Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Arnela Jabalera, Adrián; Alsina Rodríguez, Aitor, dir. App de recull de dades georeferenciades per a la generació de rutes per a visites arqueològiques. 2022. (958 Enginyeria Informàtica)

This version is available at https://ddd.uab.cat/record/264187

under the terms of the BY-NC-ND license

App de recull de dades georeferenciades per a la generació de rutes per a visites arqueològiques

Adrián Arnela

Resum– En aquest projecte s'ha desenvolupat una aplicació per a dispositius mòbils destinada a la creació de rutes formades per punts d'interès arqueològic, que facilitarà a les institucions arqueològiques l'organització i el seguiment de les visites destinades al seu públic. L'aplicatiu mostra un mapa on l'usuari pot veure la seva posició, crear marcadors i adjuntar-los informació. Els usuaris poden provar la ruta que estan creant en entrar al mode simulació, on veuran la informació adjunta a un marcador en apropar-se. L'aplicació ha estat implementada emprant React Native, amb el suport de diversos components creats per la comunitat. Per facilitar l'organització de l'execució del projecte s'ha seguit la metodologia Kanban.

Paraules clau– React Native, Google Maps, Kanban, aplicació mòbil, multiplataforma, geolocalització, visites arqueològiques, localitzacions d'interès arqueològic.

Abstract– In this project, an application has been developed for mobile devices aimed at creating routes formed by points of archaeological interest, which will make it easier for archaeological institutions to organize and monitor visits intended for their public. The application shows a map where the user is able to view his own position, create markers and attach them information. Users can test the route they are creating by entering simulation mode, where they will see the information attached to a marker as they approach. The application has been implemented using React Native, with the help of some components created by the community. To ease the organization, the Kanban methodology has been followed.

Keywords– React Native, Google Maps, Kanban, mobile application, multi platform, geolocation, archaeological tours, archaeological sites of interest.

1 Introducció

Amb l'exponencial augment en l'ús de dispositius mòbils, qualsevol aplicatiu que desitgi arribar a un gran nombre d'usuaris necessita comptar amb una versió adaptada a les dues plataformes principals emprades per aquests dispositius, Android i iOS. A més, la millora en les prestacions dels dispositius mòbils ens permeten realitzar tasques que abans només es podien executar utilitzant computadores. Aquest fet permet extreure el màxim partit d'aspectes com la geolocalització, que és el cas que ens ocupa.

En aquest projecte es pretén desenvolupar un aplicatiu per dispositius mòbils que permeti als usuaris crear rutes formades per punts d'interès arqueològic, i enregistrar-les, de manera que altres usuaris puguin recórrer-les i aprendre sobre les diferents localitzacions que les conformen.

Aquest document recollirà un resum de la informació generada arrel de la realització del projecte, detallant les diferents fases del procés; es definiran els objectius principals i secundaris, es farà una breu introducció de les eines i tecnologies més rellevants implicades en el projecte, es detallarà la planificació de les tasques, així com la metodologia que se seguirà per afavorir l'organització, es llistaran els requisits funcionals i no funcionals, es detallaran els processos de disseny, implementació i test, i finalment es plasmaran les conclusions i es definirà el treball a dur a terme en un futur per tal de completar la funcionalitat de l'aplicatiu i millorar les seves prestacions i abast.

• E-mail de contacte: adrian.arnela@gmail.com
• Menció realitzada: Tecnologies de la Informació
• Treball tutoritzat per: Aitor Alsina Rodríguez (DEIC)
• Curs 2021/2022

2 OBJECTIUS

El principal objectiu d'aquest projecte és crear una aplicació mòbil que serveixi a institucions arqueològiques com a interfície per la creació i l'enregistrament de rutes per a visites arqueològiques. Els usuaris podran visualitzar la seva posició en temps real sobre un mapa i marcar punts sobre aquest per crear una ruta, així com adjuntar informació als diferents punts que conformen la ruta. Finalment, podran enviar la ruta creada, juntament amb la informació adjunta, a un servidor amb la finalitat que aquesta quedi enregistrada i altres usuaris puguin seguir-la des d'una altra aplicació. L'usuari també tindrà la possibilitat de provar la seva ruta abans d'enregistrar-la.

Complementant l'objectiu principal, cal destacar una sèrie d'objectius secundaris:

- Proporcionar una interfície intuïtiva i completament funcional per l'aplicatiu.
- Elaborar un codi amb un bon estil de programació, facilitant l'enteniment i el manteniment.
- Conèixer el funcionament del *framework* React Native, així com les seves possibilitats i limitacions pel seu ús en futurs projectes.
- Aprendre de la realització d'un projecte real de desenvolupament de *software* en aspectes de planificació i organització.

3 EINES I TECNOLOGIES

A continuació es farà una breu descripció de les principals eines i tecnologies escollides per a l'execució del projecte.

3.1 Marvel App

Marvel App és una eina en línia que permet crear prototips visuals per tal de facilitar el disseny de la interfície d'usuari en projectes de desenvolupament de programari. També permet programar les transicions entre les diverses pantalles que conformen una aplicació, així com els elements que activen aquestes transicions. Aquest aspecte permet que es puguin crear prototips destinats al testatge per part dels usuaris. En el nostre cas, utilitzarem aquesta eina per dissenyar un *mockup* que mostri la disposició dels elements dins les diferents pantalles. No utilitzarem aquest *mockup* per a realitzar proves d'usuaris pel fet que es vol obtenir una aplicació que no disposarà d'una gran varietat de transicions i pantalles.

L'elecció d'aquest programa radica principalment en el fet que s'ha utilitzat en projectes anteriors i es coneixen les funcionalitats principals de la interfície, i la versió gratuïta disposa de les funcionalitats necessàries per realitzar el prototip visual que es requereix.

3.2 Android Studio

Es tracta de l'entorn de desenvolupament integrat oficial per la plataforma Android. En aquest projecte es farà servir per a dos aspectes importants; instal·lar les *SDK* d'Android i els paquets necessaris gràcies al *SDK Manager*, i emular dispositius Android gràcies a l'*Android Virtual Device Manager* (Fig. 8), on podrem crear dispositius virtuals per emular les diferents versions d'aquesta plataforma i els seus serveis.

3.3 Github

Github és una plataforma de control de versions que emprarem per portar un seguiment dels canvis introduïts al codi de l'aplicació, així com en l'estructura del projecte, de manera que podem recuperar versions del codi anteriors a l'estat actual del desenvolupament si es requereix.

3.4 React

React és un *framework* de JavaScript que permet construir interfícies d'usuari interactives que actualitzen el seu estat a mesura que canvien les dades [2]. Està basat en components que conformen peces de la interfície d'usuari que es poden integrar entre elles per formar una interfície més complexa. Per simplificar la codificació de components, aquest *framework* fa ús de *JSX*, una extensió de sinaxis de JavaScript que interpreta codi semblant a *html* i ho tradueix a un element de React sense especificació explícita.

3.5 React Native

React Native [1] és un *framework* de codi obert que permet aplicar els fonaments de React més enllà de la web, permetent-nos crear aplicacions per plataformes com Android, iOS, macOS i Windows. Aquest *framework* permet construir aplicacions codificades en JavaScript, però que són compilades a codi natiu, permetent així esprémer la portabilitat del codi i reduir costs.

3.6 react-native-maps

React-native-maps [3] és una llibreria de components que usen l'*API* de Google Maps per representar un mapa real per pantalla. En el cas d'iOS també ens permet escollir l'*API* d'Apple Maps si ho dessitgem. També proporciona components per col·locar sobre el mapa diferents elements com marcadors i figures geomètriques. Aquesta llibreria ens facilita les eines necessàries per materialitzar les principals característiques de la nostra aplicació.

El motiu d'aquesta elecció es basa principalment en la popularitat del component, és el més utilitzat en el seu àmbit i, per tant, afavoreix la recerca d'informació gràcies a una major contribució de la comunitat.

3.7 Google Maps API

Per a poder utilitzar els mapes que proporciona Google, hem de connectar la nostra aplicació a l'*API* de Google Maps. Per poder fer servir aquesta *API* hem de generar una *API Key* que necessitarem perquè els components de la llibreria *react-native-maps* puguin accedir als serveis de Google Maps.

