Universitat Oberta de Catalunya

TAULIREPORT

Aplicació de suport a la funció directiva de l'hospital Parc Taulí

Roger Sales López

Màster Universitari en Desenvolupament d'Aplicacions per a Dispositius Mòbils

Professor consultor: Eduard Martin Lineros

Professor responsable: Carles Garrigues Olivella

Data lliurament: 31 de desembre de 2017

Fitxa del treball final

Títol del treball: TAULIREPORT – Aplicació de suport a la funció directiva de l'hospital Parc Taulí.

Nom de l'autor: Roger Sales López.

Nom del professor consultor: Eduard Martin Lineros.

Nom del professor responsable: Carles Garrigues Olivella.

Data de lliurament:

Titulació: Màster Universitari en Desenvolupament d'Aplicacions per a Dispositius Mòbils.

Idioma del treball: català.

Paraules clau: hospital, direcció, gestió.

Resum del treball: "Taulireport" és una aplicació mòbil creada per millorar la gestió hospitalària des del punt de vista del perfil directiu. A la pràctica, l'aplicació mostrarà uns indicadors escollits de les àrees d'atenció d'urgències i hospitalització. El desenvolupament d'aquesta aplicació en el marc del treball final de màster compta amb un entorn real de proves, com és la Corporació Sanitària Parc Taulí de Sabadell. Aquest hospital ha promogut la iniciativa de crear un projecte de mobilitat i el "Taulireport" n'és la primera aposta. Previ al desenvolupament, es compta amb una primera versió ja creada de l'aplicació, projecte en el qual l'autor d'aquesta memòria ja va estar implicat. La motivació del treball final, doncs, és crear una evolució d'aquesta primera versió que proposi millores tant en el disseny com en el desenvolupament, així com la incorporació de noves funcionalitats. El resultat és una aplicació híbrida programada amb el framework Ionic i exportada per a les dues principals plataformes mòbils: Android i iOS.

Abstract: "Taulireport" is a mobile application created to improve the hospital management from the point of view of the management profile. In practice, the application will show selected indicators of the areas of emergency care and hospitalization. The development of this application in the framework of the master final project has a real test environment, such as the Corporació Sanitària Parc Taulí de Sabadell. This hospital has promoted the initiative to create a mobility project and the "Taulireport" is the first bet. Prior to development, there is a first version already created of the application, a project in which the author of this memory was already involved. The motivation of the final work, then, is to create an evolution of this first version that offers improvements in both design and development, as well as the incorporation of new features. The result is a hybrid application programmed with the Ionic framework and exported to the two main mobile platforms: Android and iOS.

Índex de capítols

Índex de figures

1. Introducció

1.1. Context i justificació del treball

El resultat del treball realitzat com a projecte final del Màster de Desenvolupament d'Aplicacions per a Dispositius Mòbils és una **aplicació d'ús per a professionals de perfil directiu i de comandament que ajudi en la presa de decisions i en la gestió d'un hospital**.

El projecte s'emmarca en un context de treball real, el Consorci Corporació Sanitària Parc Taulí de Sabadell (Parc Taulí en endavant). L'autor d'aquesta memòria forma part de l'equip de professionals no assistencials que donen tot tipus de suport (administratiu, tecnològic, de gestió) als professionals assistencials que ofereixen atenció sanitària a la població.

Concretament, l'autor d'aquesta memòria gestiona la formació i la comunicació de l'institut de recerca del Parc Taulí, a més d'estar molt vinculat al departament de Sistemes d'Informació, amb el qual ha col·laborat en projectes que impliquen noves maneres d'interacció entre els professionals i les dades clíniques.

Durant el 2016, la direcció de Sistemes d'Informació va endegar un projecte de mobilitat que proposava crear la primera aplicació mòbil del Parc Taulí, "Indicadors Taulí", amb un objectiu molt concret: millorar la gestió hospitalària i ajudar en la presa de decisions. Passat l'estiu, es va estrenar la primera versió de l'aplicació, que seria provada amb un grup reduït de professionals amb càrrec de comandament.

En el moment actual, la direcció de l'hospital es planteja seguir evolucionant l'aplicació, i és en aquest punt que es proposa **incorporar aquesta evolució com a repte per al treball final del màster**. Aquestes són algunes de les raons que justifiquen l'elecció:

- És un projecte que compta amb un entorn real de proves: el mercat[1] és el propi hospital.
- El suport de la Direcció permetrà assegurar-ne la continuïtat.
- Té influència directa sobre les funcions directives i, per tant, un benefici indirecte sobre els pacients.
- Obre noves línies de treball dins del projecte de mobilitat, ja sigui millorant l'aplicació actual o ampliant el públic objectiu cap altres perfils professionals, o la població de referència (pacients).

[1] Mercat entès com el públic objectiu: "conjunt de persones que per les seves característiques exclusives [...] són compatibles i ideals per consumir el nostre producte o servei." Salmerón, Sara; Montserrat, Roger. ***Modelos de negocio y marketing basados en dispositivos móbiles***. Barcelona, 2017. Pàgina 104. Materials de l'assignatura de mateix nom del Màster de Desenvolupament d'Aplicacions per a Dispositius Mòbils.

La mobilitat als hospitals catalans

A finals de l'any 2013, la Fundació TicSalut[2], organisme depenent del Departament de Salut que impulsa el desenvolupament i ús de les TIC al sector salut, va publicar els resultats d'una enquesta realitzada a centres sanitaris i sociosanitaris públics catalans per mesurar-ne el nivell de penetració de les TIC.

Quant a mobilitat, s'extreuen les següents conclusions de l'informe:

- Hi ha hagut un **creixement en iniciatives TIC** per augmentar **eficiència** del centres.
- De tots els **dispositius per treballar** amb la història clínica electrònica, un **11% són dispositius mòbils**.
- La salut 2.0 està en alça: quasi **12.000 professionals** participen en **comunitats virtuals**.
- El professionals que treballen en hospitals i CAPS fan ús dels **telèfons intel·ligents per comunicar, gestionar i fer servir aplicacions**.

Aquesta tendència creixent va promoure el disseny, l'any 2014, del Pla Mestre de Mobilitat en Salut i Social[3], amb la creació d'una plataforma que aglutina totes les iniciatives en mobilitat, no només a Catalunya, sinó a nivell mundial.

Figura 1. Aparador d'apps i projectes de la Fundació TicSalut, que compta ja amb 226 projectes que es poden filtrar per categories: tipus, sistemes operatius, àrees d'interès, o especialitats.

[2] *Qui som* [en línia]. Barcelona: Fundació TicSalut. <http://www.ticsalut.cat/qui-som/> [Consulta: 6 oct. 2017].

[3] *Apps i projectes de mobilitat* [en línia]. Barcelona: Fundació TicSalut, 2014. <http://www.ticsalut.cat/observatori/apps/> [Consulta: 6 oct. 2017].

L'estudi d'aquest aparador posa de rellevància el fet que totes les aplicacions s'orienten cap a la informació i gestió de malalties i salut (tant per a professionals com per a pacients), així com cap a l'ajuda en la presa de decisions clíniques (professionals).

Expressat d'una altra manera, **no es coneix cap iniciativa, com a mínim en els centres catalans, que utilitzi la informació dels HIS[4] per millorar la presa de decisions des d'un punt de vista de la gestió directiva.**

Aquest tret distintiu proporciona valor afegit al projecte i alimenta la motivació per crear un producte final que satisfaci les exigències pròpies d'un projecte final de màster, tant en la forma com en el contingut.

1.2. Objectius

Tenint en compte que per a aquest projecte final es parteix d'un producte ja existent, per entendre els objectius marcats cal revisar primerament l'aplicació "Indicadors Taulí" en la seva versió actual[5].

L'objectiu principal es va establir com una aplicació per **ajudar en la presa de decisions millorant l'eficiència i l'eficàcia de la gestió directiva i de comandament.** Està dividida en quatre grans àmbits, que són els que es va acordar que requerien d'una atenció especial des del punt de vista de l'objectiu: Urgències, Hospitalització, Cirurgia i Atenció Primària.

La primera versió de "Indicadors Taulí" només va cobrir els primers blocs d'**Urgències i Hospitalització**:

- En el bloc d'Urgències, es poden consultar dades com: altes d'urgències pendents d'ingrés hospitalari (i els llits lliures disponibles a planta), pacients segons especialitats i zones d'atenció i pacients segons el circuit d'atenció (entrada, triatge, assistència, etc.).
- En el bloc d'Hospitalització, es pot consultar l'ocupació hospitalària (llits lliures, reservats, ocupats), l'estada mitjana i les altes concedides segons especialitats mèdiques.

Algunes dades, per exemple els episodis actius a Urgències, representaven una situació actual, de manera que donaven resposta a la pregunta: "què està passant ara mateix a l'hospital?". Però altres dades mostraven una evolució durant el dia, a comptar des de les 00 hores del dia corrent (per exemple, pacients segons el circuit d'atenció), o des del dia 1 del mes corrent (per exemple, les altes concedides).

[4] Inicials de *Hospital Information System*, en anglès.

[5] Es realitza un anàlisi més profund de la versió de partida de l'aplicació "Indicadors Taulí" en el capítol 2 d'aquesta memòria.

Aquesta **diferent interpretació temporal** de les dades ha aflorat oportunitats de millora en el plantejament i el disseny de l'aplicació, ja que en ocasions es podien extreure conclusions que no estaven ajustades a la realitat.

Tecnològicament, el projecte es va plantejar com una aplicació híbrida construïda amb **Framework 7**, un sistema relativament jove però amb una corba d'aprenentatge suau i compatible amb Cordova, de manera que podríem generar aplicacions per a les principals plataformes mòbils.

Si bé Framework 7 va ser útil per al llançament d'un primer prototip, actualment –i sobretot després de conèixer les possibilitats d'**Ionic** durant l'aprenentatge del Màster- afloren les oportunitats de millora per poder evolucionar l'aplicació amb un ***framework* més sòlid** i segons les propostes de l'equip de desenvolupament i de les necessitats que els propis usuaris han anat aportant.

Amb aquests antecedents, s'exposen a continuació els **6 objectius marcats per al present treball final de màster**:

1. **Redissenyar de nou l'app**, des dels prototips de baixa fidelitat fins als d'alta qualitat.
2. Incorporar **Ionic com a *framework* de desenvolupament** per acabar generant una aplicació híbrida que permeti exportar a **Android i iOS**, adaptant en cada cas els dissenys segons els llibres d'estil de cada plataforma.
3. **Ampliar les funcionalitats** actuals aprofitant les possibilitats d'Ionic: sistema de notificacions, alarmes configurables, noves àrees d'informació.
4. Crear **serveis REST** del costat *backend* que permetin fer una gestió eficaç de les peticions entre l'aplicació i la base de dades.
5. Generar una aplicació **multiplataforma** plenament funcional i crear un **mètode de distribució** entre els seus usuaris.
6. Generar una plataforma de **monitoratge** dels accessos, comportament, anàlisi i gestió d'errors.

1.3. Enfocament i mètode seguit

La necessitat d'evolucionar "Indicadors Taulí" obeeix a la manca d'experiència del Parc Taulí en termes de mobilitat. Sense referències ni referents, calia definir-ho tot per primera vegada: *framework*, tecnologia, tipus de desenvolupament, etc.

La primera versió de l'aplicació va gaudir d'una gran acollida per part de l'equip de directius amb qui es va fer la prova pilot, fet que constatava el bon camí seguit. No obstant, els coneixements adquirits durant l'estudi del Màster, juntament amb les necessitats identificades per part de tot l'equip de treball i els requeriments extrets de la prova pilot, deixaven palesa la idoneïtat de donar aquest pas.

La proposta d'incloure l'aplicació com a projecte del TFM compta amb l'aprovació i la conformitat de la direcció de Sistemes, destacant el valor afegit que aporta per al projecte el seguiment acadèmic en el marc del Màster de Desenvolupament d'Aplicacions per a Dispositius Mòbils.

Durant el període de gestació de la primera versió de l'aplicació, es va seguir una **metodologia molt aplicable a l'Scrum**: un equip petit de treball, amb uns rols molt definits (dissenyador –rol exclusiu adquirit per l'estudiant dins l'equip-, programadors web, gestor de bases de dades) es marcaven tasques i endegaven processos iteratius, al final dels quals s'analitzava el resultat i es continuava amb la següent tasca, incorporant les modificacions necessàries.

Tenint en compte que l'equip de treball seguirà sent el mateix, i que tots els rols seran igualment necessaris, s'optarà seguir amb la metodologia Scrum per a la consecució dels objectius d'aquest treball. En aquest cas, l'estudiant adquirirà un paper més predominant en adquirir gran part dels diferents rols, però algunes tasques hauran d'estar igualment delegades a l'equip (per exemple, tot el que comporti gestió de bases de dades).

Aquestes tasques delegades estaran degudament documentades i assenyalades en la planificació i el diagrama de Gantt generat.

1.4. Planificació

La planificació del projecte ve marcada per les entregues de les PAC. Sense comptar la PAC inicial, que suposa l'inici real del treball, l'inici i finalització de les PAC s'ha establert així:

- **PAC2: Disseny. Del 13 d'octubre a l'1 de novembre.**
- **PAC3: Implementació. Del 2 de novembre al 13 de desembre.**
- **PAC4: Entrega final. Del 14 de desembre al 3 de gener.**

Durant els **dies laborables**, es dedicarà una mitja de **3 hores diàries**. Els **caps de setmana i festius**, es preveu poder pujar fins a les **5 hores diàries**. Tenint en compte això, a continuació es detalla cada una d'aquestes PAC amb la temporització. Les tasques que es treballaran conjuntament amb l'equip de Sistemes d'Informació del Parc Taulí es marquen en cursiva i en gris. Així mateix, al diagrama de Gantt també estan senyalitzades en gris.

En el cas del bloc quirúrgic, es marca sencer en gris perquè es desconeix si podrà realitzar-se, en dependre de la finalització d'una aplicació web externa que s'hauria d'integrar amb aquest apartat.

PAC2: Disseny (66 hores)

- Anàlisi de requeriments (22 hores)
- Prototipatge (24 hores)

    - Baixa fidelitat (12 hores)
    - Alta fidelitat (12 hores)
- *Preparació BBDD (10 hores)*
- Documentació memòria (10 hores)

PAC3: Implementació (149 hores)

- Preparació entorn treball (9 hores)
- 1a Iteració: URGÈNCIES (25 hores)
    - Disseny (6 hores)
    - Programació (15 hores)
    - *Prova i anàlisi (4 hores)*
- 2a Iteració: HOSPITALITZACIÓ (25 hores)
    - Disseny (6 hores)
    - Programació (15 hores)
    - *Prova i anàlisi (4 hores)*
- 3a Iteració: HOME (25 hores)
    - Disseny (6 hores)
    - Programació (15 hores)
    - *Prova i anàlisi (4 hores)*
- *4a Iteració: CIRURGIA (25 hores)*
    - *Disseny (6 hores)*
    - *Programació*
    - *Prova i anàlisi (4 hores)*
- 5a Iteració: NOTIFICACIONS I ALARMES (25 hores)
    - Disseny (6 hores)
    - Programació (15 hores)
    - *Prova i anàlisi (4 hores)*
- Documentació memòria (15 hores)

PAC4: ENTREGA FINAL (50 hores)

- Resolució d'incidències (8 hores)
- Integració i exportació (8 hores)
- Plataforma de monitoratge (10 hores)
- Presentació en vídeo (18 hores)
- Documentació memòria (6 hores)

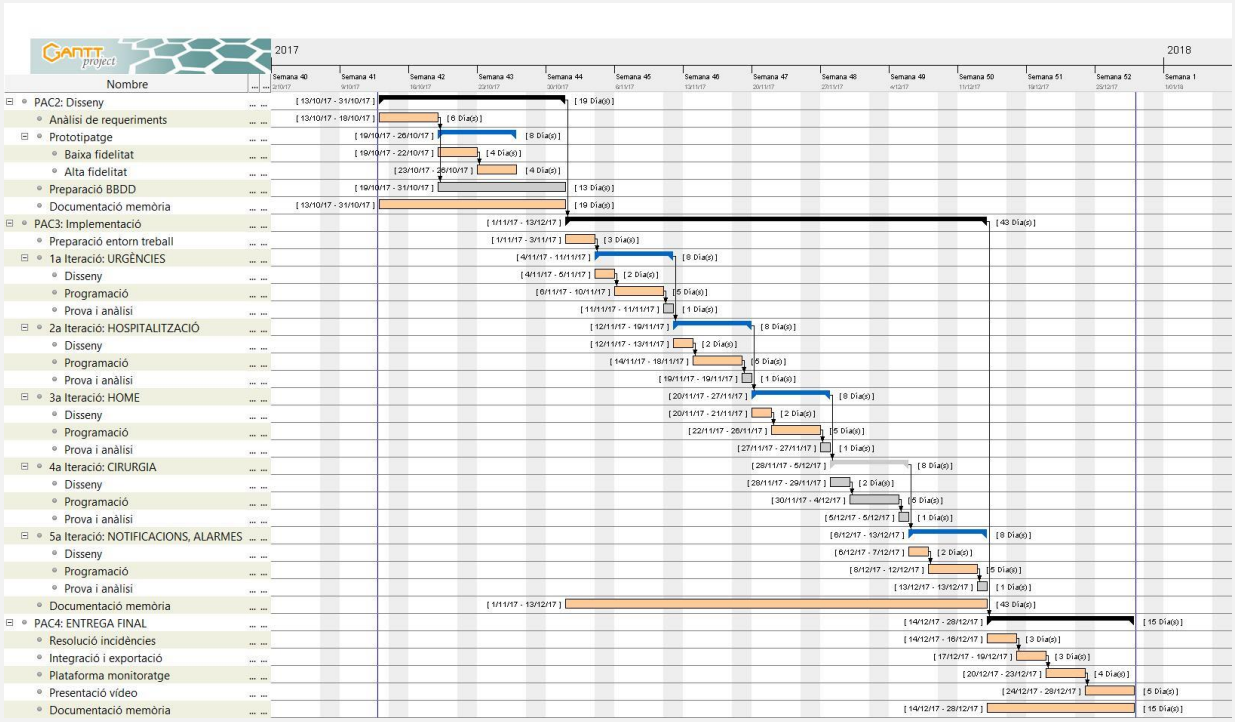


Figura 2: Diagrama de Gantt del projecte, amb les tasques, temporització i dependències.

