022]

[3] "LMS 101: Rethinking Your Aproach To Employee Training", Forbes 2018 [En línia] Disponible a: https://www.forbes.com/sites/paycom/2017/02/14/learning-management-systems-101-rethinking-your-approach-to-employee-training/ [Accedit: 22-01-2022]

[4] "Desarrollo de una App para el aprendizaje móvil mediante actividades interactivas", Laura Martínez Delgado, 2017. [En línea] Disponible a: https://e-archivo.uc3m.es/bitstream/handle/10016/27837/TFG_Laura_Martinez_Delgado.pdf;jsessionid=A9130B30425F23E79279B472B3C13DBD?sequence=1 [Accedit: 25-01-2022]

[5] RIBAS LEQUERICA, Joan, 2012. Desarrollo de aplicaciones para Android. 1ª ed. Madrid: GRUPO ANAYA, S.A. ISBN: 978-84-415-390 [Accedit: 26-01-2022]

[6] "SQLite: herramientas y casos prácticos de gestión de bbdd", Academia Android, 2014, [En línea] Disponible a: https://academiaandroid.com/sqlite-introduccion-herramientas-administracion/ [Accedit: 30-01-2022]

[7] "SQLite en Android: creación y acceso base de datos e inserción de registros", Academia Android, 2016, [En línea] Disponible a: https://academiaandroid.com/sqlite-android-creacion-acceso-base-datos-insercion/ [Accedit: 30-01-2022]

[8] "¿Qué modelos de negocio para aplicaciones móviles existen?", Dimensiona, 2021, [En línea] Disponible a:https://www.dimensiona.com/es/modelos-de-negocio-para-aplicaciones-moviles/ [Accedit: 01-02-2022]

[9] "Metodologías de desarrollo de software", Universitat Carlemany, 2018, [En línea] Disponible a: https://www.universitatcarlemany.com/actualidad/metodologias-de-desarrollo-de-software [Accedit: 01-02-2022]

[10] "Metodologías para el desarrollo de aplicaciones móviles", Syntonize, 2020, [En línea] Disponible a:https://www.syntonize.com/metodologias-desarrollo-de-aplicaciones-moviles/#:~:text=Se%20basa%20en%20metodolog%C3%ADas%20para,y%20la%20fase%20de%20pruebas [Accedit: 01-02-2022]

[11] "Metodologías de desarrollo de software", Santander universidades, 2020, [En línea] Disponible a:https://www.becas-santander.com/es/blog/metodologias-desarrollo-software.html [Accedit: 01-02-2022]

[12] "GUIA BÀSICA SOBRE PROTECCIÓ DE DADES, PROPIETAT INTEL·LECTUAL I KNOW-HOW EN DOCÈNCIA, INVESTIGACIÓ I EMPRENEDORIA", Universitat de València, Javier Plaza Penadés, 2018, [En línea] Disponible a: https://www.uv.es/lopd/Informacion%20de%20interes/GUI%CC%81A%20datos%20para%20inves%20y%20Tfm%20y%20tfg%20cat.pdf [Accedit: 05-02-2022]

[13] "Evolución del mercado de las apps hasta 2021", Vanessa Estorach, 2021, [En línea] Disponible a: https://www.vanessaestorach.com/evolucion-mercado-de-las-apps-2021/ [Accedit: 07-02-2022]

[14] "HTML: Lenguaje de etiquetas de hipertexto", MDN Web Docs, 2010, [En línea] Disponible a: https://developer.mozilla.org/es/docs/Web/HTML [Accedit: 07-02-2022]

[15] "Información jurídica", Agencia Estatal Boletín Oficial del Estado, 2010, [En línea] Disponible a:https://www.boe.es/ [Accedit: 10-02-2022]

[16] "Centro de políticas para desarrolladores", Google Play, [En línia] Disponible a: https://play.google.com/intl/es/about/developer-content-policy/ [Accedit: 20-02-2022]

[17] "Licencias de apps", Developers Android, [En línia] Disponible a: https://developer.android.com/google/play/licensing?hl=es-419 Accedit: 20-02-2022]

[18] "MongoDB Atlas Documentation", MongoDB, [En línia] Disponible a: https://www.mongodb.com/docs/atlas/driver-connection/

[19] "MongoDB Realm Java SDK", MongoDB, [En línia] Disponible a: https://www.mongodb.com/docs/realm/sdk/java/

[20] "La plataforma diseño para no-diseñadores", Uizard, [En línia] Disponible a: https://app.uizard.io/