3.8 Trello

L'eina Trello [12] consisteix en un taulell virtual ideal per implementar el mètode Kanban per a la gestió de projectes [11]. S'ha escollit aquesta eina pel fet que ha estat emprada en projectes anteriors i cobreix les funcionalitats necessàries per a facilitar l'organització de l'actual projecte; ens permet crear targetes, assignar-los dates d'inici i finalització estipulades i crear diferents llistes per a organitzar les diverses targetes.

4 PLANIFICACIÓ I METODOLOGIA

Al llarg d'aquesta secció es detalla l'organització i la metodologia emprada en la realització del projecte.

A partir dels requisits del projecte, definits i exposats en la següent secció, s'han extret les tasques principals que s'han utilitzat com a full de ruta a l'hora de realitzar la planificació temporal del projecte (Fig. 9). El diagrama de Gantt s'ha realitzat utilitzant l'aplicació web TeamGantt [14].

Les tasques definides al diagrama de Gantt s'han implementat seguint un model en cascada, per tant, s'ha treballat sobre un únic prototip al qual se li han anat afegint les diferents funcionalitats en l'ordre definit a la planificació temporal del projecte.

Durant la realització del projecte s'ha seguit la metodologia de desenvolupament àgil Kanban, fent ús de l'eina Trello. El motiu d'aquesta elecció radica fonamentalment en la simplicitat que ens presenta el mètode, a més que no imposa cap limitació al ser utilitzat per equips amb un nombre reduït de persones o d'una sola persona com és el cas.

S'han dividit les tasques principals esmentades a la planificació en subtasques més petites. Cadascuna d'aquestes subtasques representa una targeta Kanban, d'aquesta forma s'ha pogut extreure el màxim partit a aquesta metodologia focalitzant-nos en el desenvolupament de funcionalitats més concretes que requerien menys temps per a ser implementades i que ens han ajudat a portar una millor organització (Fig. 10).

Cada divendres s'han destinat trenta minuts a analitzar el taulell per tal de valorar l'estat de les tasques, modificar alguns aspectes erronis i afegir noves tasques.

La realització de les tasques del taulell ha seguit sempre el mateix procés; s'estudien formes d'implementar la tasca, s'implementa i es valida el correcte funcionament. Si en realitzar les proves de funcionament no s'ha obtingut el resultat esperat, es tornen a estudiar formes d'implementar la tasca, o bé es prova alguna alternativa trobada anteriorment.

5 REQUISITS

En aquesta secció es presenten els requisits de la nostra aplicació classificats entre funcionals i no funcionals.

5.1 Requisits funcionals

Els següents requisits descriuen com s'ha de comportar l'aplicació en els diferents escenaris possibles.

RQF1. En iniciar l'aplicació apareixerà un menú d'inici de sessió.

RQF2. Es pot accedir a un menú de registre d'usuari des del menú d'inici de sessió.

RQF3. Els usuaris es poden enregistrar i iniciar sessió.

RQF4. En iniciar sessió correctament, es mostrarà un mapa real.

RQF5. S'ha de visualitzar la situació geogràfica actual de l'usuari sobre el mapa.

RQF6. L'usuari pot situar marcadors sobre el mapa polsant el punt del mapa on vol col·locar-lo.

RQF7. En situar un marcador s'obrirà un formulari on es podrà adjuntar informació i imatges al marcador.

RQF8. No es podran crear marcadors sense completar tots els camps del formulari.

RQF9. En polsar sobre un marcador ja existent, s'ha de mostrar un formulari d'edició similar al que apareix en crear un nou marcador.

RQF10. El formulari d'edició permetrà editar la informació adjunta a un marcador i eliminar-lo.

RQF11. Existeix opció per eliminar tots els marcadors.

RQF12. En eliminar un marcador, s'elimina la informació adjunta.

RQF13. L'aplicació ha d'estar connectada amb una *API* que permetrà guardar les rutes a un servidor.

RQF14. Si l'usuari ha situat més d'un marcador, podrà enviar la ruta a l'*API* perquè quedi enregistrada.

RQF15. Si l'usuari ha situat més d'un marcador, podrà simular la ruta abans d'enregistrar-la.

RQF16. En el mode simulació, es llençarà una pantalla amb la informació d'un marcador al situar-nos geogràficament prop d'ell.

RQF17. Es pot sortir del mode simulació i continuar creant la ruta.

RQF18. L'aplicació notificarà si la ruta s'ha enregistrat correctament.

RQF19. En enregistrar una ruta es tornarà al menú d'inici de sessió.

5.2 Requisits no funcionals

Els següents requisits descriuen característiques conceptuals de l'aplicació.

RQNF1. El conjunt dels marcadors creats formen una ruta.

RQNF2. L'aplicació presentarà un disseny intuïtiu.

RQNF3. Es necessitarà connexió a Internet per utilitzar l'aplicació.

RQNF4. Es necessitarà concedir el permís de geolocalització per fer servir l'aplicació.

RQNF5. Es necessitarà concedir permisos d'accés a la galeria d'imatges i/o a la càmera per adjuntar imatges a un marcador.

6 Disseny

La validació dels requisits del projecte dóna pas a la fase de disseny, destinada a estudiar i enginyar com materialitzar els requisits definits en el producte que es vol elaborar. L'elaboració del diagrama de flux de l'aplicació (Fig. 11), elaborat utilitzant l'aplicació web *draw.io* [15], ha marcat l'inici d'aquesta fase. A causa de la necessitat de familiarització tant amb una tecnologia relativament nova com amb unes plataformes desconegudes en termes de desenvolupament de *software*, el disseny de l'aplicació ha estat molt lligat amb una fase de preparació on s'han assolit nous coneixements sobre React Native, s'han provat components potencialment útils i s'ha estudiat el potencial del *framework* per tal de predir els punts del desenvolupament més problemàtics i com es podrien enfocar.

6.1 Interfície d'usuari

S'ha realitzat un *mockup* utilitzant l'eina Marvel (Fig. 12) amb l'objectiu de plantejar la disposició dels elements a les diferents pantalles que conformen l'aplicació per tal que la interfície d'usuari quedi definida prèviament a la implementació, no havent de prendre decisions importants relacionades amb el disseny de la interfície durant aquesta etapa.

6.2 Arquitectura

S'han fet servir els components que ens facilita la llibreria *react-native-maps* per implementar la base de la nostra aplicació, conformada per la representació del mapa i la creació i gestió de marcadors. S'ha escollit aquesta llibreria pel fet que ens proporciona les eines necessàries per implementar la funcionalitat desitjada, a més de comptar amb una àmplia comunitat que facilita la resolució de possibles problemes gràcies a la informació proporcionada pels usuaris.

Per representar la posició actual de l'usuari s'ha considerat la utilització de la llibreria *react-native-community/geolocation* [5]. Tot i que per obtenir la localització en la plataforma Android, la llibreria fa ús de l'*API android.location*, no recomanada per Google, s'han realitzat proves per validar el seu funcionament i s'han obtingut resultats satisfactoris. Tenint en compte les recomanacions de Google i la possibilitat de patir inconvenients en la geolocalització, s'estudiarà la utilització de la llibreria *react-native-geolocation-service* [4].

Inicialment es va decidir crear sis components principals: *App*, *Menu*, *Mapa*, *Formulari*, *Simulacio* i *InfoMarcador*, finalment s'ha descartat el component *Menu* i s'han afegit els components *Login*, *Signin* i *AppNavigator*.

A continuació una breu descripció del paper de cada component: El component *App* és el punt d'entrada de la nostra aplicació. És l'encarregat d'incloure la resta de components. La funcionalitat del component *AppNavigator* és gestionar les transicions entre les diferents pantalles de l'aplicatiu. Els components *Login* i *Signin* mostren interfícies perquè l'usuari inicií sessió o es registri, respectivament. El component *Mapa* inclou tots els elements relacionats amb la gestió del mapa i els marcadors. El component *Formulari* conté un formulari amb els components necessaris per adjuntar imatges i text als marcadors. El component *Simulacio* és molt similar al component Mapa, però està conformat per diferents events i funcions necessàries per a la simulació de la ruta per part de l'usuari. Finalment, el component *InfoMarcador* es crida en apropar-nos a un marcador en el mode simulació i mostra la informació que hi té associada.

Per enregistrar les rutes necessitarem establir connexió amb una *API rest*. Per tal de realitzar l'intercanvi de dades amb aquesta *API* s'utilitzarà l'*API Fecth* proporcionada per React Native. La documentació i connexió amb l'*API* forma part del treball futur d'aquest projecte i es donaran més detalls en la darrera secció d'aquest document.