1.5. Sumari de productes obtinguts

En finalitzar el TFM, s'obtindrà un projecte Ionic que serà exportat en les versions per a les principals plataformes mòbils: **Android i iOS**. Ambdues aplicacions seran estilitzades individualment per complir amb el llibre d'estil de cada plataforma.

2. "Indicadors Taulí": l'origen

La concepció de la primera versió de l'aplicació va seguir el desenvolupament típic d'un disseny centrat en l'usuari: anàlisi, disseny i avaluació. En aquestes fases, un equip de treball, amb un profund coneixement del HIS i vinculat a la Direcció, va definir les dades de consulta més rellevants i les va dividir en quatre grans àmbits: Urgències, Hospitalització, Cirurgia i Atenció Primària.

En la part més operativa, l'equip de treball estava format per:

- **2 analistes programadors**
- **1 analista de base de dades**
- **1 dissenyador multimèdia (autor d'aquest treball)**
- **1 cap de projecte**

Seguint una metodologia propera a l'Scrum, l'analista de base de dades i la cap de projecte recollien els continguts per blocs obtinguts en la fase d'anàlisi. A mida que s'anaven perfilant els continguts i es preparava la base de dades, es definien parts concretes de l'aplicació (per exemple, la pantalla "Per ingressar" del bloc d'Urgències), amb el qual el dissenyador podia començar a preparar el disseny dels prototips i els analistes programaven la lògica d'aquesta part de l'aplicació.

2.1. Continguts i arquitectura

Dels quatre possibles blocs, es van arribar a concloure els dos primers, **Urgències** i **Hospitalització**, amb motiu de la seva especial rellevància per a l'objectiu de l'aplicació. Aquests dos blocs tenien el seu contingut dividit en tres apartats respectivament:

- **Urgències "Per ingressar":** mostra el nombre de pacients que hi ha actualment a Urgències (episodis actius) i quants d'aquests requereixen hospitalització (altes pendents d'ingrés), organitzats per serveis. També es mostra els llits lliures que hi ha disponibles a hospitalització, amb informació sobre el pacient del llit del costat (edat i sexe), per resoldre en una sola vista quina pot ser l'assignació més adequada.
- **Urgències "Per zones":** mostra en un gràfic interactiu quants pacients hi ha actualment a Urgències segons les zones d'atenció i els nivells de gravetat, així com una taula amb el temps mig d'espera.
- **Urgències "Per recorregut":** mostra els pacients que han passat per Urgències des de les 00 h del dia corrent i el temps mig d'espera, segons el recorregut típic del servei.
- **Hospitalització "Ocupació":** mostra el percentatge d'ocupació de l'hospital actualment (llits ocupats), llits reservats i llits lliures. També mostra aquests valors per les quatre àrees d'hospitalització principals: Adults, Pediatria-UCA, Salut Mental, Sociosanitari.
- **Hospitalització "Estada":** mostra l'estada mitjana dels pacients que porten hospitalitzats en el mes corrent i la compara amb l'estada mitjana esperable. També mostra quin percentatge de pacients està per sota de l'estada esperable i quin per sobre.
- **Hospitalització "Altes":** mostra quantes altes s'han donat des del dia 1 del mes corrent, organitzades per serveis.

Com es veurà en les captures reals, l'aplicació es va construir amb un menú lateral que contenia la **navegació principal**:

- **Visió general:** amb les principals dades dels blocs d'Urgències i Hospitalització.
- **Urgències:** amb els tres apartats relatius al bloc d'Urgències.
- **Hospitalització:** amb els tres apartats relatius al bloc d'Hospitalització.

Així mateix, per assegurar la privacitat de les dades, es van construir **dues capes de seguretat**. D'una banda, l'aplicació no accediria directament al HIS per extreure les dades, sinó que es van preparar unes **vistes especials** amb les dades que a priori se sabia que l'aplicació demanaria.

Així, independentment dels canvis que es puguin produir al HIS, només caldria modificar les vistes per garantir el correcte funcionament de l'aplicació.

D'altra banda, calia crear uns serveis web que funcionarien a mode d'API REST entre l'aplicació i les vistes. Aquests serveis en PHP utilitzarien un **usuari especial que només tindria accés a les vistes anteriors**, de manera que si un atacant pogués interceptar les credencials, mai podria accedir a la base de dades completa del HIS, garantint-ne la integritat i la seguretat.

Per acabar de garantir la seguretat, es va crear un **sistema d'inici de sessió** que atacaria directament cap al LDAP de la institució, concedint accés només a l'equip de directius que formaria part de la prova pilot de l'aplicació. L'inici de sessió es produiria sempre en obrir l'aplicació i no deixaria veure cap contingut sense unes credencials correctes.

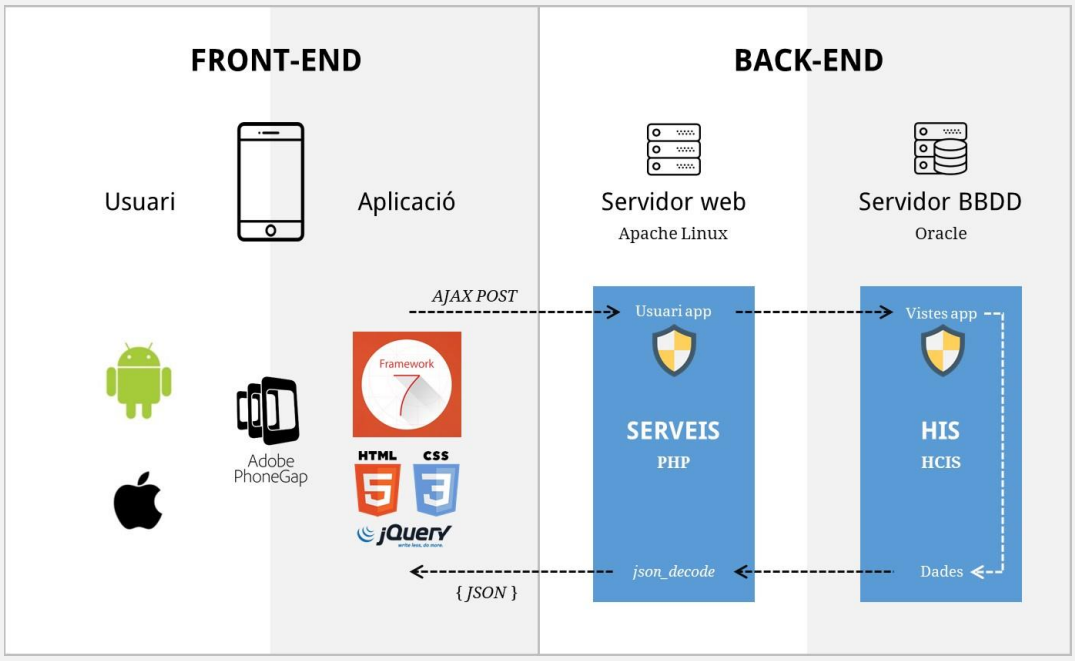


Figura 3: Arquitectura actual de l'aplicació. En el *front-end*, trobem Framework 7 i la capa de Phonegap per exportar les aplicacions híbrides en Android i iOS. En el *back-end*, trobem el servidor web amb les peticions que es fan des de l'aplicació a l'API REST, que mitjançant un usuari especial interacciona amb les vistes del HIS i retorna les dades, que són enviades cap a l'aplicació en format JSON.

Durant un temps, l'aplicació va estar funcionant en els dispositius mòbils dels directius de la prova pilot i va tenir una gran acollida per la utilitat que va aportar, en ser capaç de mostrar en temps real informació valuosa del HIS des d'un dispositiu mòbil, fet impensable amb el programari habitual de gestió de la història clínica.

2.2. Captures

Es mostren a continuació captures de pantalla reals de l'aplicació, que han de servir per entendre l'estat actual de l'aplicació i el procés evolutiu proposat en el capítol 3 d'aquesta memòria.

Figura 4: "Indicadors Taulí" inici de sessió.

Figura 5: "Indicadors Taulí" visió general.

Figura 6: "Indicadors Taulí" urgències per ingressar.

Figura 7: "Indicadors Taulí" urgències per ingressar.

Figura 8: "Indicadors Taulí" urgències per zones.

Figura 9: "Indicadors Taulí" urgències per recorregut.

Figura 10: "Indicadors Taulí" hospitalització ocupació.

Figura 11: "Indicadors Taulí" hospitalització estada.

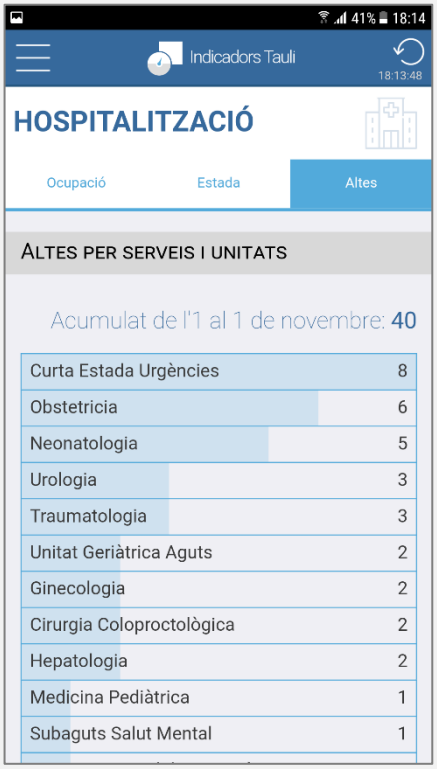


Figura 12: "Indicadors Taulí" hospitalització altes.

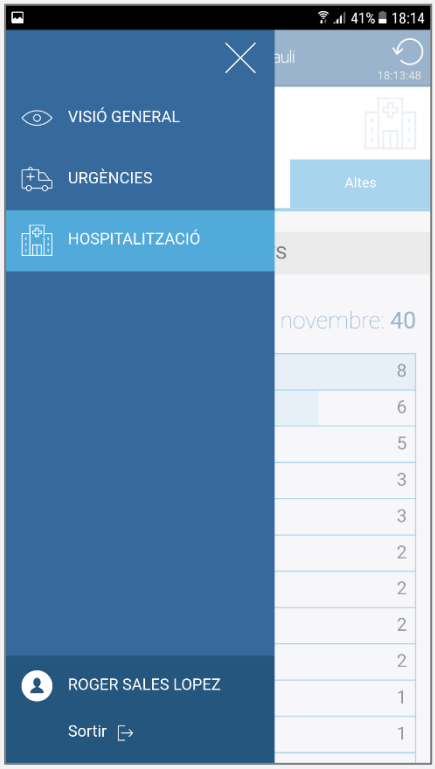


Figura 13: "Indicadors Taulí" menú lateral.

3. "TAULIREPORT": l'evolució

La nova versió de "Indicadors Taulí", si bé manté les mateixes funcionalitats que l'original, comença la seva evolució amb un canvi de nom: **"TAULIREPORT"**.

El "taulireport"[6] va ser, durant 8 anys, una revista institucional que informava sobre aspectes assistencials, organitzacionals i de salut en general. L'any 2010 es va discontinuar la revista a favor d'altres canals més efectius i econòmics, com l'actual blog intern de notícies.

3.1. Arbre de navegació

La nova versió de l'aplicació no té masses diferències a nivell de continguts sobre l'actual, però es important deixar palès l'arbre de navegació que justificarà els prototips presentats en el següent subapartat.

[6] Es pot veure un recull de tots els exemplars digitals a <http://www.tauli.cat/tauli/informacio-corporativa/sala-de-premsa/taulireport>.

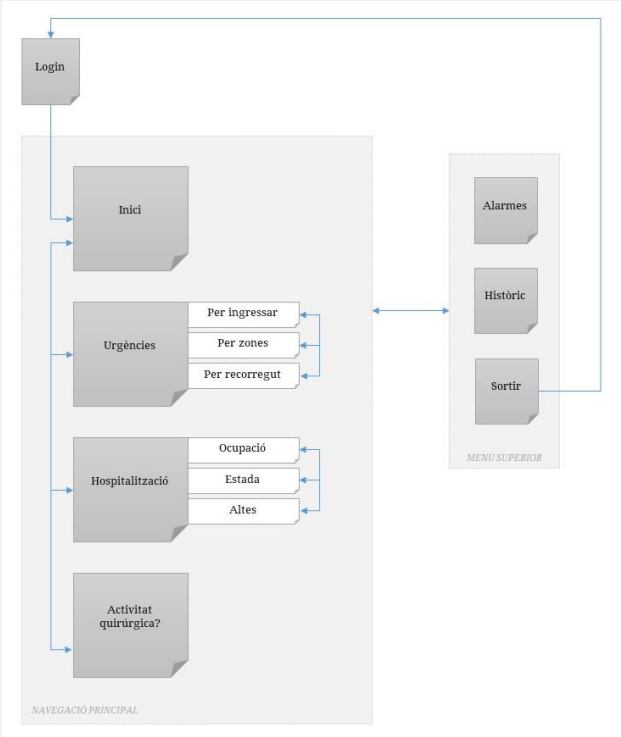


Figura 14: Arbre de navegació del "TAULIREPORT".

3.2. Prototips de baixa fidelitat

El canvi de nom ve seguit d'un canvi en l'arquitectura visual de l'aplicació. En les captures del capítol anterior, es pot veure com per **saltar d'un bloc d'informació a un altre** cal passar pel menú lateral, fet que es podia millorar si **l'accés s'ubicava permanentment a la part inferior**.

També es va veure que la **capçalera mantinguda en totes les pantalles amb el nom o logotip de l'aplicació no aportava cap valor**, a més de ser una pràctica que no està recomanada en les guies d'estils de les plataformes iOS i Android.

Un altre dels punts a millorar era el **context de la informació** aportada: calia evidenciar de manera clara la **temporització** de les dades (si són del moment actual, o a partir de certa data fins al moment actual), i la **interpretació** de les mateixes.

Com a nova aportació, es treballarà en un sistema de **notificacions *push* per avisar de certs esdeveniments (alertes)** i, si la temporització del treball ho permet, en una pàgina que reculli els **històrics de dades**. Aquests blocs transversals tindran el seu accés a la nova capçalera proposada.

Per últim, convé recordar que, degut al context d'ús, per ara "TAULIREPORT" és una aplicació plantejada per a **telèfons intel·ligents** i es prioritzaran els esforços de disseny i programació per a aquest tipus de dispositiu.

Amb aquestes premisses, s'aporten a continuació els prototips de baixa fidelitat.

Figura 15: "TAULIREPORT" inici de sessió.

Figura 16: "TAULIREPORT" inici.

Figura 17 "TAULIREPORT" urgències per ingressar.

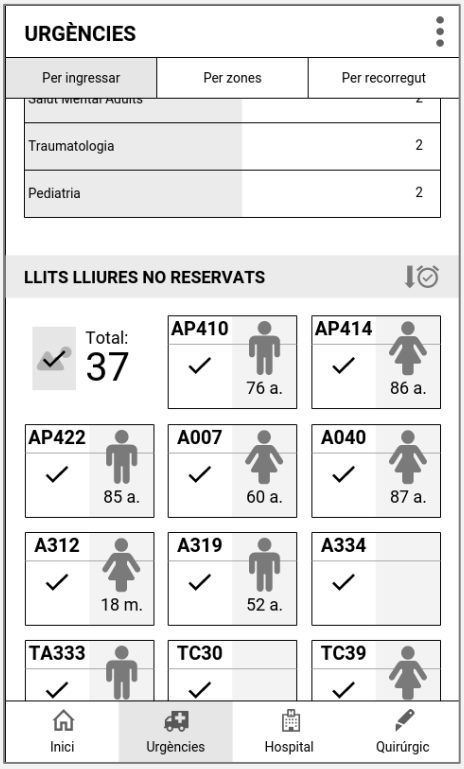


Figura 18: "TAULIREPORT" urgències per ingressar.

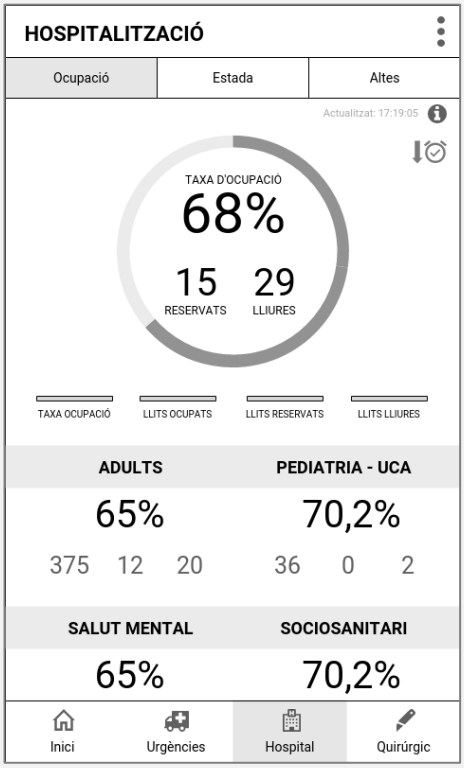


Figura 19: "TAULIREPORT" hospitalització ocupació.

Figura 20: "TAULIREPORT" accés del menú superior.

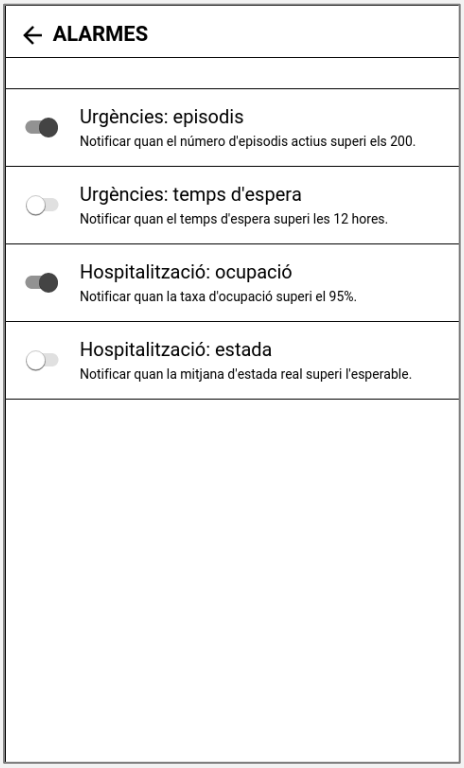


Figura 21: "TAULIREPORT" alarmes.

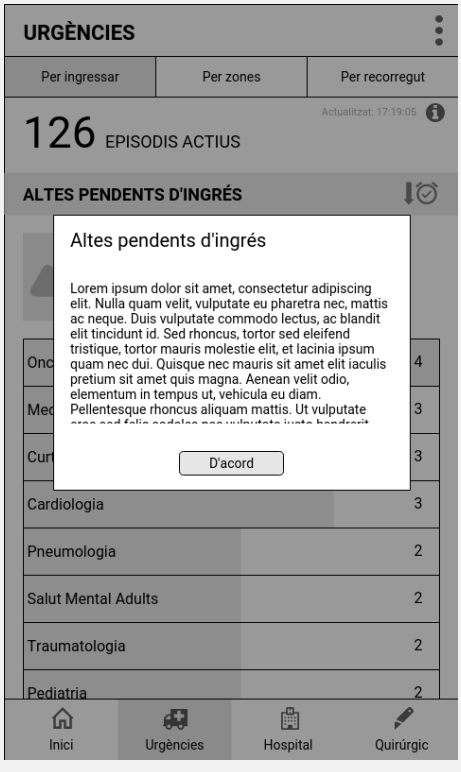


Figura 22: "TAULIREPORT" ajuda per interpretar dades

3.3. Prototips d'alta fidelitat

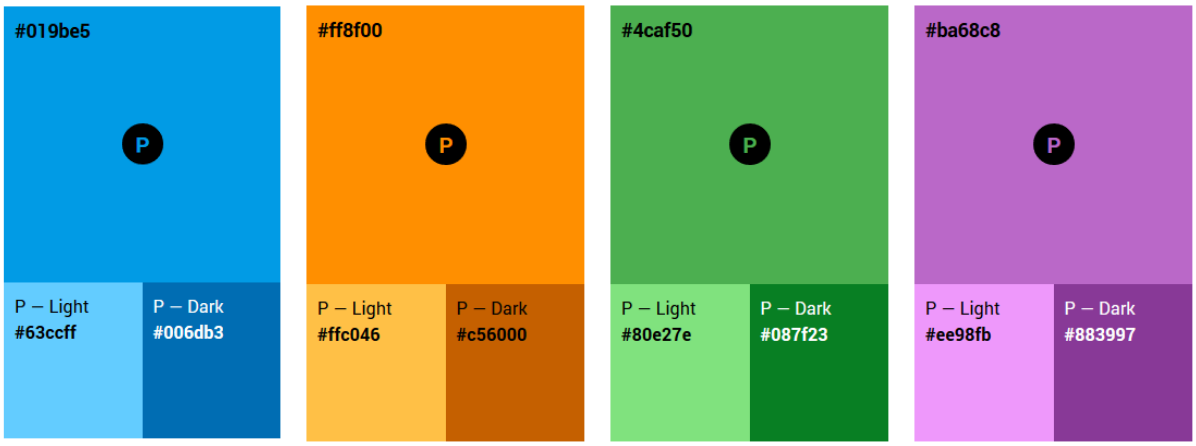
3.3.1. Identitat corporativa

Colors

Amb previsió que moltes vegades el context en què es consulta l'aplicació no és relaxat (passadissos de l'hospital, urgències), s'ha decidit identificar cadascun dels blocs principals amb diferents colors, de manera que ajudi a situar l'usuari en la interpretació de les dades.

Amb l'ajuda de l'eina Color Tool[7] de Material Design, s'ha optat pel blau com a vertebrador de l'aplicació i tres colors ben diferenciats amb els seus corresponents complementaris, que ajudaran a donar contrast a la interfície per mitjà de la seva combinació en els blocs principals.

L'eina també recomana el color del text que es superposarà al color escollit, segons si contrastarà millor en blanc o en negre.



Logotip

La marca proposada per al "TAULIREPORT" aprofita el logotip actual del Parc Taulí per fer-ne una deconstrucció i acaba resultant en un imagotip que evoca uns gràfics de dades, com a paral·lelisme del tipus d'aplicació que és.

Parc Taulí
Hospital Universitari

TAULIREPORT

[7] <https://material.io/color/>

Tipografia

Per a la versió Android, es farà servir la tipografia Roboto, principalment en les seves versions *Light, Regular, Medium* i *Bold*.

Roboto

Thin
Thin Italic
Light
Light Italic
Regular
Regular Italic
Medium
Medium Italic
Bold
Bold Italic
Black
Black Italic

Per a la versió iOS, es farà servir la tipografia San Francisco, recomanada per les guies de disseny d'Apple.

San Francisco

Ultralight
Thin
Light
Regular
Medium
Semibold
Bold
Heavy
Black

Iconografia

Per a ambdues versions, Android i iOS, es farà servir una iconografia fidel a les recomanacions de les corresponents guies d'estils, sobretot pel que fa a la barra de navegació inferior, la capçalera i les opcions de menú. En aquest sentit, la base de dades d'icones Icons8 ofereix una àmplia gamma d'opcions, adaptades a l'estil de cada plataforma.

Addicionalment, s'ha creat la següent icona per indicar si les dades estan representant informació del moment actual, a partir de certa hora del dia corrent o a partir de cert dia del mes corrent.

3.3.2. Prototips Android

Figura 23: "TAULIREPORT" Android inici de sessió.

Figura 24: "TAULIREPORT" Android inici.

Figura 25 "TAULIREPORT" Android urgències per ingressar.

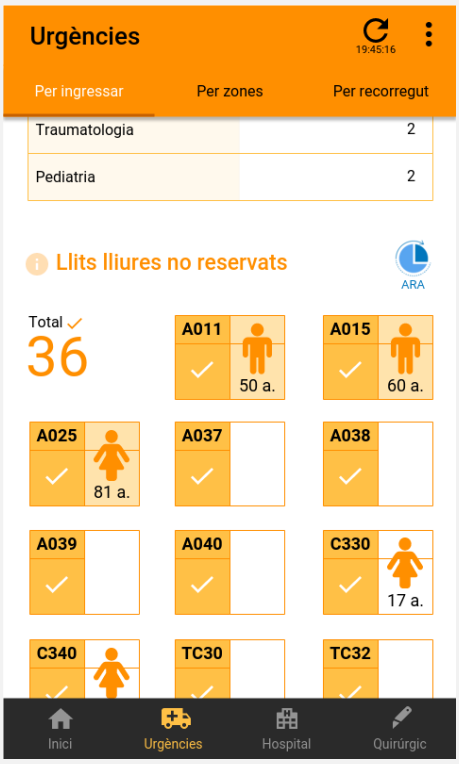


Figura 26: "TAULIREPORT" Android urgències per ingressar.

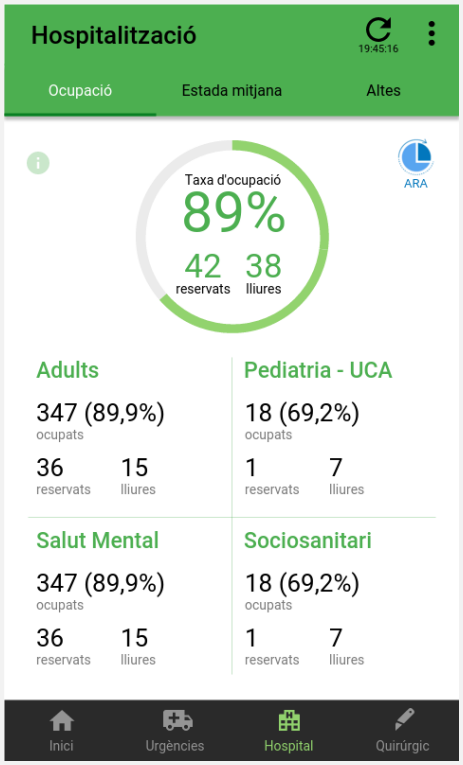


Figura 27: "TAULIREPORT" Android hospitalització ocupació.

Figura 29: "TAULIREPORT" Android menú superior.

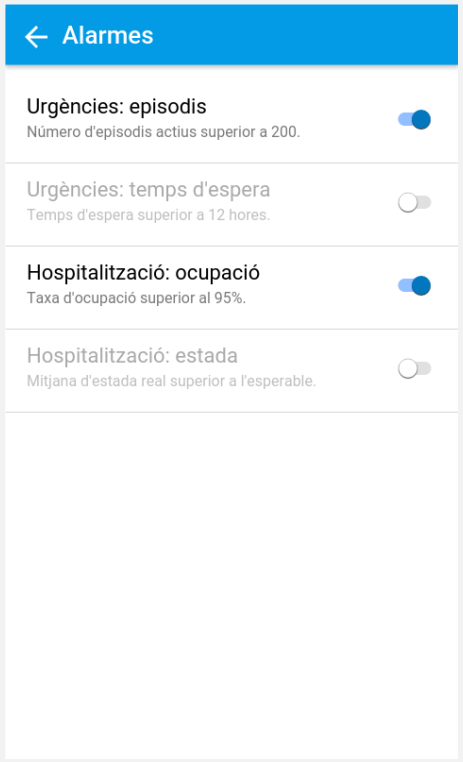


Figura 28: "TAULIREPORT" Android alarmes.

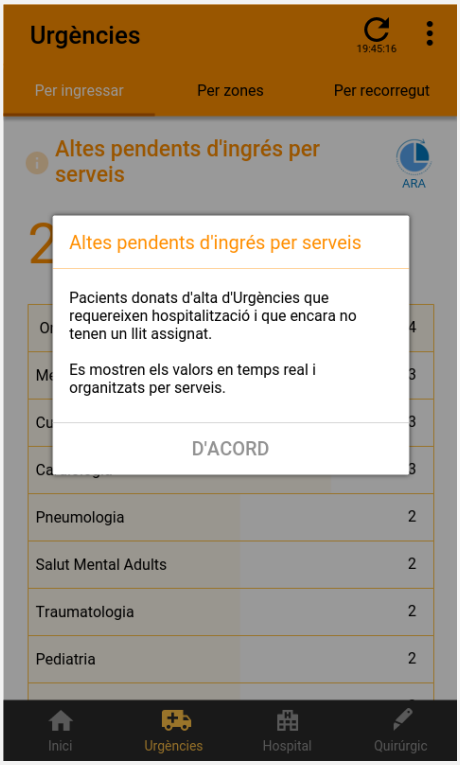


Figura 30: "TAULIREPORT" Android ajuda per interpretar dades.

3.3.3. Prototips iOS

Figura 31: "TAULIREPORT" iOS inici de sessió.

Figura 32: "TAULIREPORT" iOS inici.

Figura 33: "TAULIREPORT" iOS urgències per ingressar.

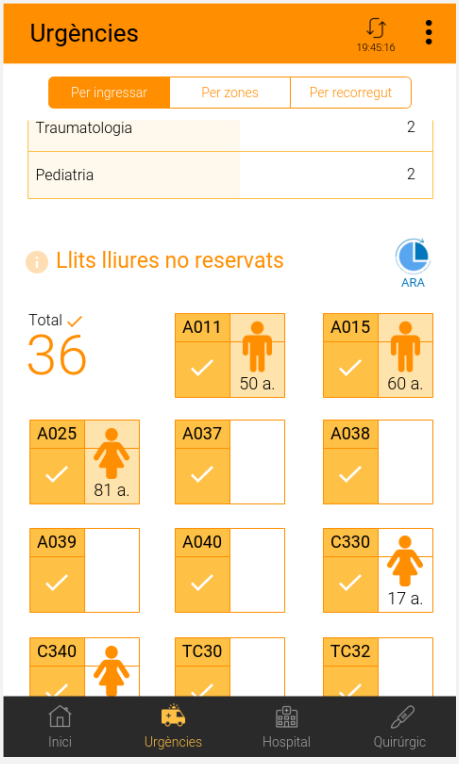


Figura 34: "TAULIREPORT" iOS urgències per ingressar.