7 Implementació

Durant aquesta fase s'han materialitzat els requisits seguint tant les decisions de disseny com la planificació estipulada. Existeixen funcionalitats no considerades durant la planificació del projecte, i d'altres que finalment s'ha decidit no implementar, en aquests casos es detallen els motius dels canvis.

Per un millor enteniment d'aquesta secció, es poden obtenir més detalls de l'estructura dels components que conformen l'aplicació a la figura 13.

A continuació s'esmenten els principals aspectes desenvolupats:

- Gestió de pantalles.
- Menú d'inici de sessió.
- Menú de registre d'usuari.
- Mostrar el mapa i la posició actual de l'usuari per pantalla.
- Gestió dels marcadors: Col·locar marcador en fer clic al mapa, adjuntar informació, eliminar marcadors, em-

magatzemar informació i coordenades i editar la posició i la informació dels marcadors.

- Implementació del mode de simulació de rutes.
- Realització de millores en l'aspecte visual.

7.1 Gestió de pantalles

En React Native la gestió de les diferents pantalles de l'aplicació funciona com una pila. Quan iniciem l'aplicació, la pantalla que establim com a principal s'afegeix com el primer element de la pila. A mesura que es navega per l'aplicació i es visiten noves pantalles, aquestes es van afegint a la part superior de la pila i quan es vol tornar a la pantalla anterior, la pantalla actual s'elimina de la pila i es carrega l'últim element d'aquesta, que serà la darrera pantalla visitada, i que romandrà en el mateix estat en què estava l'últim cop que es va visitar.

L'aplicatiu consta de sis pantalles diferents que l'usuari podrà visitar en realitzar una determinada acció i que corresponen als principals components desenvolupats per l'aplicatiu: *Login* (Inici de sessió), *Signin* (registre d'usuari), *Mapa* (mapa principal), *Formulari* (formulari de creació i edició de marcadors), *Simulacio* (mode simulació) i *InfoMarcador* (pantalla d'informació del marcador per al mode simulació).

Com que la connexió amb l'*API* que gestionarà els usuaris i les rutes forma part del futur del projecte, inicialment no es va considerar prioritari implementar interfícies d'inici de sessió i registre d'usuari. Finalment, s'han acabat implementant per tal de facilitar la integració de l'*API* amb el projecte quan aquesta estigui disponible, i la creació d'un menú principal ha passat a formar part del futur del projecte, pel fet que inicialment aquest aplicatiu està destinat principalment a la creació de rutes i no suposaria cap benefici implementar un menú principal.

Per implementar el canvi de pantalles s'han utilitzat els components *createNativeStackNavigator* i *NavigationContainer* de la llibreria *react-navigation* [6]. S'ha creat el component *AppNavigator* que retorna el component *NavigatorContainer*. Aquest component conté una instància del component *Screen* per a cadascuna de les sis pantalles que conformen l'aplicació i està configurat perquè consideri el component *Login* com la pantalla inicial de l'aplicatiu. Aquesta estructura permet implementar la pila de pantalles de l'aplicació.

El punt d'entrada de l'aplicatiu és el component *App*. Aquest s'encarrega principalment de cridar a la resta de components perquè cadascú realitzi la seva funció. En el cas que ens ocupa, en comptes de renderitzar els components directament invoca al component *AppNavigator* que gestionarà les transicions entre pantalles.

7.2 Menú d'inici de sessió

Es tracta de la pantalla inicial de l'aplicatiu. Pregunta l'usuari i la contrasenya a l'usuari prèviament enregistrat.



Fig. 1: Interfície d'inici de sessió

En introduir les dades i polsar sobre el botó *Entrar*, es crida a la funció *login*. Aquesta funció s'encarrega de realitzar una validació de les dades introduïdes per l'usuari mitjançant Javascript. Comprova que les dades introduïdes s'ajusten al format estipulat abans que siguin enviades a la *API*. Un cop s'ha validat el format de les dades, aquestes queden llestes per poder ser enviades a la *API* per comprovar que l'usuari existeix i que les dades són correctes. Finalment, si l'inici de sessió resultés erroni, s'informarà l'usuari del motiu pel qual no pot iniciar sessió i si l'inici de sessió és correcte, es redirigeix a l'usuari a la pantalla *Mapa*.

7.3 Menú de registre d'usuari

A la part inferior del formulari d'inici de sessió es mostra un text on es proposa a l'usuari enregistrar-se en el cas que encara no ho hagi fet. Si es polsa sobre la paraula *Enregistrar-se*, es crida a la funció *signin*, que redirigeix a l'usuari cap a la pantalla *Signin*.



Fig. 2: Interfície de registre d'usuari

Aquesta pantalla sol·licita a l'usuari la informació necessària per a enregistrar-se al sistema; Nom, cognom, email, telèfon, organització, nom d'usuari i contrasenya. En prémer el botó *Enregistrar-se* es crida a la funció *signin*, i tal com es fa amb el formulari d'inici de sessió, es valida el correcte format de les dades, per ser enviades posteriorment

a l'*API*. Si el registre és correcte, apareixerà un *pop-up* notificant a l'usuari, i se'l redirigirà a la pantalla *Login* quan aquest es tanqui, si l'enregistrament resulta erroni, es mostrarà un *pop-up* informant del motiu.

7.4 Visualització del mapa i geolocalització

Un cop l'usuari hagi iniciat sessió, ja podrà visualitzar el mapa principal de l'aplicació i la seva posició actual sobre aquest.

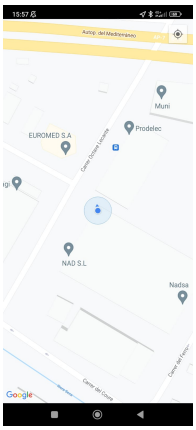


Fig. 3: Mapa inicial

Per tal de mostrar el mapa per pantalla i que aquest mostri inicialment la zona on es troba l'usuari, s'han fet servir dos elements principals, el component *MapView* de la llibreria *react-native-maps* i l'*API* que ens proporciona la llibreria *react-native-geolocation-service* per obtenir la posició inicial del dispositiu, per tant, en invocar-se el component *Mapa* es crida a la funció *location* que fa una crida a aquesta *API* i les dades de geolocalització obtingudes es passen al component *mapview* com a *prop* perquè aquest renderitzi el mapa mostrant la zona actual del dispositiu. A més es visualitza la posició i la direcció del dispositiu al centre del mapa. Per tal de poder gaudir d'aquesta funcionalitat, l'usuari ha d'acceptar la sol·licitud de permisos de geolocalització requerits per l'aplicació.

Durant la implementació d'aquest bloc van sorgir problemes a l'hora d'obtenir la zona del mapa que es vol enfocar abans de renderitzar el mapa. Aquest fet provocava que es mostrés una visualització general del mapa global en comptes de centrar el zoom sobre la zona on es troba el dispositiu. Finalment, es va solucionar cridant a la funció *location* al constructor del component mapa i, a més, no es renderitza el mapa fins que l'*API* retorna la situació geogràfica del dispositiu, d'aquesta manera obtenim les coordenades del dispositiu abans de la renderització del component *MapView*.

7.5 Gestió dels marcadors

Al llarg d'aquesta secció es detalla el procés d'implementació de les diferents funcionalitats associades als marcadors: creació de marcadors, associació de dades als marcadors, edició de les dades adjuntes, edició de la posició dels marcadors i eliminació de marcadors. El conjunt de funcionalitats associades als marcadors es troben encapsulades al component *Mapa*, excepte el formulari per adjuntar-los dades, que compta amb el seu propi component anomenat *Formulari*.

7.5.1 Creació de marcadors

L'estat del component *Mapa* guarda els següents elements relacionats amb la gestió dels marcadors:

- Una llista anomenada *markers*, que inicialment és buida, on es guarda un objecte per cada marcador que es crea. Aquesta llista es recorre contínuament i renderitza un marcador, utilitzant el component *Marker* de la llibreria *react-native-maps*, per cada objecte de la llista.
- Dues variables enteres, *lastIndex* i *markerToEditIndex*, que guarden, respectivament, l'índex de l'últim marcador creat i l'índex del marcador que ha polsat l'usuari en el cas que hagi polsat algun. Inicialment, guarden el valor -1.