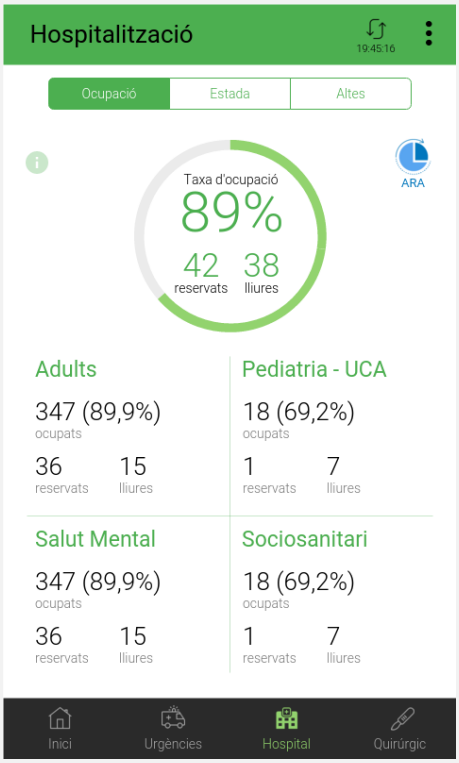


Figura 35: "TAULIREPORT" iOS hospitalització ocupació.

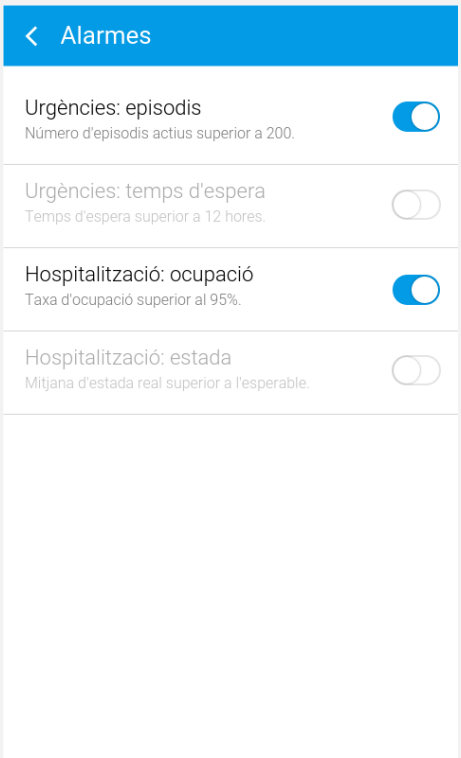


Figura 36: "TAULIREPORT" iOS alarmes.

Figura 37: "TAULIREPORT" iOS menú superior.

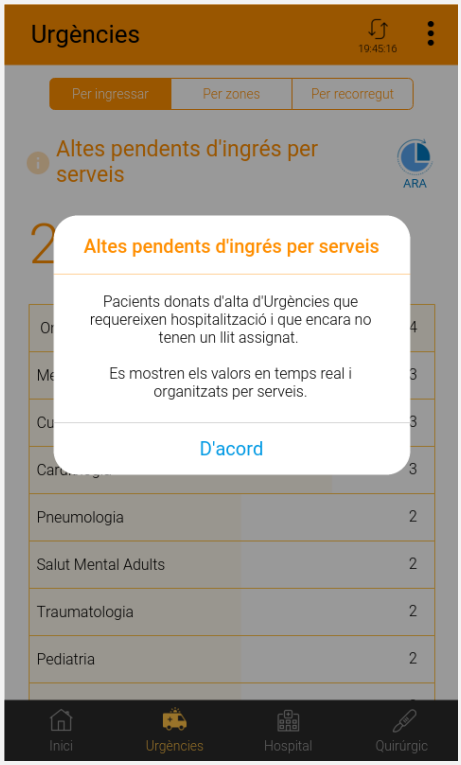


Figura 38: "TAULIREPORT" iOS ajuda per interpretar dades.

3.4. Patrons d'interacció i gestos

La majoria de zones d'interacció dins de l'aplicació responen a un **clic simple (*tab*)** amb el dit: el menú inferior principal, el menú superior contextual, l'actualització, o les pestanyes dins de cada bloc principal.

Cal apreciar en els prototips anteriors que l'aplicació té dos nivells d'estructuració dels continguts: el menú inferior amb els blocs principals d'informació i el **contingut tabulat dins de cada bloc**. Per enriquir la interacció amb aquest segon nivell, s'incorporarà un moviment de lliscament lateral (*swipe*) per passar d'una pestanya a l'altra.

Així el canvi es podrà realitzar des de qualsevol punt de la pestanya activa i no únicament des del propi botó superior.

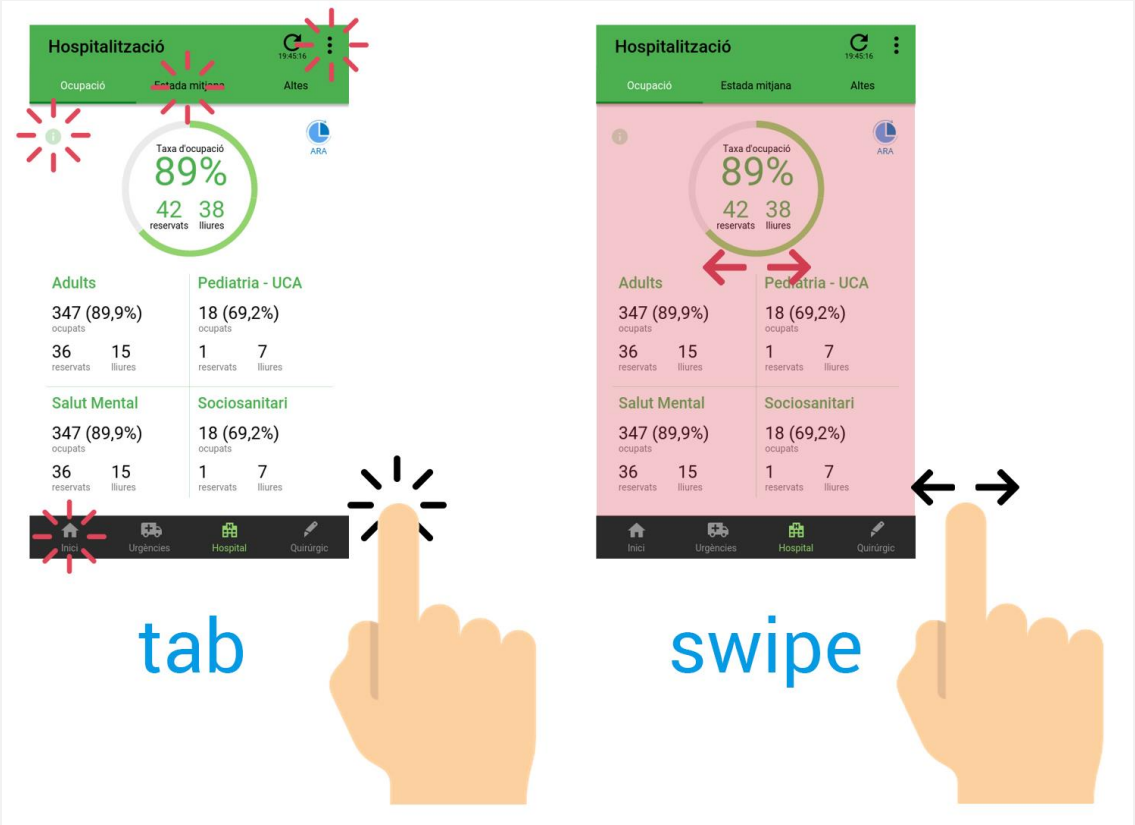


Figura 39: Principals interaccions que escoltarà l'aplicació, *tab* i *swipe*, aquest últim reservat per la tabulació dels blocs principals.

4. Desenvolupament

En aquest quart capítol, es descriu el procés de desenvolupament del "Taulireport", des de la creació de l'entorn de treball fins a l'assoliment del primer prototip real. Així mateix, es farà èmfasi en aquelles parts que mereixen una especial atenció per la seva peculiaritat.

A partir d'aquest capítol, totes les imatges de l'aplicació correspondran a captures reals.

4.1. Entorn de treball

Maquinari

- **Ordinador sobretaula DELL model XPS 8900, Intel-Core I7 3.4 GHz, 16 MB RAM:** eina principal de desenvolupament, on està instal·lat la majoria del programari.
- **Telèfon intel·ligent Samsung Galaxy S7 Edge:** entorn de prova de l'aplicació en un dispositiu real.

Programari

- **Windows 10 Home 64 bits:** sistema operatiu de l'ordinador de sobretaula.
- **Visual Studio Code[8]:** editor de text altament compatible amb Ionic gràcies a la instal·lació de diferents extensions que faciliten l'edició del codi.
- **Adobe Illustrator CS6:** programa de creació de gràfics vectorials amb el qual s'ha generat el logotip de l'aplicació i altres elements gràfics.
- **Android 7.0 Nougat:** sistema operatiu mòbil on correrà l'aplicació com a entorn de prova real.

Visual Studio Code ha sigut, doncs, l'eina escollida per suportar tot el desenvolupament del codi de l'aplicació. A banda de ser un programari lleuger, la possibilitat d'instal·lar extensions permet al desenvolupador ajustar l'entorn a l'API escollida i facilitar així l'experiència de programació: Angular v5 Snippets, Ionic 2 snippets, o Path Autocomplete, entre d'altres.

Gràcies a l'API de **Legacy Ionic View[9]**, el telèfon de Samsung ha servit com a entorn de prova immediat per assegurar que l'aplicació corria correctament i avaluar l'experiència d'ús des d'un

8 <https://code.visualstudio.com/>

9 Legacy Ionic View es troba en procés d'extinció i serà accessible fins el 31 de gener del 2018. La nova versió d'Ionic View ja està disponible. <https://ionicframework.com/docs/pro/migration/view.html>

dispositiu mòbil. Aquesta experiència ha estat clau, ja que molts canvis que ha patit el disseny des de l'etapa de prototipatge han vingut d'aquí, com s'explicarà més endavant.

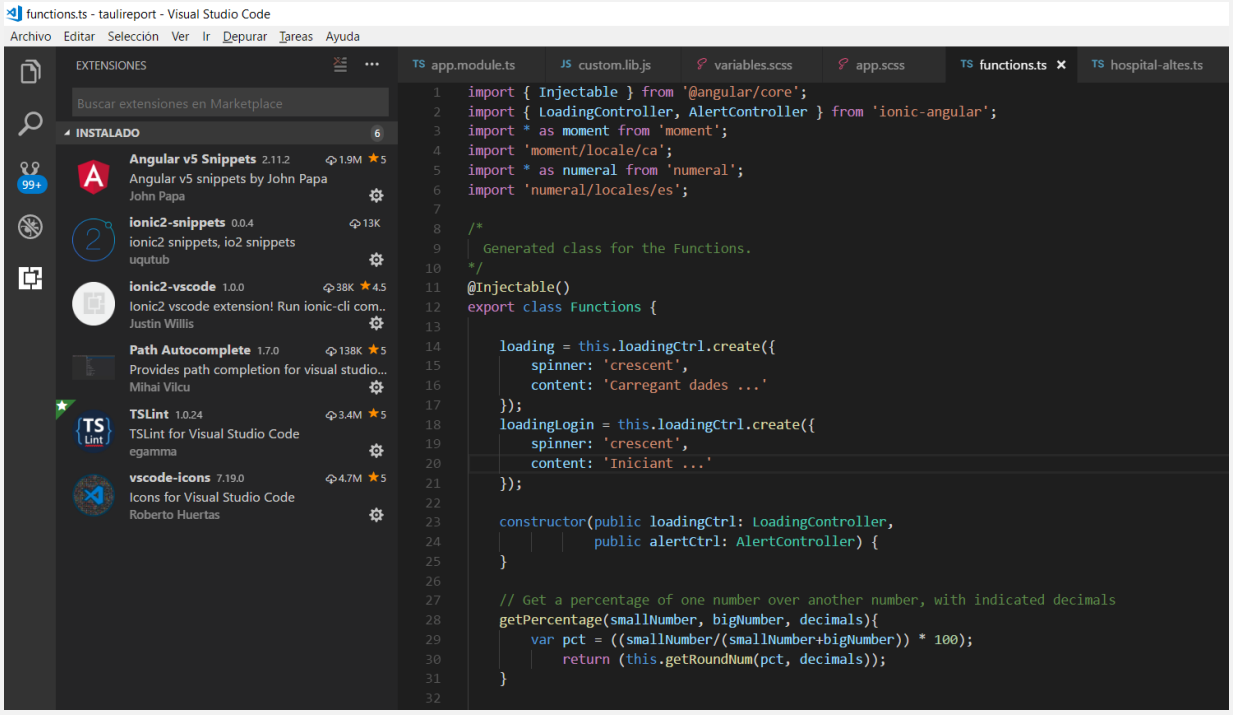


Figura 40: Visual Studio Code amb les sis extensions instal·lades per millorar l'experiència de programació amb Ionic 2.

4.2. Arquitectura

Seguint el model d'arquitectura creat per a l'anterior versió de l'aplicació, la part de l'aplicació és la que ha patit més renovacions. En aquesta nova versió, es deixa de costat Framework7 per **incorporar el framework d'Ionic 2**, que necessàriament arrossega les capes d'**Angular** i **Node** per sota. Aquest nou entorn, tot i que manté **HTML5** com a llenguatge de marcatge web, implica l'ús de **Typescript** com a llenguatge de programació, que suportarà tota la lògica de l'aplicació, i un llenguatge de preprocessament de CSS, en aquest cas **SCSS**.

Igualment, es segueix requerint **Cordova** per l'exportació de tot el projecte cap a les plataformes mòbils. Bàsicament el codi de fons és el mateix que es pot trobar a Phonegap, però és preferible emprar la denominació Cordova ja que serà aquest framework el que s'instal·larà des de Node i a partir del qual es generaran les aplicacions híbrides instal·lables.

De la part de *back-end* no s'han produït canvis. Originàriament, es pretenia modificar els serveis PHP allotjats al servidor web Linux, per convertir-los en una API REST real que facilités les crides des de l'aplicació cap al HIS. Tanmateix, els serveis actuals retornen uns objectes JSON que poden ser perfectament tractats des del nou *front-end* i sembla lògic concentrar tots els esforços en el disseny i desenvolupament del nou entorn d'aplicació.

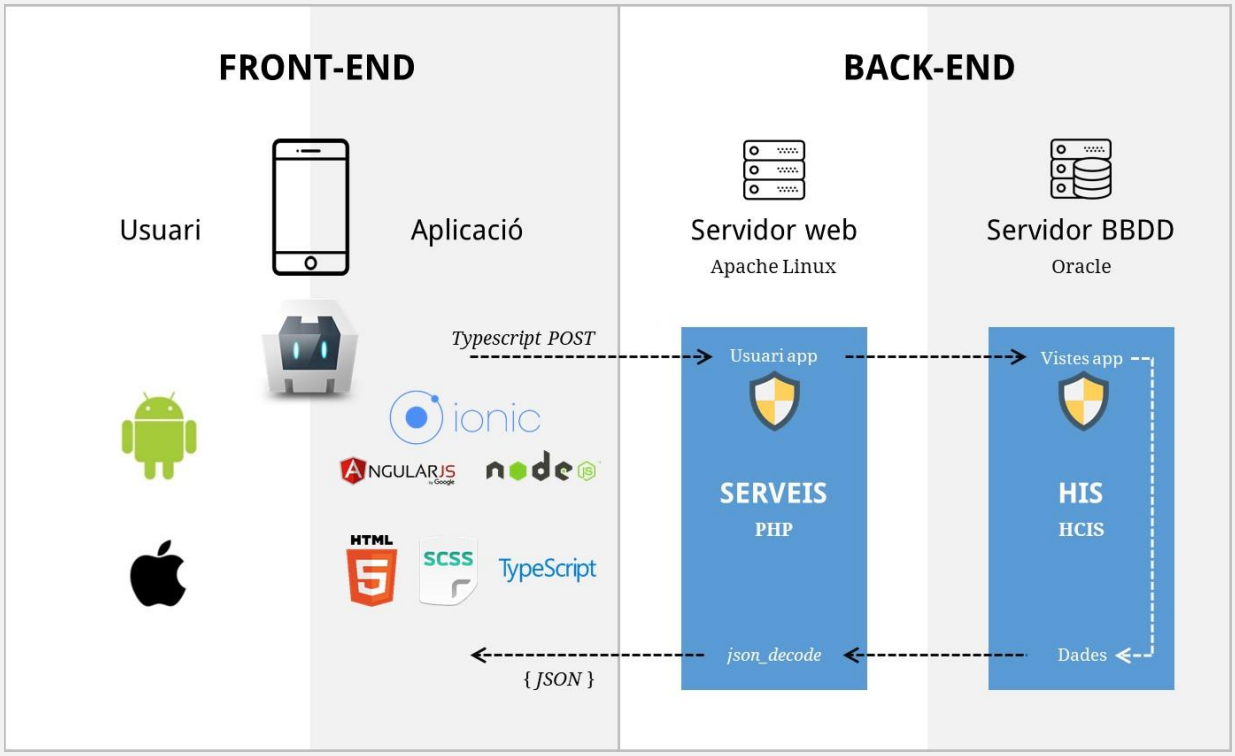


Figura 41: Arquitectura proposada per la nova versió de l'aplicació.

4.3. Llibreries externes

Les llibreries que es mencionen a continuació, tret de les *Super Tabs*, no són estrictament necessàries per construir el primer prototip funcional, però ajuden a millorar la interfície i a processar millor les dades.

4.3.1. Super Tabs

Les *Tabs*[10] que incorpora Ionic de manera nativa són una gran ajuda a l'hora de crear i organitzar el contingut d'una aplicació seguint aquest tipus de disseny tant estès en aplicacions mòbils. No obstant, tenen una mancança que resulta necessàri cobrir per al Taulireport: **no permeten interactuar amb el contingut amb gestos *swipe*.**

Amb aquest objectiu, s'incorpora la llibreria *Super Tabs*[11], que permet, no només generar la tabulació del contingut i la interacció amb *swipe* (dreta i esquerra), sinó també ajudes de disseny com la barra inferior que llisca entre pestanyes i que ja s'havia plantejat en el disseny dels prototips.

[10] <https://ionicframework.com/docs/api/components/tabs/Tabs/>

[11] <https://github.com/zyra/ionic2-super-tabs>

Aquesta incorporació al projecte s'ha hagut de fer seguint un ordre concret, ja que també es fan servir les *Tabs* natives d'Ionic per a la navegació principal. Així, primerament, s'han creat les *Tabs* inferiors per encapsular tota l'aplicació en les tres opcions de menú[12]:

```
<ion-tabs name="mainNav">
  <ion-tab [root]="tab1Root" tabTitle="Inici" tabIcon="home" class="home"></ion-tab>
  <ion-tab [root]="tab2Root" tabTitle="Urgències" tabIcon="medkit" class="urg"
(ionSelect)="refreshUrgencies()"></ion-tab>
  <ion-tab [root]="tab3Root" tabTitle="Hospital" tabIcon="fa-hospital" class="hos"
(ionSelect)="refreshHospital()"></ion-tab>
  <!--<ion-tab [root]="tab4Root" tabTitle="Cirurgia" tabIcon="fa-cirurgia" class="cir"></ion-
tab>-->
</ion-tabs>
```

Així, cadascuna de les *tabs* porta a una pàgina d'Ionic: *tab1Root* a la *Home*, *tab2Root* a Urgències, i *tab3Root* a Hospital. Així es reconeix en el component de les *Tabs*:

```
import { HomePage } from '../home/home';
import { UrgenciesPage } from '../urgencies/urgencies';
import { HospitalPage } from '../hospital/hospital';
/*import { CirurgiaPage } from '../cirurgia/cirurgia';*/

@Component({
  templateUrl: 'tabs.html'
})
export class TabsPage {

  tab1Root = HomePage;
  tab2Root = UrgenciesPage;
  tab3Root = HospitalPage;
  /*tab4Root = CirurgiaPage;*/

  constructor() {
  }
}
```

Es veu, doncs, com les *Tabs* no són pàgines en sí mateixes (tot i declarar-la com a tal), sinó un component d'Ionic derivat de la classe *NavController*[13] que permet navegar entre pàgines. Les *Super Tabs* tenen una declaració molt similar, però la principal diferència és que cadascun dels components contenidors de les pestanyes sí que s'ha de considerar com una pàgina.

[12] Es prescindeix de la quarta, l'àmbit quirúrgic, ja que finalment no es desenvoluparà per a aquest treball final, tot i que s'inclogui en futures actualitzacions.

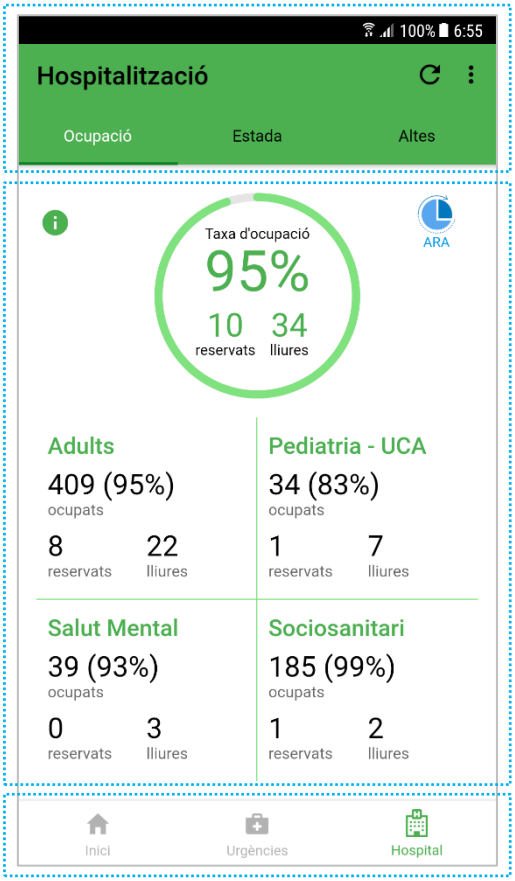
[13] <https://ionicframework.com/docs/api/navigation/NavController/>

Les *Super Tabs* s'incorporen tant a la secció d'Urgències com a la d'Hospital. Si es pren d'exemple l'html de la pàgina Hospital, es pot veure com es declara, no només el codi necessari per a la presentació de les *Super Tabs*, sinó també els components per a la capçalera (<*ion-header*>) i per al contingut (<*ion-content*>):

```
<ion-header>
  <ion-navbar>
    <ion-title>Hospitalització</ion-title>
    <ion-buttons end>
        <button (click)="refresh()" ion-button icon-only>
          <ion-icon name="refresh"></ion-icon>
        </button>
        <button (click)="presentPopMenu($event)" ion-button icon-only>
          <ion-icon name="more"></ion-icon>
        </button>
    </ion-buttons>
  </ion-navbar>
</ion-header>

<ion-content no-bounce>
    <super-tabs id="hosTabs" indicatorColor="hosdark">
      <super-tab [root]="tab1Root" title="Ocupació"></super-tab>
      <super-tab [root]="tab2Root" title="Estada"></super-tab>
      <super-tab [root]="tab3Root" title="Altes"></super-tab>
    </super-tabs>
</ion-content>
```

En el seu component, es pot comprovar que la pàgina Hospital també fa crides a les sub-pàgines declarades com a *tab1Root* (Ocupació), *tabRoot2* (Estada) i *tabRoot3* (Altes):

```
import { HospitalOcupacioPage } from "../hospital-ocupacio/hospital-ocupacio";
import { HospitalEstadaPage } from "../hospital-estada/hospital-estada";
import { HospitalAltesPage } from "../hospital-altes/hospital-altes";
@IonicPage()
@Component({
  selector: 'page-hospital',
  templateUrl: 'hospital.html',
})
export class HospitalPage {

  tab1Root = HospitalOcupacioPage;
  tab2Root = HospitalEstadaPage;
  tab3Root = HospitalAltesPage;

}
```



Per tant, cal tenir en compte que si es vol modificar l'aspecte de la capçalera (amb el botó de refrescar, el menú contextual, pestanyes, color, ...), s'ha de recórrer a la pàgina contenidora de les *Super Tabs*, Hospital.

Per a modificar del contingut de cada *supertab*, cal anar a buscar la pàgina que correspongui, Ocupació en aquest cas.

Per modificar les *tabs* de la navegació principal inferior, llavors cal anar a buscar el component *Tabs*.

4.3.2. Material Icons i FontAwesome

Ionic incorpora nativament una llibreria d'icones que resolen la majoria de necessitats relacionades amb facilitar la interpretació i navegació per les interfícies. No obstant, les llibreries de Material Icons[14] i FontAwesome[15] afegeixen més riquesa a les opcions disponibles, sobretot aquesta última, que ha permès vestir l'apartat d'hospitalització amb una icona molt representativa.

Per a la seva inclusió, es pot seguir la documentació dels seus webs. Bàsicament es tracta de crear una còpia dels arxius de fonts i dels SCSS des de *node_modules* a la carpeta d'exportació *www*:

```
copyFontawesomeFonts: {
        src: ['{{ROOT}}/node_modules/font-awesome/fonts/**/*'],
        dest: '{{WWW}}/assets/fonts'
    },
copyFontawesomeCss: {
        src: ['{{ROOT}}/node_modules/font-awesome/css/font-awesome.min.css'],
        dest: '{{WWW}}/assets/css'
    }
```

[14] <https://github.com/zyra/ionic-material-icons>

[15] <https://charlouze.github.io/ionic/2017/05/31/Ionic-3-and-Font-Awesome.html>

Per emprar les icones en els diferents components, n'hi haurà prou amb declarar la fulla d'estils des de la pàgina *index.html* d'exportació:

```
<link href="assets/css/font-awesome.min.css" rel="stylesheet">
```

4.3.3. Moment.js

Moment.js és una coneguda llibreria que permet presentar variables amb informació temporal, estalviant la programació de complexes algoritmes Javascript que podrien arribar a resoldre el mateix problema.

La seva compatibilitat amb Node.js i per tant amb Ionic és absoluta i n'hi ha prou amb instal·lar el paquet i importar-lo des del component on es farà servir. En el cas del projecte, es poden trobar usos de moment per indicar les referències temporals sobre les dades que es mostren en pantalla:

```
// Moment functions
    getDay () {
        return moment().format("D");
    }
    getMonth () {
        return moment().format("MMMM");
    }
    getMonthShort () {
        let month = moment().format("MMM");
        return month.substr(0,3); // without the unexplicable final dot
    }
    getDateHour () {
        return moment().format("DD/MM/YYYY HH:mm:ss");
    }
```

4.3.4. Numeral.js

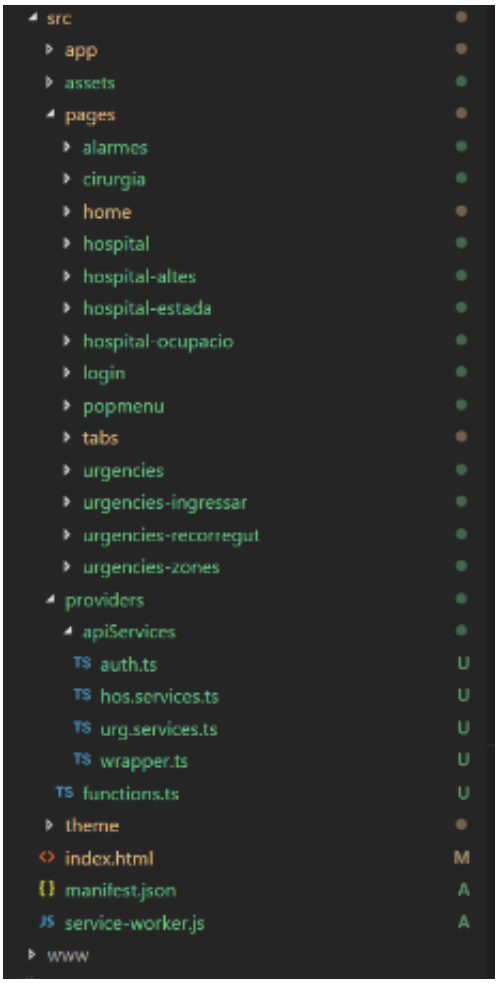
Seguint l'estela de Moment.js (el propi autor confessa que li ha servit com a inspiració), la llibreria Numeral.js permet donar format a cadenes de text que representen nombres, oferint una varietat amplíssima de recursos que, de nou, eviten les línies de codi amb funcions a mida.

```
// Numeral funcions
    setThousands ($string) {
        return numeral($string).format('0,0');
    }
    setPercentage ($value1,$value2) {
        return numeral($value1/($value1+$value2)).format('0%');
    }
```

4.4. Codificació

Un cop definits l'entorn de treball, l'arquitectura i les llibreries emprades, es pot aprofundir en el codi generat per al projecte, tant en la seva organització com en l'explicació de les parts més rellevants.

4.4.1. Organització per carpetes



Tot projecte d'Ionic (per herència d'Angular), conté dues carpetes fonamentals:

- ***src***: conté tot el conjunt de carpetes i arxius editables que es processaran en fer la publicació.
- ***www:*** conté totes les carpetes i arxius processats durant la compilació.

Dins de ***src***, doncs, és on es troba el codi font real del "Taulireport". Seguint l'estructura dictada per Angular, aquesta carpeta està organitzada de la següent manera:

app

Carpeta amb els components principals de l'aplicació, on es defineixen totes les importacions (*app.module*), l'arrel del projecte (*app.component*) i la fulla d'estils comuna a totes les pàgines de l'aplicació (*app.scss*).

assets

Carpeta dels recursos d'imatges i icones presents a l'aplicació.

pages

Carpeta contenidora de tots els components que són pàgines de l'app, incloent els arxius *.ts*, *.html* i *.scss*. Tenint en compte l'ús de *Tabs* i *Super Tabs* explicant anteriorment, pot entendre's millor la llista de pàgines generades:

- ***login***: pàgina arrel.
- ***tabs***: navegació principal.
- ***home***: primera *tab* de la navegació principal
- ***urgencies:*** segona *tab* de la navegació principal, contenidora de les *supertabs* d'urgències.

- ***urgencies-ingressar, urgencies-zones, urgencies-recorregut*:** pàgines que són el contingut de les *supertabs* d'urgències.
- ***hospital*:** tercera *tab* de la navegació principal, contenidora de les *supertabs* d'hospitalització.
- ***hospital-ocupacio, hospital-estada, hospital-altes*:** pàgines que són el contingut de les *supertabs* de l'hospital.
- ***popmenu*:** pàgina que s'obrirà com a menú contextual des de totes les capçaleres de les tres pàgines de la navegació principal.
- ***alarmes*:** pàgina accessible des del menú contextual, que s'obrirà una capa per damunt de les *tabs* i *supertabs*.

providers

La creació de la carpeta *providers* obeeix al fet de voler separar els serveis i les crides externes amb el tractament en sí mateix de les dades, fet que també facilita l'edició posterior i l'actualització dels components.

D'una banda, es troben els serveis a la carpeta *apiServices* que realitzen les crides externes i retornen els diferents objectes que seran tractats a les pàgines corresponents:

- Un servei amb totes les *url* de connexió amb els serveis externs i la construcció dels encapçalaments (*headers*) que s'inclouran a les crides (***wrapper.ts***).
- Un servei que és una extensió de la classe anterior, amb la lògica d'autenticació contra l'LDAP de la institució i que assegura que l'usuari que accedeix té efectivament accés autoritzat (***auth.ts***).
- Un altre servei, també extensió del primer, amb les connexions als diferents serveis web que proporcionen les dades per a les diferents pàgines d'urgències (***urg.services.ts***). L'únic que ha de fer aquest servei és unir l'*url* d'allotjament dels serveis web amb l'*url* del servei en concret, dins d'una funció amb una crida POST que es queda en el punt de mapejar la resposta en format JSON, com es pot veure en aquest exemple que retorna les dades d'urgències per a la pàgina principal:

```
getHomeData() {
    return this.http.post( this.urlAppServices + this.urlUrgHome, this.options)
      .map(res=>res.json())
      .catch((error:any) => Observable.throw(error.json().error || 'Server error') );
  }
```

- Un darrer servei, extensió del primer, semblant a l'anterior però en aquest cas per a les pàgines d'hospitalització (***hos.services.ts***).

Les pàgines *login*, *home*, *urgencies* i *hospital* importaran els seus respectius serveis i esperaran a tenir la resposta per a cada crida (*subscribe*) per realitzar el tractament específic de les dades.