Quan l'usuari polsa sobre un punt del mapa es llença l'event *onPress* del component *MapView*, que crea un nou objecte a *markers* amb les següents propietats:

- *position*: guarda les coordenades del punt polsat per l'usuari.
- *images*, *name* i *info*: inicialment buides. S'utilitzen per emmagatzemar la informació associada al marcador introduïda per l'usuari.
- *created*: propietat de tipus *boolean* que serveix per saber si la creació del marcador ja ha estat confirmada o no. Ens permet saber si l'usuari ha polsat sobre un marcador o sobre el mapa. Inicialment, guarda el valor *false*.

També incrementem el valor de *lastIndex* una unitat i cridem a la funció *form*, que porta a l'usuari a la pantalla *Formulari* per tal de crear un marcador i adjuntar-li informació.



Fig. 4: Formulari de creació de marcador

El component *Formulari* rep com a propietats les funcions *addMarker* i *deleteMarker* que s'usen per afegir i esborrar

marcadors de la llista *markers* de l'estat del component *Mapa*. La funció *cancelMarkerCreation* que cancel·larà la creació d'un nou marcador i l'esborrarà de la llista *markers*. També rep les propietats *markerToEditIndex* i *lastIndex*, i per últim la llista de marcadors.

Aquest formulari permet introduir un nom, informació i imatges associades al punt d'interès arqueològic que representa el marcador. Quan l'usuari prem el botó *Adjuntar imatges* es mostra un *pop-up* que li permet escollir si vol pujar imatges emmagatzemades localment, o bé utilitzar la càmera del dispositiu per a prendre una fotografia. L'usuari ha d'acceptar la sol·licitud de permisos de l'aplicatiu per accedir a la galeria d'imatges i/o a la càmera del dispositiu. Per implementar aquesta funcionalitat es fa ús de la llibreria *react-native-image-picker* [7].

L'efecte *pop-up* s'aconsegueix amb el component *Modal*, natiu de React Native, que mostra els components que encapsula quan la variable de control, *modalVisible* en el cas que ens ocupa, pren el valor *true*.

Quan l'usuari introdueix aquestes dades, s'emmagatzemen a les propietats *name*, *info* i *images* de l'estat del component *Formulari*. En el cas de les imatges, s'emmagatzema a la llista *images* la codificació en *base64* de cada imatge seleccionada per l'usuari, i es mostren al formulari fent ús del component natiu *Image*. Les imatges adjuntes es poden eliminar si es polsa sobre la creu que apareix a l'esquerra de cada imatge. Aquesta acció provoca una crida a la funció *deleteImage* que elimina de la llista *images* la imatge desitjada, deixant-la així de mostrar al formulari.



Fig. 5: Formulari d'edició de marcador

El formulari conté dos botons: *Afegir* i *Cancel·lar*. Si l'usuari omple totes les dades que sol·licita el formulari i prem el botó *Afegir*, s'utilitza la funció *addMarker* rebuda com a propietat per a traspassar els valors de *name*, *info* i *images* de l'estat de *Formulari* a les propietats *images*, *name* i *info* de *markers*, la propietat *created* pren el valor *true* i es redirigeix a l'usuari de nou a la pantalla *Mapa*, on ja pot visualitzar el marcador que acaba de crear. En el cas que no informi totes les dades sol·licitades i premi *Afegir* se li notifica que ha d'omplir tots els camps i el formulari no es tanca perquè l'usuari pugui continuar omplint els camps. Si l'usuari prem el botó *Cancel·lar*, es tanca el formulari, s'elimina el marcador recentment creat de *markers* i la variable *lastIndex* es decrementa perquè apunti a l'últim marcador creat.

7.5.2 Edició i eliminació dels marcadors

Quan *MapView* recorre *markers*, crea una instància del component *Marker* per cada objecte que troba a la llista i els associa un índex corresponent a la seva posició dins la llista *markers*. Quan l'usuari polsa un marcador ja creat es llença l'event *onPress* corresponent a la instància del marcador polsat, que guarda l'índex associat al marcador en qüestió dins la variable *markerToEditIndex* i redirigeix a l'usuari a la pantalla *Formulari*. El formulari que es mostra en aquest cas mostra elements addicionals a causa del fet que no s'està creant un nou marcador, sinó editant un d'existent.

La versió del formulari a mostrar es controla amb la propietat *created* de l'últim marcador de *markers*. Si estem creant un nou marcador, *created* conté el valor *false* i si es polsa sobre un marcador ja creat, conté *true*. Aquest formulari mostra el nom, la informació i les imatges adjuntes al marcador, que l'usuari pot editar. El formulari conté tres botons que desencadenaran tres accions diferents: la icona d'una paperera, extreta de la llibreria *react-native-vector-icons* [10], que, en prémer-la, crida a la funció *deleteMarker* per tal d'eliminar el marcador de *markers* i decrementar *lastIndex* perquè continuï guardant l'índex de l'últim element de *markers*, el botó *Editar* que, de la mateixa forma que es feia quan es creava un marcador, guarda les noves dades introduïdes per l'usuari a les propietats *images*, *name* i *info* del marcador en qüestió dins de *markers* i el botó *Cancel·lar* que avorta l'edició del marcador. Totes les accions provoquen el tancament del formulari i el reencaminament cap a la pantalla *Mapa*.

7.5.3 Edició de la posició dels marcadors

L'usuari pot modificar la posició d'un marcador fent una polsada llarga sobre aquest i arrossegant-lo sobre una nova posició. Això és possible gràcies al fet que les instàncies del component *Marker* creades tenen activa la propietat *draggable*. Quan es realitza aquesta acció s'activa l'event *onDrag* del marcador que es manté polsat, que crida a la funció *updatePostion* per tal de modificar la propietat *position* del marcador en qüestió de la llista *markers*, emmagatzemant les noves coordenades on se situa.

7.6 Mode simulació

En confirmar-se la creació d'un nou marcador, o bé en eliminar-se un marcador, es comprova que la llista *markers* contingui més d'un marcador, si és així es renderitzen tres botons; un botó *Simular ruta*, un botó *Enregistrar ruta* i la icona d'una paperera que, en prémer-la, elimina tots els marcadors creats.

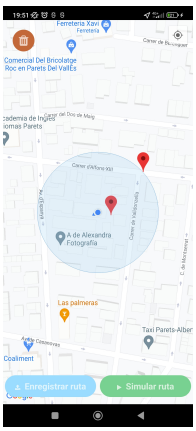


Fig. 6: Mapa amb marcadors creats

En prémer el botó *Simular ruta*, l'aplicació canvia de pantalla i entra en mode simulació. El component *Simulacio* rep la llista de marcadors i la posició on es troba l'usuari en el moment d'entrar en aquest mode. Aquesta pantalla mostra el mapa, la posició actual del dispositiu i els marcadors que s'han creat. Per saber que s'està simulant la ruta creada i facilitar la distinció amb el mode de creació de ruta, els marcadors canvien el seu color i podem visualitzar el títol *Simulació* a la part superior de la pantalla. En aquest mode l'usuari pot provar la ruta que ha estat creant abans d'enregistrar-la. L'usuari pot moure's amb el seu dispositiu i contínuament es va cridant la funció *checkProximity*, que s'encarrega de comparar la seva posició amb les coordenades dels diferents marcadors de *markers*. Quan les coordenades de l'usuari disten menys de 0,00013 graus respecte a les coordenades d'algun marcador es transita cap a la pantalla *InfoMarcador*, on es mostra la informació associada al marcador proper i un botó *Tancar*.



Fig. 7: Informació del marcador en mode simulació

Aquest marge ha estat escollit empíricament durant el desenvolupament. Visualment, sobre el mapa, es van col·locar dos marcadors separats per la distància desitjada a la qual es volia que fos necessari apropar-se per tal que es mostri la informació associada al marcador, es van extreure les coordenades dels dos marcadors i es va calcular la diferència, obtenint així el marge de 0,00013 graus, que correspondria a 14,5 metres aproximadament [13].

L'estil d'aquesta pantalla és molt similar al de la pantalla *Formulari*. La principal diferència és la utilització del component *react-native-image-swiper* per a mostrar les imatges [8]. Aquest component ens permet mostrar totes les imatges adjuntes en un mateix espai i que aquestes vagin transitant automàticament entre elles, o bé l'usuari pot passar-les manualment.