Seguint l'exemple anterior, la pàgina *home.ts* importa el servei d'urgències i passa el *provider* pel constructor, i dins del cos de la classe associa la resposta a una variable (*iniUrgencies*) que des de l'*html* es recollirà per mostrar-ne els valors:

home.ts

```
import { UrgenciesServicesProvider } from '../../providers/apiServices/urg.services';

export class HomePage {

constructor( public urgServices: UrgenciesServicesProvider ) {}

getUrgHomeData(){
    this.urgServices.getHomeData().subscribe(res => {
      this.commonFunction.dismissLoading();
      this.iniUrgencies = res[0];
    }, error =>
    {
       alert("error");
    });
  }
```

home.html

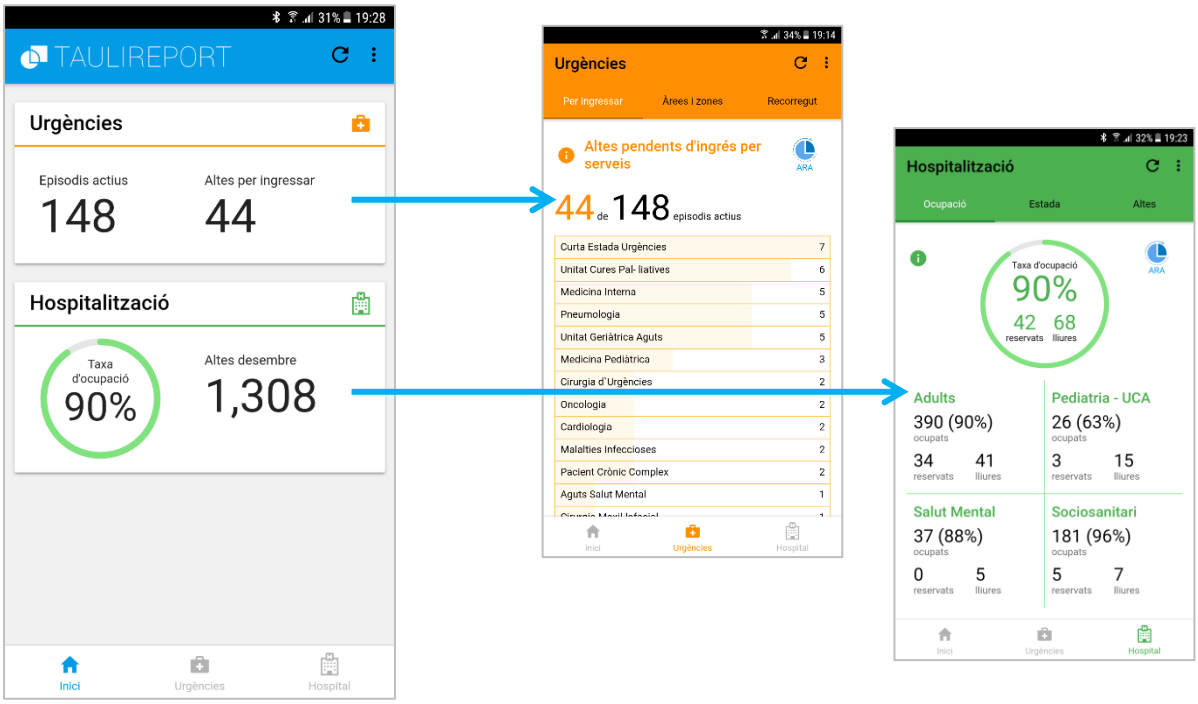
```
<ion-card class="initurg" (click)="goToUrgenciesTab()">
    <ion-card-header>
      <ion-title>Urgències</ion-title>
      <ion-icon name=medkit></ion-icon>
    </ion-card-header>
    <ion-card-content padding-top>
      <ion-grid>
        <ion-row>
          <ion-col>
            <p>Episodis actius</p>
            <p class="dadesL">{{iniUrgencies.EPISODIS_ACTIUS}}</p>
          </ion-col>
          <ion-col>
            <p>Altes per ingressar</p>
            <p class="dadesL">{{iniUrgencies.ALTES_PENDENT_INGRES}}</p>
          </ion-col>
        </ion-row>
      </ion-grid>
    </ion-card-content>
</ion-card>
```

Aquest flux de treball és el que predomina per a tot el que ha implicat connexió amb el HIS i posterior tractament a l'aplicació. Fora dels serveis, però, també s'ha creat una classe *functions.ts*

que recull les funcions d'ús comú per a tota l'aplicació i que també s'haurà d'importar com un *provider* dins de les pàgines que en facin ús. La finalitat última és crear un codi uniforme que es pugui reutilitzar i faciliti l'edició posterior.

4.4.2. Canvi programat de *tab* actiu

Ja s'ha comentat que les *Tabs* de la navegació principal obeeixen el gest de toc simple amb el dit a sobre de cadascuna de les pestanyes a les quals es vol navegar. El "Taulireport" s'ha dissenyat de manera que, estan a la pantalla d'inici, es pugui accedir a urgències o hospital des de les *cards* corresponents.



El problema que planteja aquesta decisió de disseny és que **l'enllaç per *push* no es pot fer servir en aquest cas**, ja que *push* crea un salt cap a dins en la navegació i posaria la pàgina d'urgències o hospital en una capa per damunt de la *home*. **Cal mantenir tota la navegació sempre en un mateix nivell**, excepte en el cas de les alarmes que sí que es col·loca per damunt fent un *push*.

La solució a aquest problema passa per emprar el mètode *getNavByIdOrName()* del component *App*. Havent assignat un nom als *Tabs* principals des del seu html, es pot assignar el nom a una variable des del component *home.ts* que serà una referència directa al component dins de l'aplicació:

```
tabsNav = this.app.getNavByIdOrName('mainNav') as Tabs;
```

Ara amb una funció per a cada *card* es pot navegar a la pestanya d'urgències o hospital sense modificar les capes de profunditat en la navegació, mitjançant el mètode *select(index)* que proporciona aquesta variable que referencia a les *Tabs*:

```
goToUrgenciesTab() {
    this.tabsNav.select(1);
  }
goToHospitalTab() {
    this.tabsNav.select(2);
  }
```

4.4.3. Actualització per esdeveniments

Per assegurar que les **dades estiguin actualitzades** cada cop que l'usuari entra en una de les pàgines, s'ha fet ús d'una funció del **cicle de vida de les pàgines d'Ionic, *ionViewWillEnter()***, per incloure-hi totes les funcions de tractament de la resposta de les crides al servidor i assignació de variables.

Aquesta funció s'executa quan la pàgina està a punt de convertir-se en la pàgina activa, per tant es presenta com el millor moment per a fer la connexió amb el servidor i tornar les dades. Durant aquest temps d'espera, es mostra un component *LoadingController* que s'oculta en quant s'entra en el *subscribe()* de la funció de connexió.

Per l'arquitectura de les *Super Tabs*, apareix el problema que per defecte **tant a urgències com a hospital ni les dades s'actualitzen ni es mostra el controlador de càrrega**. La causa és que quan s'entra a algun d'aquest dos apartats, **la pàgina que està a punt de fer-se activa no és una *supertab*, sinó la pàgina *tab* contenidora.** De la mateixa manera, la capçalera d'aquestes dues pàgines contenidores tenen un botó per actualitzar que tampoc faria el seu efecte, ja que només es podrien cridar funcions de la pàgina *tab* contenidora i no de les pàgines *supertabs* que conté.

La solució d'aquest repte passa per utilitzar l'**emissió i captura d'esdeveniments programats.** L'avantatge dels esdeveniments és que poden ser escoltats per altres pàgines, encara que no estiguin actives, i provocar una resposta en aquestes.

Així, primerament calia crear una funció que es cridés des del botó de la capçalera de cadascuna de les pàgines contenidores, que emetés un esdeveniment (component *Event*) específic que fos capturat per les *supertabs* d'urgències i les d'hospitalització, respectivament. Per al cas d'urgències:

```
refresh(){
      this.events.publish('functionCall:refreshUrgencies');
  }
```

El botó de la capçalera de la *tab* d'Urgències crida la funció *refresh()* que emet un esdeveniment *refreshUrgencies*. Només cal programar totes les *supertabs* d'Urgències perquè, en capturar aquest esdeveniment, provoquin una actualització de crides i variables amb *ionViewWillEnter()*. Aquesta captura s'haurà d'inserir dins del constructor per a que l'escolta sigui efectiva:

```
constructor( public events: Events ) {

    // Subscribe to refresh event
    this.events.subscribe('functionCall:refreshUrgencies', eventData => {
      this.ionViewWillEnter();
    });

  }
```

Per hospitalització es crea una emissió idèntica, però amb diferent nom, *refreshHospital*, que serà escoltada per totes les *supertabs* d'hospitalització.

Aquestes mateixes operacions es repeteixen per a la navegació directa des de les *Tabs* principals. És a dir, que en seleccionar cada *tab* (Urgències o Hospital) es criden funcions que emeten els mateixos esdeveniments que s'han descrit anteriorment.

4.4.4. Notificacions Push

El major repte d'aquest treball ha estat, sens dubte, la implementació de notificacions *push* davant l'aparició d'esdeveniments programats. El servei de missatgeria multiplataforma de Google, **Firebase Cloud Messaging (FCM)**, ha estat clau per aconseguir-ho, però no aporta totes les solucions que el "Taulireport" necessita.

La secció d'Alertes està ideada de manera que, quan alguna de les dades que es mostren a l'aplicació superi un determinat llindar, l'usuari rebi una notificació amb informació sobre aquest esdeveniment. Per exemple, si els episodis d'Urgències està per damunt dels 100, la notificació informaria a l'usuari sobre aquest fet.

No es pot utilitzar la pròpia app per escoltar canvis en les dades crítiques, ja que l'app només les gestiona quan es troba en primer pla i fa crides als serveis web a partir de la interacció de l'usuari. Això comporta que **l'escolta de dades crítiques s'hagi de fer des d'un servei extern.** Aquest servei PHP, que es pot anomenar **servei de notificació,** faria una crida a un dels serveis de l'aplicació ja establerts en el servidor per recuperar la dada crítica a mesurar i, en cas de superar el llindar, gestionar l'enviament cap el servei de missatgeria de Firebase.

Aquest seria l'exemple per al servei de notificació que comprova si hi ha més de 100 episodis actius a Urgències:

```
<?php
$json = file_get_contents(urlAppService);
$obj = json_decode($json);
if ($obj[0]->EPISODIS_ACTIUS >= 100 ) {
    // API access key from Google API's Console
    define( 'API_ACCESS_KEY', apiKey );
```

```
    /**** PREP THE BUNDLE ****/
    // $msg for background notification
    $msg = array
    (
        'body'  => "Hi ha més de 100 episodis actius a l'àrea d'Urgències.",
        'title'     => "Alerta: episodis d'Urgències",
        'vibrate'   => 1,
        'sound'     => 'default',
    );
    // $data for foreground alert
    $data = array
    (
        'body'  => "Hi ha més de 100 episodis actius a l'àrea d'Urgències.",
        'title'  => "Alerta: episodis d'Urgències",
    );
    $fields = array
    (
        'notification' => $msg,
        'data' => $data,
        'to'  => '/topics/urgepisodis',
        'priority' => 'high',
    );
    $headers = array
    (
        'Authorization: key=' . API_ACCESS_KEY,
        'Content-Type: application/json'
    );
    // send message to FCM
    $ch = curl_init();
    curl_setopt( $ch,CURLOPT_URL, 'https://fcm.googleapis.com/fcm/send' );
    curl_setopt( $ch,CURLOPT_POST, true );
    curl_setopt( $ch,CURLOPT_HTTPHEADER, $headers );
    curl_setopt( $ch,CURLOPT_RETURNTRANSFER, true );
    curl_setopt( $ch,CURLOPT_SSL_VERIFYPEER, false );
    curl_setopt( $ch,CURLOPT_POSTFIELDS, json_encode( $fields ) );
    $result = curl_exec($ch );
    curl_close( $ch );
    echo $result;
}
?>
```

Es pot veure que l'objecte enviat a FCM conté dos *arrays* de missatges: *$msg*, que s'inclou al camp *notification* de l'objecte *$fields*, i *$data*, que s'inclou al camp *data*. La raó és que **quan l'aplicació rep la notificació, la gestiona de manera diferent segons si es troba activa en primer pla o tancada**. En cas d'estar tancada, pren el contingut del camp *notification* i la notificació es rep des de la barra d'estat. No obstant, si l'aplicació es troba en primer pla, pren el contingut del camp *data* i s'ha de programar de quina manera es vol que l'aplicació mostri la notificació. Pel cas del

"Taulireport", es mostra una caixa d'alerta. Tota aquesta gestió es realitza des de la configuració de FCM a l'*app.component* de l'aplicació:

```
// Notification received
      this.fcm.onNotification().subscribe((data) => {
          if (data.wasTapped) {
            // App is in background, notification received in status bar, do nothing
          } else {
            // App is in foreground, show notification in alert message
            let title = data.title;
            let message = data.body;
            let button = "D'acord";
            commonFunction.showAlert(title,message,button);
          }
        }, error => { console.error("Error in notification",error) }
      );
```

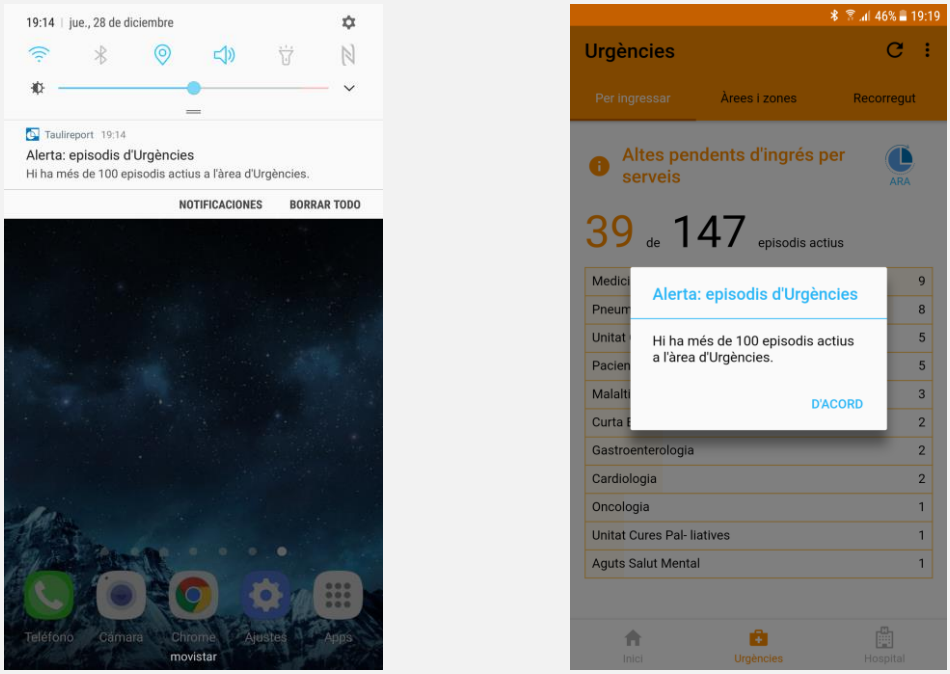


Figura 42: Gestió de les notificacions des de l'aplicació en segon pla (esquerra) i primer pla (dreta).

Un arxiu PHP, però, no pot fer crides per sí mateix. És necessari crear algun tipus de mecanisme que, cada cert temps, invoqui el servei de notificació per iniciar les crides al servei de l'aplicació i a FCM. Aquest problema s'ha resolt amb la creació de tasques *cron*[16], comandes Linux que permeten l'execució periòdica d'*scripts*.

[16] En el moment de l'entrega d'aquest treball final de Màster, les tasques *cron* s'han programat des d'un servidor extern al Parc Taulí, per facilitar la tasca de l'estudiant i poder implementar-les amb agilitat i no dependre dels serveis informàtics de la institució. No obstant, quan l'aplicació s'instal·li en els dispositius dels usuaris finals, les tasques estaran degudament integrades en el servidor Linux del Parc Taulí.

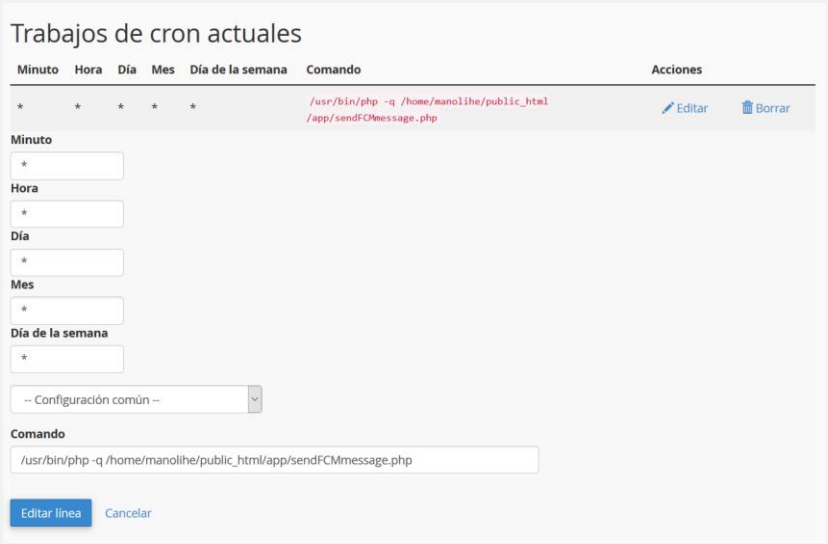


Figura 43: Exemple de tasca *cron* programada per executar-se cada minut.