Per evitar que, si l'usuari no es mou, es mostri contínuament la informació del marcador proper sense possibilitat de tancar-la, quan la funció *checkProximity* detecta que estem a prop d'un marcador, abans de redirigir l'usuari a *InfoMarcador*, afegeix aquest marcador a la llista *doNotShowInfo*. Els marcadors que es trobin dins d'aquesta llista són exclosos de la comprovació de proximitat, per tant, no es mostra la informació associada encara que el dispositiu es trobi a prop. Quan l'usuari prem el botó *Tancar* es crida a la funció *closeInfo*, que configura la invocació de la funció *makeMarkerVisible*, rebuda com a propietat, amb un *timeout* de 10 segons. La funció *makeMarkerVisible* elimina el marcador de la llista *doNotShowInfo* provocant que es torni a incloure el marcador en qüestió a la comprovació de proximitat. D'aquesta forma es permet a l'usuari visualitzar el mapa durant un temps sense haver d'allunyar-se del punt d'interès perquè deixi de mostrar-se la informació del marcador.

Quan l'usuari desitgi deixar de provar la seva ruta i continuar amb la creació de nous marcadors, pot prémer el botó de retorn del seu dispositiu mòbil o bé el que apareix a la cantonada superior esquerra de la pantalla, i serà redirigit a la pantalla *Mapa*.

7.7 Enregistrar ruta

Seguint la mateixa lògica utilitzada per mostrar el botó *Simular ruta*, quan s'ha creat més d'un marcador també apareix el botó *Enregistrar ruta*. En prémer aquest botó es crida a la funció *register*, que mitjançant la funció *fetch*, enviarà a l'*API* la llista *markers* per tal d'emmagatzemar-la al servidor. Si l'operació finalitza satisfactòriament, es mostra un *pop-up* informant l'usuari, i en tancar-lo el redirigeix al menú d'inici de sessió.

8 Test

Cada nova funcionalitat ha sigut testejada contínuament durant tot el procés d'implementació. Aquest tipus de testatge està inclòs a la fase d'implementació i, per tant, el temps destinat a aquesta tasca està implícit a la planificació temporal d'aquesta fase. Per altra banda, s'ha considerat una fase exclusivament destinada a la realització de proves que ha tingut lloc posteriorment a la finalització de la fase d'implementació.

La fase de test de l'aplicació ha constat de dues etapes; durant la primera etapa, un cop construïda l'aplicació, s'ha verificat el seu correcte funcionament pels casos d'ús contemplats per part d'un usuari coneixedor del funcionament de l'aplicació. Un cop realitzat el testatge, s'ha avançat a una segona fase destinada a la correcció d'errors.

Pel que fa a les funcionalitats relacionades amb la gestió dels marcadors es va detectar que, en algunes ocasions, en cancel·lar la creació del marcador, aquest es creava igualment encara que no se li hagi adjuntat cap informació. També s'ha detectat que en intentar crear un marcador sense adjuntar-li informació, apareixia el missatge d'advertència informant l'usuari que ha d'omplir tots els camps, però es tancava el formulari i el marcador es creava igualment. En últim lloc, alguns cops, en cancel·lar l'edició d'un marcador, s'eliminava algun marcador de la llista *markers*. Tots els errors sorgits han estat relacionats amb la gestió de marcadors. Ha sigut l'aspecte més problemàtic del procés de desenvolupament a causa del fet que implementa la lògica més complexa de l'aplicació, així com continues transicions entre les pantalles *Mapa* i *Formulari*, i calia mantenir la coherència i persistència de l'estat del component *Mapa*

Les proves del mode simulació s'han realitzat utilitzant un dispositiu real per a verificar el correcte funcionament de la geolocalització i de les comprovacions de proximitat amb els marcadors en un entorn no virtual. Es va crear una ruta fictícia i es va sortir a l'exterior per tal de seguir-la. Els resultats obtinguts van ser satisfactoris. El marge de 0,00013 graus ha resultat convenient en els casos provats, encara que romandrà en observació, ja que es poden donar casos en què sigui convenient una major aproximació al marcador, o bé que sorgeixi algun problema amb la precisió de les coordenades obtingudes en situacions de mala connexió que afectin el correcte funcionament de l'aplicatiu

9 CONCLUSIONS

Aquesta secció està enfocada a elaborar una anàlisi del treball dut a terme durant el projecte, podent extreure conclusions sobre el nivell d'èxit de la planificació i la implementació dels requisits. Durant aquesta etapa s'elaborarà el document final del projecte per tal de ser lliurat, i finalment es durà a terme una presentació del nostre producte.

Durant el plantejament del projecte es va considerar el desenvolupament de l'aplicatiu en paral·lel per les plataformes Android i iOS, però es va desestimar la idea a causa dels obstacles que suposa la compilació del codi en iOS, principalment perquè ens veiem obligats a fer ús d'un ordinador que utilitzi el sistema operatiu macOS, per tant, l'aplicatiu no ha estat provat en la plataforma iOS. Malgrat això, el punt fort de React Native és la possibilitat de programar en Javascript i compilar el codi en llenguatge natiu d'ambdues plataformes, per tant gran part del codi podrà ser aprofitada pel desenvolupament per iOS. Tots els components externs usats són compatibles amb iOS, excepte *react-native-image-swiper* que els seus desenvolupadors no proporcionen informació al respecte a la seva pàgina oficial. Caldrà un procés d'instal·lació de dependències de les diferents llibreries de components, que es podrà realitzar de manera àgil seguint les instruccions proporcionades pels desenvolupadors de cada llibreria, que es poden trobar a les seves respectives pàgines web, un procés de familiarització amb Cocoapods i Xcode, afegir al fitxer *info.plist* les configuracions requerides per cada component, i finalment revisar el codi per detectar punts no compatibles amb dispositius iOS i elaborar fragments de codi alternatius per assegurar la màxima compatibilitat.

També es va esmentar la possibilitat de substituir la llibreria *react-native-community/geolocation* per *react-native-geolocation-service* basant-nos en les recomanacions de Google. Tot i que en un principi es va configurar el projecte per utilitzar *react-native-community/geolocation*, finalment es va migrar a *react-native-geolocation-service*, pel fet que s'utilitza d'una manera molt similar i no suposava grans canvis en l'estructura del codi, podent estalviar així el sorgiment de possibles problemes.

Inicialment es va considerar la utilització del component *Modal*, natiu de *React Native*, per a mostrar els formularis de creació i edició dels marcadors, i la informació del marcador en el mode simulació. Finalment, s'ha prescindit d'aquest component per aquestes funcionalitats, i aquesta informació es mostra en pantalles diferents en comptes d'en *pop-ups*. Aquest canvi ve motivat per les dificultats trobades en la implementació d'un *pop-up* amb una vista *scrollview*.

La planificació temporal de les tasques ha patit alguns canvis respecte a la proposada inicialment. El motiu principal ha estat l'increment en el temps d'implementació de la lògica en la gestió dels marcadors. S'ha suprimit el menú principal, principalment perquè actualment comptaria amb una sola opció, i s'han afegit les interfícies d'inici de sessió i de registre d'usuari.

10 TREBALL FUTUR

Per concloure el document del projecte, s'ha considerat convenient dedicar una secció per mencionar tant possibles millores en l'aplicatiu, com noves funcionalitats i prestacions interessants d'afegir en futures versions.

Per aconseguir la completa funcionalitat de l'aplicació, el següent pas a realitzar és la implementació de l'*API*. Durant la realització d'aquest projecte s'ha considerat en tot moment la futura connexió de l'aplicatiu amb una *API*, per tant, s'ha estructurat el codi per tal d'afavorir la inclusió d'aquest nou element. S'han deixat preparades funcions per enviar les dades relacionades amb les rutes, el registre de nous usuaris i la comprovació de l'existència d'un determinat usuari, d'aquesta forma, quan l'*API* estigui implementada, només haurem d'especificar la seva direcció i fer petits ajustaments.

En aquest punt, l'aplicatiu només és completament compatible amb la plataforma Android a causa dels impediments que suposa la compilació del codi per iOS. El sistema iOS suposa aproximadament el 30% del mercat de dispositius mòbils a tot el món [9], si es desitja desenvolupar un producte competitiu que pugui arribar al major nombre de persones possible, aquest producte ha de comptar estrictament amb una versió compatible amb iOS, per tant, seria un aspecte interessant poder comptar amb

un ordinador que utilitzi macOS per procedir amb la realització dels canvis necessaris, esmentats en l'anterior secció, per aconseguir la completa compatibilitat i procedir amb la compilació del codi per aquesta plataforma.

Un aspecte interessant a considerar per tal de millorar el producte és la possibilitat d'emmagatzemar rutes localment al dispositiu, d'aquesta forma en iniciar l'aplicatiu l'usuari podria escollir entre crear una nova ruta o bé obrir una ruta creada anteriorment, permetent així modificar rutes creades anteriorment o guardar una ruta localment abans d'enregistrar-la per continuar amb la seva creació més endavant.

En arribar a l'obtenció d'un producte completament consistent, seria necessària una revisió del codi en termes d'estil de programació, així com considerar la migració de *class components* a *functional components*. La manca d'experiència amb *React Native* ha suposat dificultats a l'hora de codificar solucions per les tasques del projecte, que podrien millorar en termes d'optimització, reutilització de codi i lògica d'algorismes.

REFERÈNCIES

[1] "React native · learn once, write anywhere," React Native RSS [En línia]. Disponible a: https://reactnative.dev/. Consultat: 09-Mar-2022.

[2] "React – una biblioteca de javascript para construir interfaces de usuario," – Una biblioteca de JavaScript para construir interfaces de usuario [En línia]. Disponible a: https://es.reactjs.org/. Consultat: 09-Mar-2022.

[3] "React-native-maps/react-native-maps: React native Mapview component for IOS + android," GitHub [En línia]. Disponible a: https://github.com/react-native-maps/react-native-maps. Consultat: 23-Mar-2022.

[4] "React-native-geolocation-service," npm. [En línia]. Disponible a: https://www.npmjs.com/package/react-native-geolocation-service. Consultat: 24-Mar-2022.

[5] "@react-native-community/geolocation," npm. [En línia]. Disponible a: https://www.npmjs.com/package/@react-native-community/geolocation. Consultat: 26-Jun-2022.

[6] "React navigation: React navigation," React Navigation RSS. [En línia]. Disponible a: https://reactnavigation.org/. Consultat: 20-Mai-2022.

[7] "React-native-image-picker," npm. [En línia]. Disponible a: https://www.npmjs.com/package/react-native-image-picker. Consultat: 20-Mai-2022.

[8] "React-native-image-swiper," npm. [En línia]. Disponible a: https://www.npmjs.com/package/react-native-image-swiper. Consultat: 20-Mai-2022.

[9] "Mobile Operating System Market Share Worldwide," StatCounter Global Stats. [En línia]. Disponible a: https://gs.statcounter.com/os-market-share/mobile/worldwide. Consultat: 21-Mai-2022.

[10] Oblador, "Oblador/react-native-vector-icons: Customizable icons for react native with support for image source and full styling." GitHub. [En línia]. Disponible a: https://github.com/oblador/react-native-vector-icons. Consultat: 10-Jun-2022.

[11] "Qué Es Kanban: Definición, características Y Ventajas," Kanban Software for Agile Project Management. [En línia]. Disponible a: https://kanbanize.com/es/recursos-de-kanban/primeros-pasos/que-es-kanban. Consultat: 23-Jun-2022.

[12] K. Solomon, "¿Qué es Trello? La plataforma de gestión de proyectos explicada."Trello. [En línia]. Disponible a: https://blog.trello.com/es/que-es-trello. Consultat: 23-Jun-2022.

[13] "Decimal degrees," Decimal degrees - GIS Wiki — The GIS Encyclopedia. [En línia]. Disponible a: http://wiki.gis.com/wiki/index.php/Decimal_degrees. Consultat: 12-Jun-2022.

[14] "TeamGantt features," TeamGantt Features. [En línia]. Disponible a: https://www.teamgantt.com/features. Consultat: 26-Jun-2022.

[15] "Diagrams.net - free flowchart maker and diagrams online," Flowchart Maker amp; Online Diagram Software. [En línia]. Disponible a: https://app.diagrams.net/. Consultat: 26-Jun-2022.