El **flux de gestió de les notificacions *push***, doncs, segueix aquest ordre:

1. Una tasca *cron* executa el servei de notificació cada minut.
2. El servei de notificació crida el servei d'aplicació corresponent.
3. El servei d'aplicació retorna un objecte *json* que conté la dada esperada.
4. Si es compleix la condició de superar el llindar, el servei de notificació crea l'objecte amb la notificació *push* que envia al servei d'FCM sota un únic *topic*.
5. FCM gestiona l'objecte i el fa arribar a tots aquells dispositius que estiguin subscrits al *topic* al qual s'ha enviat el missatge.
6. El dispositiu processa i mostra la notificació a la barra d'estat o com una caixa d'alerta.

Aquest flux de gestió de les notificacions *push* es pot veure representat a la figura 44.

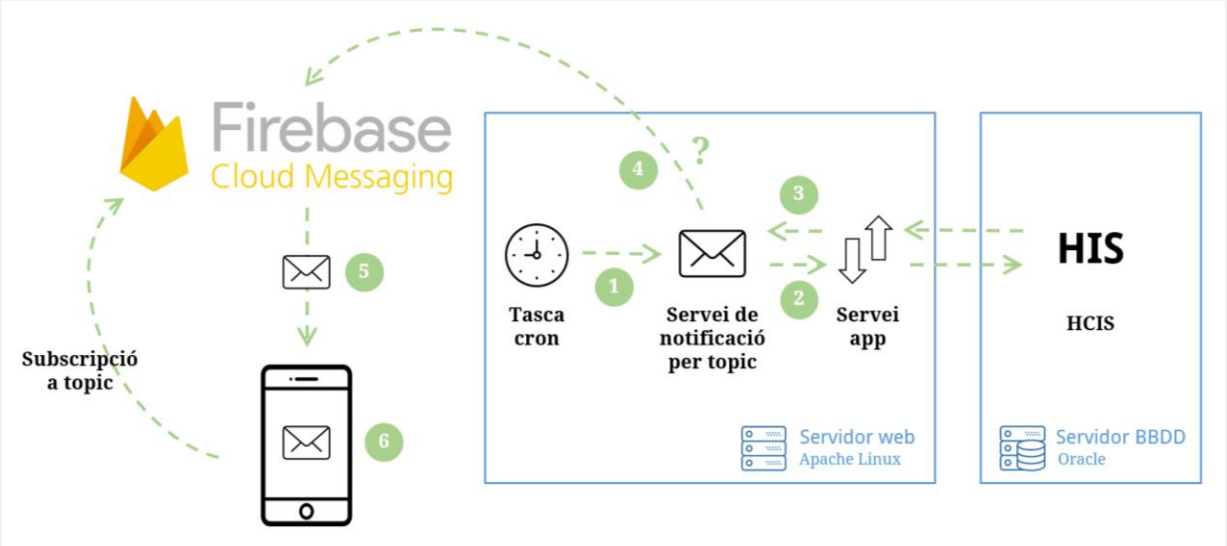


Figura 44: Flux de gestió de les notificacions *push*.

Addicionalment, per rebre la notificació, **cal que el dispositiu on corre l'aplicació s'hagi subscrit al *topic*** al qual el servei de notificació envia el missatge. Aquesta subscripció (i cancel·lació de la subscripció) es fa des de la pàgina d'Alertes, mitjançant l'activació o desactivació de cadascuna de les alertes individuals.

Per a la **gestió d'alertes des de l'aplicació**, són d'extrema utilitat les **directives d'Ionic i Angular**. Prenent com a exemple l'html del botó *toggle* de les alertes d'episodis a Urgències:

```
<ion-item>
   <ion-label [ngClass]="{'inactive': !urgEpisodisActius}">Urgències: episodis
      <span>Número d'episodis actius superior a 100</span>
    </ion-label>
    <ion-toggle [(ngModel)]="urgEpisodisActius"
(ionChange)="toggleAlert(urgEpisodisActius,'urgepisodis')"></ion-toggle>
</ion-item>
```

El botó <*ion-toggle*> està lligat a la directiva ***[(ngModel)]***, de manera que la variable booleana "urgEpisodisActius" obtindrà els valors *true* o *false* segons si el botó està actiu o desactivat.

Gràcies a la directiva ***[ngClass]***, el valor d'aquesta variable "urgEpisodisActius" servirà per afegir o treure la classe "inactive" a l'element <*ion-label*> on hi ha el text de l'alerta, amb el qual es baixarà l'opacitat de l'element en cas que l'alerta estigui desactivada.

<*ion-toggle*> també està lligat a l'esdeveniment ***(ionChange)***, que crida una funció del component encarregada de la subscripció o cancel·lació de la subscripció al *topic* de la notificació. Aquesta funció també desa a *Storage* l'estat dels botons per recuperar-los en tancar i obrir de nou la pàgina Alertes.

```
/* Subscribe or unsubscribe from topic, depending on alert value */
  toggleAlert(alert,topic) {
    alert? this.fcm.subscribeToTopic(topic) : this.fcm.unsubscribeFromTopic(topic);
    //store new values of alerts
    this.storage.set('storedAlerts', JSON.stringify({
      urgepisodis : this.urgEpisodisActius,
      urgtemps : this.urgTempsEspera,
      hosocupacio : this.hosOcupacio,
    }));
  }
```

L'estat de les alertes s'ha de comprovar abans d'entrar a la pàgina Alertes, recuperant els valors desats a l'*Storage*. Per això, el moment ideal és quan la pàgina esdevindrà activa, que seguint el cicle de vida de les pàgines d'Ionic es captura a la funció *ionViewWillEnter()*:

```
ionViewWillEnter() {
    // Check if alerts are stored
    this.storage.ready().then(() => {
      this.storage.get("storedAlerts").then((storedAlerts) => {
        if(storedAlerts != null) {
          let alert = JSON.parse(storedAlerts);
          this.urgEpisodisActius = alert.urgepisodis;
          this.urgTempsEspera = alert.urgtemps;
          this.hosOcupacio = alert.hosocupacio;
        }
      });
    });
  }
```

4.5. Depuració i prova

Tal i com es va plantejar en la planificació del capítol 1, el desenvolupament ha seguit un procés iteratiu de disseny, programació i prova / anàlisi. Aquestes iteracions s'estenen més enllà dels blocs principals de continguts (Urgències, Hospitalització, Alertes, etc.), ja que cada bloc conté multitud de processos iteratius en les pàgines que el conformen o inclús en els components de cadascuna d'aquestes pàgines.

S'identifiquen tres moments clau en la depuració i prova del codi programat i del disseny:

1. Ionic Lab.
2. Ionic View.
3. Cordova Build (Debug).

4.5.1. Ionic Lab

Ionic Lab és l'eina que proporciona el framework Ionic per depurar i comprovar l'aplicació des del propi navegador. La limitació d'aquest entorn de proves és que totes les funcions natives que depenguin de Cordova no es poden testejar, però és una primera aproximació per comprovar la interacció i el disseny programat des de Visual Studio Code.

Una de les gran avantatges d'Ionic Lab és que permet visualitzar les tres plataformes mòbils des de la mateixa finestra del navegador, fet que ajuda a avaluar els matisos de disseny i ajustar amb més precisió la personalització de les vistes. Així mateix, Ionic Lab proporciona un accés directe a la documentació d'Ionic per facilitar la resolució de dubtes.

Per poder executar Ionic Lab, calia superar la limitació que els navegadors tenen per defecte de no permetre les sol·licituds de recursos que es troben en dominis diferents al domini on es troba l'script des d'on s'ha fet la petició (**CORS, Cross-Origin Resource Sharing**). En efecte, com es veu

a la figura 45, les peticions s'estan fent des de *localhost:8100* cap al servidor del Parc Taulí. Tot i que es podrien incloure encapçalaments per permetre aquestes peticions, en realitat només és una limitació que apareix en la depuració per navegador. N'hi ha hagut prou amb instal·lar i activar l'extensió de Chrome **Allow-Control-Allow-Origin: *** [17] per saltar la limitació de CORS.

Per obrir Ionic Lab, s'executa la comanda des del mateix terminal integrat a Visual Studio Code:

```
ionic serve --lab
```

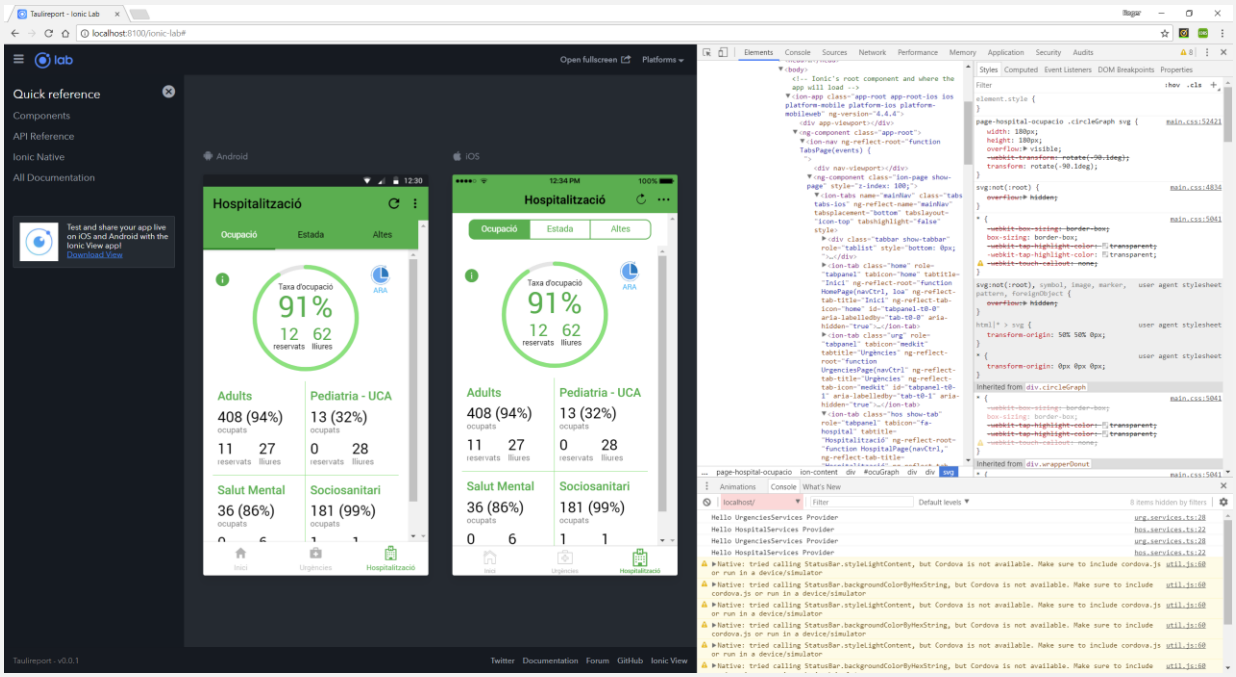


Figura 45: Ionic Lab des del navegador Chrome, amb les vistes d'Android i iOS.

4.5.2. Ionic View

Ionic View és la primera aproximació en un dispositiu real. És una aplicació que proporciona un servei al núvol on es pot pujar l'aplicació des del propi terminal de l'entorn de desenvolupament.

Primerament, cal vincular el projecte a una de les apps del compte personal a Ionic[18]. Un cop vinculats, es pot anar pujant el projecte per a ser provat des d'un dispositiu mòbil.

```
ionic link
ionic upload
```

[17] <https://chrome.google.com/webstore/detail/allow-control-allow-origi/nlfbmbojpeacfghkpbjhddihlkkiljbi>

[18] <https://apps.ionic.io/>

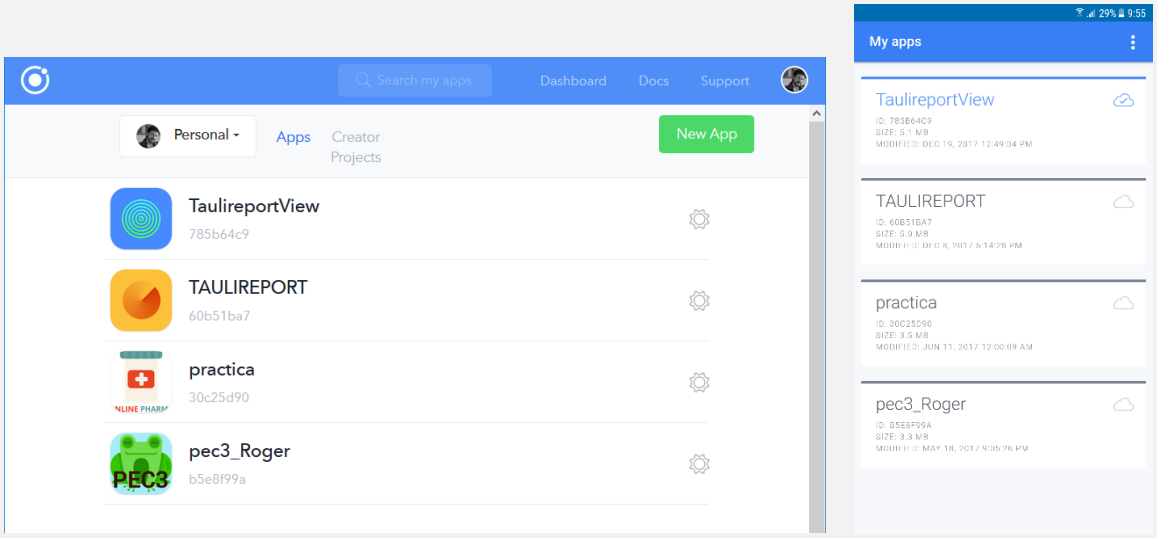


Figura 46: Compte d'Ionic des de navegador (esquerra) i l'app Ionic View per a Android (dreta).

En aquest nivell de testeig, es poden posar a prova patrons d'interacció *swipe* i sobretot es posa a prova el propi disseny. Els prototips d'alta fidelitat presentats durant la segona entrega s'han realitzat sobre el paper, però per qüestions de temps i planificació no s'ha emprat cap programari de prototipatge com ara Justinmind[19] o Axure[20] per fer una aproximació a dispositiu real.

Tenir l'aplicació corrent des d'un dispositiu mòbil modifica les sensacions que s'obtenien al dissenyar i observar els prototips des del paper. Un exemple és el *tabbar* inferior amb la navegació principal, que en els prototips s'havia plantejat amb un color de fons molt fosc, però que ara des d'un *smartphone* donava la impressió d'encapsular l'aplicació i no permetre que respiri.

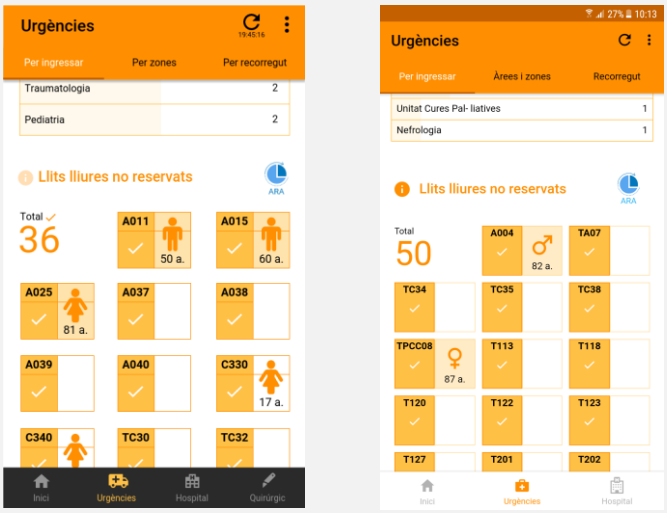


Figura 47: Petites diferències de disseny entre el prototip HD (esquerra) i l'app real (dreta).