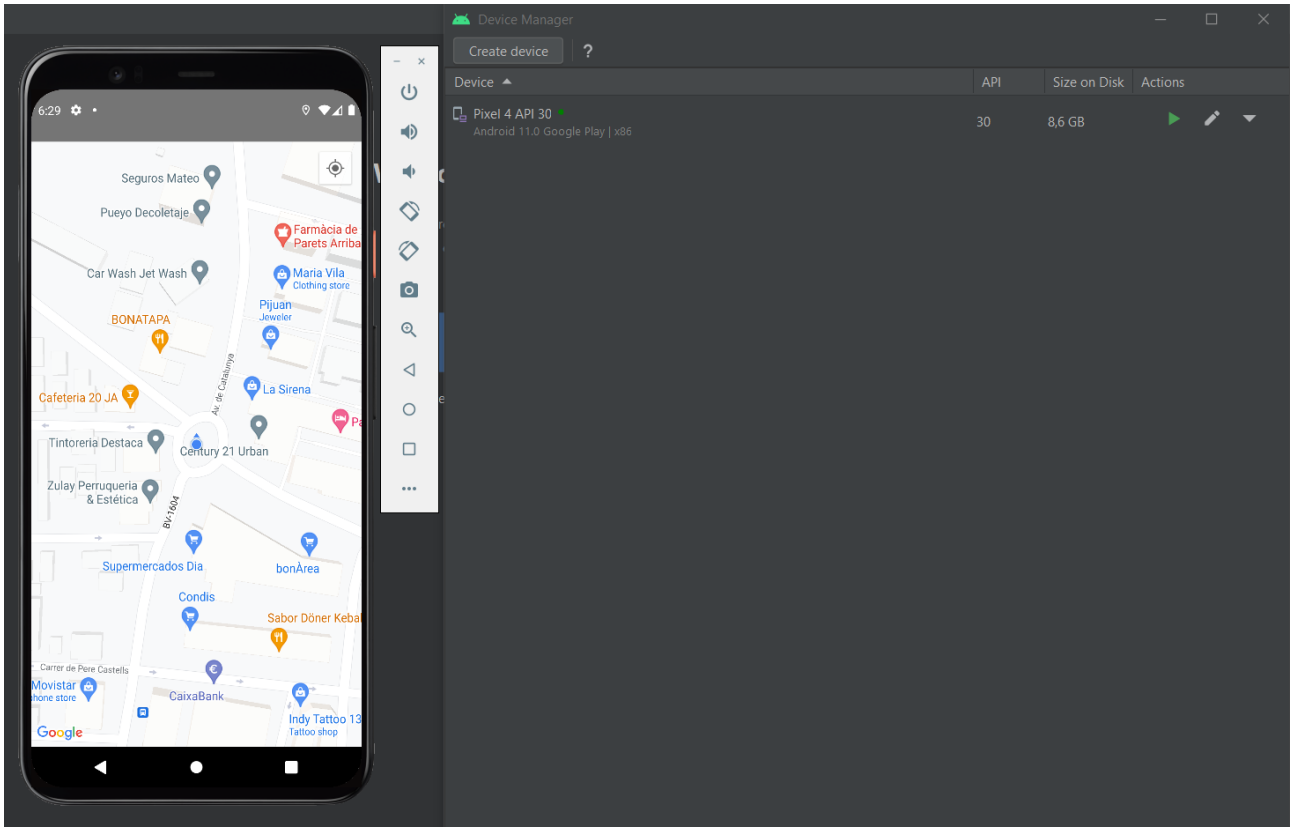


Fig. 8: Android Virtual Device

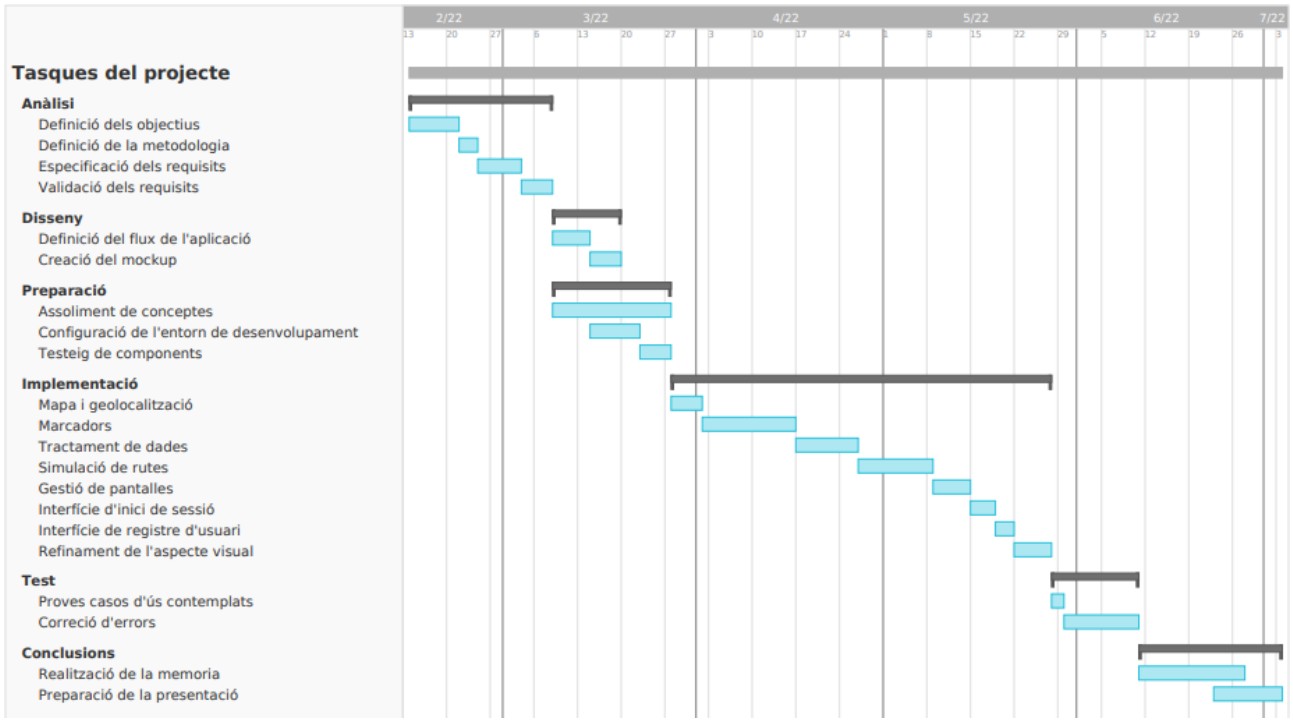


Fig. 9: Diagrama de Gantt

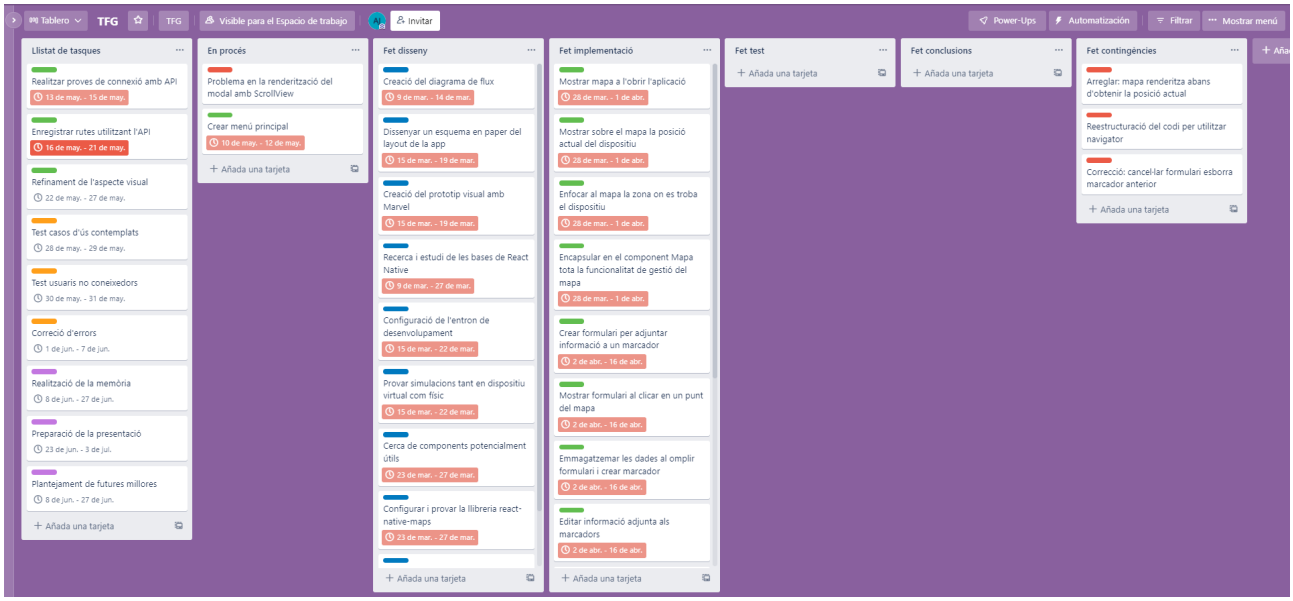


Fig. 10: Taulell Kanban del projecte

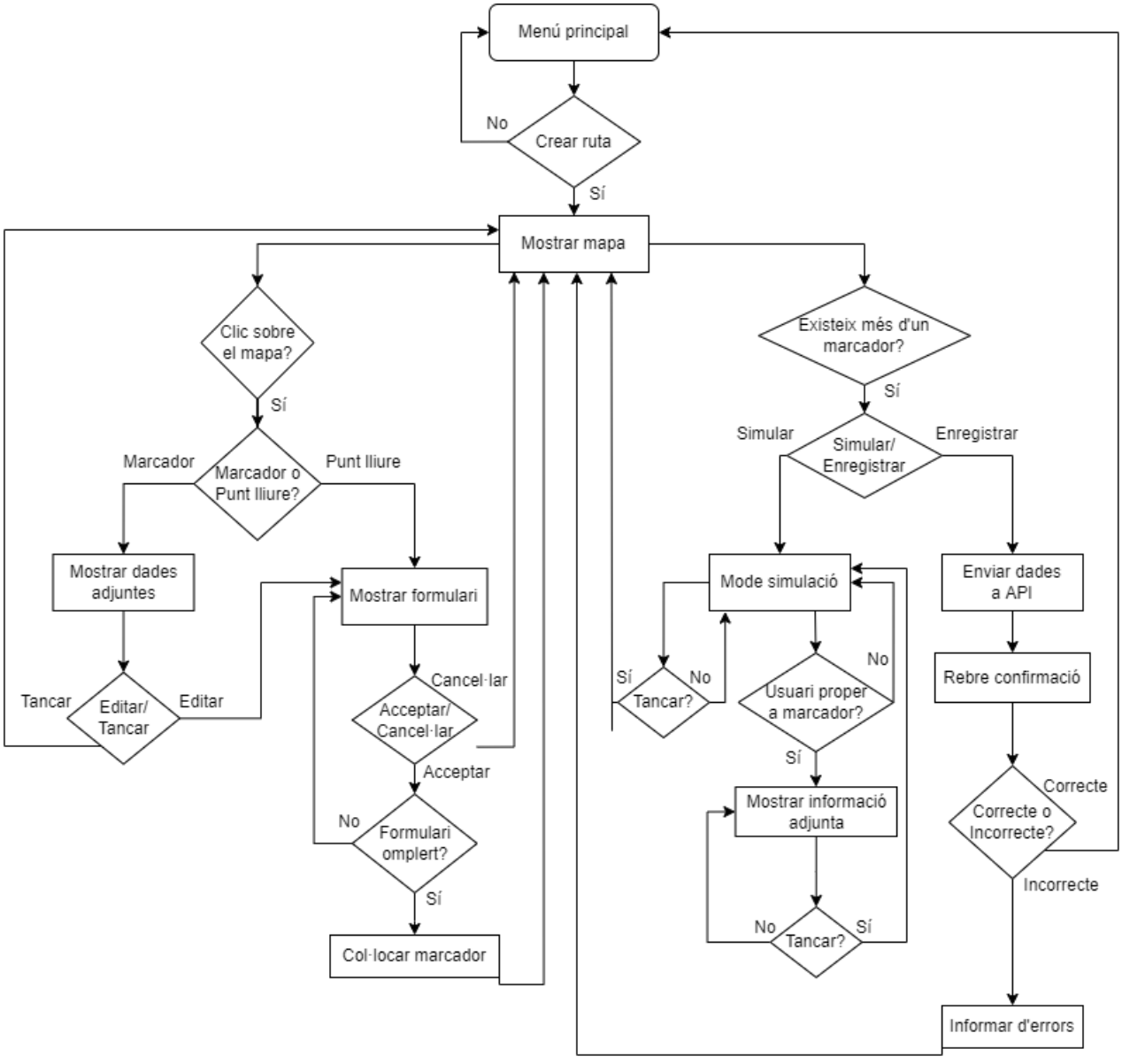


Fig. 11: Diagrama de flux

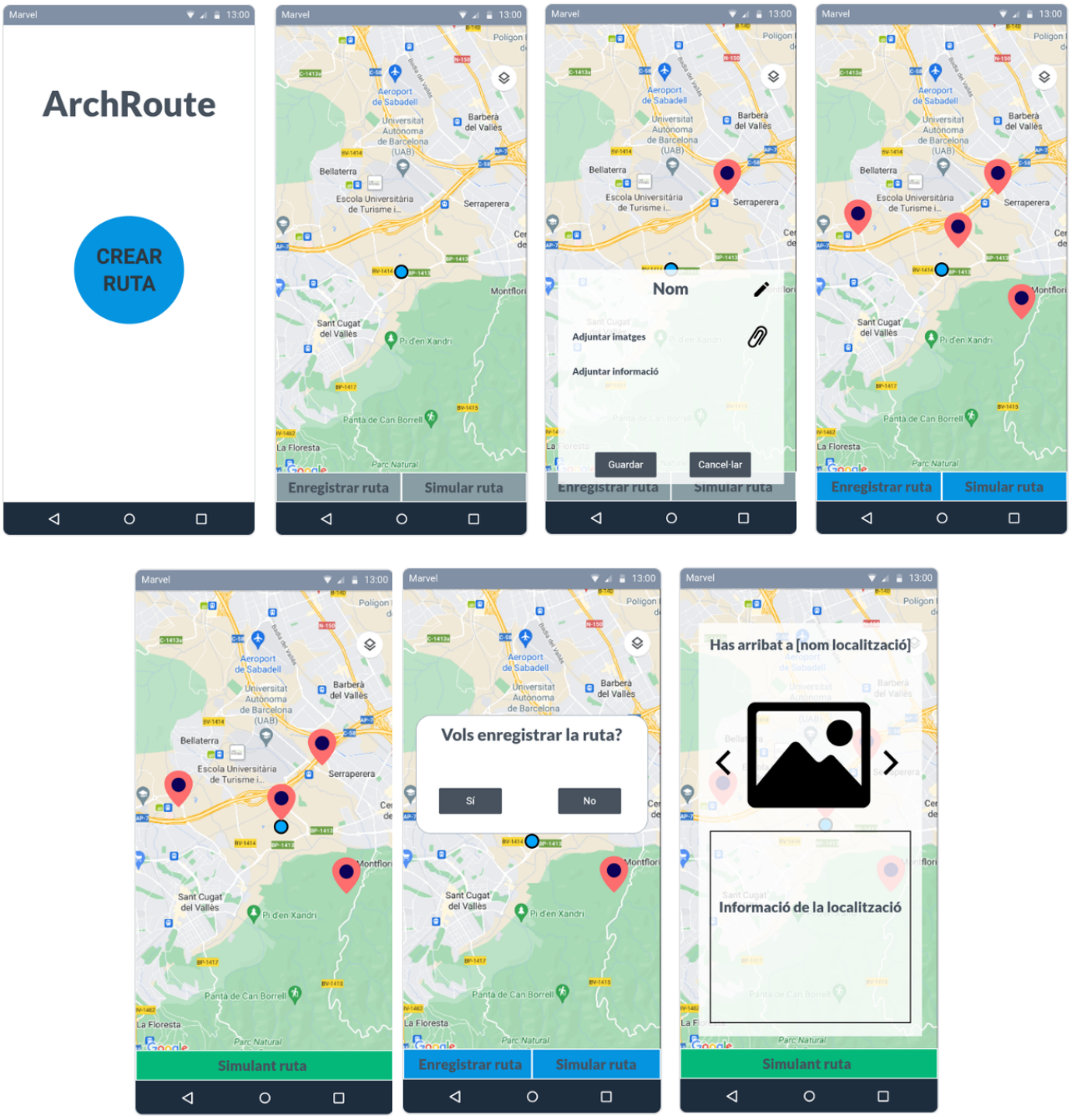


Fig. 12: Prototip visual el·laborat amb l'eina Marvel

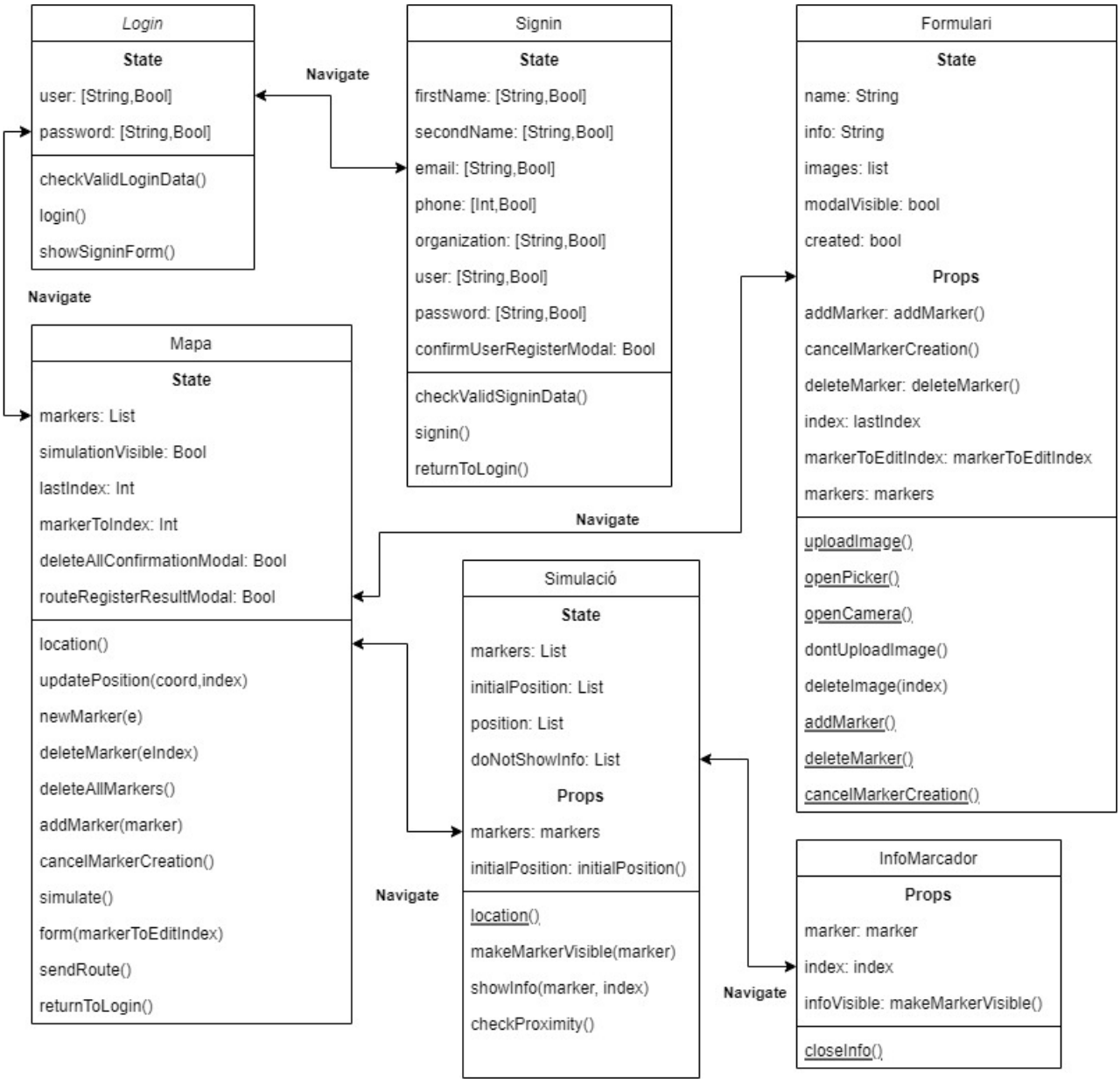


Fig. 13: Esquema dels components