[19] <https://www.justinmind.com/>

[20] <https://www.axure.com/>

4.5.3. Cordova Build (Debug)

La comprovació definitiva del disseny de l'aplicació i de la correcta execució del codi es realitza exportant l'aplicació a la plataforma mòbil escollida (Android o iOS) i instal·lant l'arxiu en un dispositiu real.

Aquest tercer moment de depuració és l'indicat per **provar tots els plugins que depenen de Cordova** i en els altres tests no podien executar-se, com ara la **barra d'estat** (que canvia de color segons la pàgina) i les **notificacions *push***.

Android permet fer l'exportació d'un *.apk* de depuració (*debug*) sense necessitat de disposar d'una clau de desenvolupador. Aquesta *.apk* es pot instal·lar en un dispositiu Android sempre i quan estigui activada l'opció d'instal·lar aplicacions provinents de fonts desconegudes.

Per exportar un arxiu *.apk*, primerament cal incloure la plataforma Android dins del projecte:

```
ionic platform add android
```

Seguidament, s'executa la comanda per exportar l'arxiu *.apk* de testeig, que sortirà amb el nom *android-debug.apk*:

```
ionic cordova build android
```

Per exportar l'aplicació definitiva, si ja es disposa d'una signatura vàlida, només caldria afegir una paraula a la comanda anterior per indicar que és la versió de llançament:

```
ionic cordova build --release android
```

5. Conclusions

"Taulireport" és el resultat de l'aplicació de moltes competències adquirides durant l'aprenentatge del Màster: dissenyar l'arquitectura, la interacció i la interfície de productes interactius multidispositiu; utilitzar de forma efectiva els llenguatges de programació de plataformes mòbils; o emprar eines i entorns de desenvolupament disponibles per a plataformes mòbils, entre d'altres.

L'interès personal per realitzar el Màster en Desenvolupament d'Aplicacions per a Dispositius Mòbils era, sobretot, l'exploració del **desenvolupament híbrid**. Anteriorment s'havia treballat amb entorns com JQuery Mobile (en el projecte final del grau de Multimèdia) i Framework7 (emprat en l'aplicació original "Indicadors Taulí"), però definitivament la combinació de Node.js / Angular / Ionic esdevé l'entorn paradigmàtic per al futur del desenvolupament híbrid.

Un bon framework de desenvolupament, però, no crea per sí mateix una bona aplicació. Tot i que per a aquest treball final de màster es parteix d'un projecte real que es vol evolucionar, **s'ha treballat des del principi com un projecte nou**. S'han qüestionat totes les decisions de disseny preses en el projecte base, elaborant els prototips des d'una fase inicial d'*sketching* fins a l'obtenció de vistes que encaixen dins del mercat actual d'aplicacions mòbils.

Les següents comparatives denoten una evolució justificada en el dissenys, que també traspassen a la mà quan l'usuari interacciona amb un dispositiu real: **ràpida resposta de l'aplicació, suavitat de les animacions, o absència d'errors inesperats.**

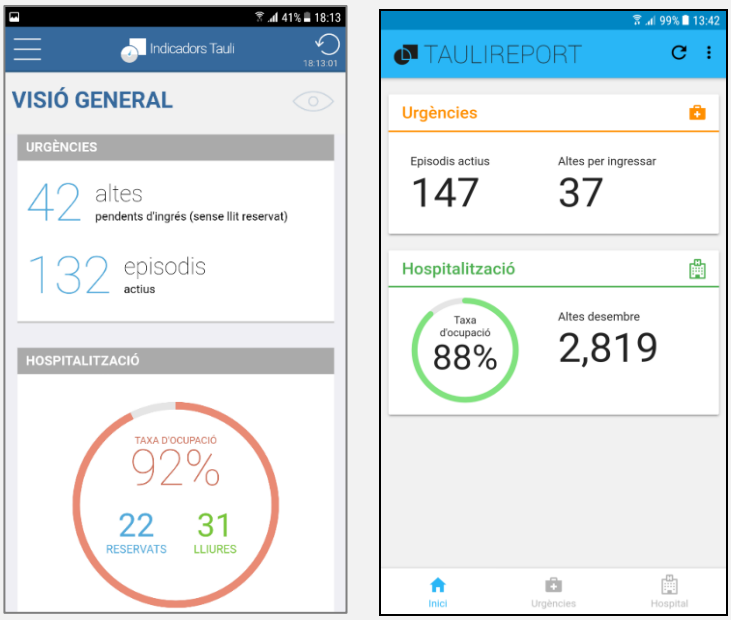


Figura 48: Evolució de la pàgina principal des d'"Indicadors Taulí" a "Taulireport".

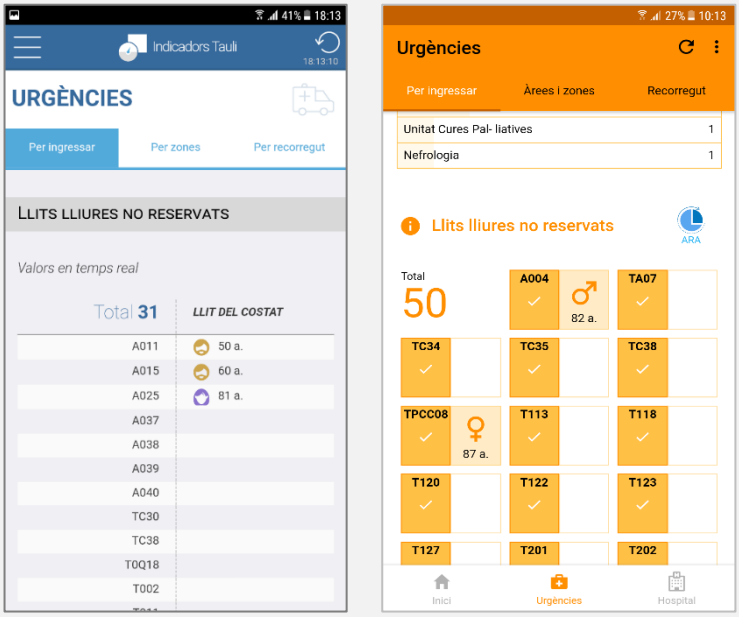


Figura 49: Evolució de la pàgina d'Urgències des d'"Indicadors Taulí" a "Taulireport".

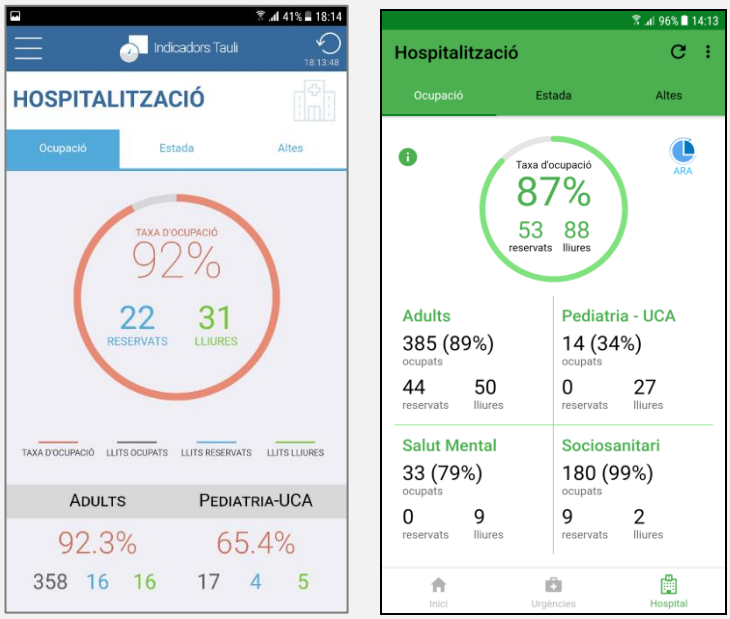


Figura 50: Evolució de la pàgina d'Hospitalització des d'"Indicadors Taulí" a "Taulireport".

Aquests mesos també han permès valorar en la seva justa mesura la **necessitat d'organitzar correctament l'entorn de treball**. Separar les crides a servidor (serveis *providers*) del tractament específic de les dades, tot i requerir un esforç d'abstracció inicial, permet no només la ràpida implementació de les diferents pàgines, sinó també una àgil correcció d'errors i millora. Sobretot, pensant en el futur, l'alt grau d'**escalabilitat** permetrà que l'aplicació pugui créixer amb nous continguts i apartats. Només caldrà crear els nous serveis, afegir-los al *wrapper provider* i realitzar la crida des del *provider* específic.

Seguiment de la planificació

Durant el transcurs de l'etapa tercera d'implementació, la planificació s'ha vist alterada degut a la gran inversió de temps que ha suposat aconseguir tenir una aplicació funcional fidedigne als prototips d'alt nivell presentats en la segona etapa:

- La pàgina "Estada" d'Hospitalització s'ha replantejat recentment des de la Direcció, ja no es contempla l'estada mitjana global com a l'aplicació original "Indicadors Taulí", sinó que es representarà l'estada per serveis i unitats. L'equip de desenvolupament de l'àrea de Sistemes ha de crear les vistes a la base de dades d'HCIS i programar els serveis web. Un cop llestos, s'implementarà aquesta part.
- Les pàgines d'Urgències "Àrees i zones" i "Per recorregut" s'han acabat de dissenyar i programar durant l'etapa 4.
- El sistema d'Alertes i Notificacions *push* hauria d'haver-se implementat durant l'etapa 3 d'Implementació, però ha ocupat bona part de l'etapa 4. Aquest temps s'extreu de la no implementació d'una plataforma de monitoratge. Per ara, la consola de Firebase ja proporciona uns serveis mínims que poden fer la funció fins que no s'hi aprofundeixi en el futur.
- En no comptar amb els certificats i *provisioning files* d'Apple del Parc Taulí, no es podrà generar l'arxiu *.ipa* per a iPhone.

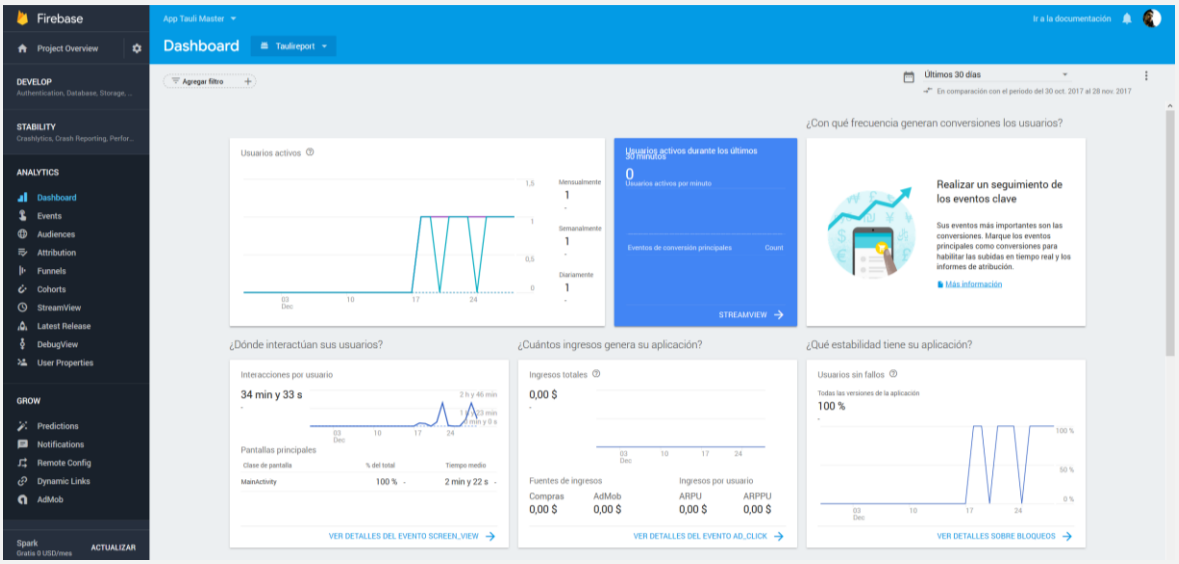


Figura 51: Consola de Firebase que monitoritza l'accés a l'app.

Línies de futur

El més interessant d'afrontar un projecte real com el "Taulireport" és que el codi desenvolupat no es quedarà en l'arxiu empaquetat de l'entrega del treball, sinó que seguirà viu dins de l'entorn controlat d'usuaris del Parc Taulí.

Un cop presentat el projecte en el context del Màster, es realitzarà una presentació per a l'equip de treball del Parc Taulí que va estar implicat en el desenvolupament de la primera versió "Indicadors Taulí". Juntament amb la Direcció de Sistemes d'Informació, es decidirà de quina manera es farà el canvi de versió i la temporització del procés.

A mode d'aproximació, es proposen les següents fites per al projecte "Taulireport" i per al projecte global de mobilitat del Parc Taulí:

- **Immediatament (abans de substituir l'antiga versió):**
    - Revisió de les alertes programades, sobretot quant als llindars proposats.
    - Traspàs de les tasques *cron* al servidor del Parc Taulí.
    - Finalització de la pàgina "Estada" del bloc Hospitalització.
    - Durant el flux d'inici de sessió, comprovar que l'usuari té autorització per emprar l'aplicació (ara mateix qualsevol usuari de l'LDAP podria entrar).
    - Exportació de l'arxiu *.apk* amb una signatura vàlida.
    - Exportació de l'arxiu *.ipa* amb els certificats i *provisioning files* d'Apple del Parc Taulí.
- **A curt termini:**
    - Inclusió de noves alertes.
    - Gestionar les alertes perquè, un cop enviada, s'esperi un temps de gràcia abans de tornar d'enviar la notificació. Per defecte es torna a enviar al cap d'un minut, si el valor segueix per sobre del llindar, tot i que l'usuari pot desactivar manualment l'alerta.
    - Gestionar òptimament la captura del *back button* per *hardware*, en aquests moments tanca l'aplicació per complet i seria preferible posar-la en segon pla.
    - Gestionar millor els errors en la càrrega de dades, ara per defecte es mostra un missatge d'error genèric.
    - Inclusió del bloc quirúrgic.
- **A mig i llarg termini:**
    - Inclusió de nous apartats i funcionalitats, com per exemple una pàgina amb l'històric de dades, o les alertes configurables (valors llindars, temps d'avís).
    - Explotació del projecte de mobilitat al Parc Taulí: noves aplicacions per a professionals i entrada en l'ecosistema d'apps per a pacients.

Glossari

Angular: Framework de Google de codi obert per aplicacions web de pàgina única i escrites en Typescript.

FCM: Inicials de Firebase Cloud Messaging, servei de missatgeria multiplataforma de Google.

Ionic: Framework per a la creació d'aplicacions web mòbils híbrides.

HIS: Hospital Information System.

Node: entorn en temps d'execució multiplataforma, d'arquitectura basada en esdeveniments.

SCSS: extensió de la sintaxi CSS per escriure fulles d'estil millorades.

Swipe: en la interacció amb un dispositiu tàctil, clic amb arrossegament cap als costats.

Tab: en la interacció amb un dispositiu tàctil, clic simple.

TFM: Treball Final de Màster.

Bibliografia

Llibres i publicacions

Salmerón, Sara; Montserrat, Roger. *Modelos de negocio y marketing basados en dispositivos móbiles*. Barcelona, 2017. Materials de l'assignatura de mateix nom del Màster de Desenvolupament d'Aplicacions per a Dispositius Mòbils.

León Silvestre, Javier Jesús. *App Web Scrum* [en línia]. Barcelona: Universitat Politècnica de Catalunya, 2015. <https://upcommons.upc.edu/bitstream/handle/2099.1/23350/Resum.pdf> [Consulta: 7 oct. 2017]

Cuello, Javier; Vittone, José. *Diseñando apps para móviles.* Barcelona: Catalina Duque Giraldo, 2017. <http://appdesignbook.com/es/> [Consulta: 6 nov. 2017]

Articles i Guies

Un 70% dels hospitals i centres de primària catalans utilitzen eines TIC mòbils [en línia]. Barcelona: Fundació TicSalut, 2013. <http://www.ticsalut.cat/actualitat/noticies/seccio/31/56/un-70-dels-hospitals-i-centres-de-primaria-catalans-utilitzen-eines-tic-mobils> [Consulta: 6 oct. 2017].

Sabater Domènech, Jordi. *El método Lean Mobile* [en línia]. El Economista, 12 Agosto 2014. <http://www.eleconomista.es/blogs/expande-tu-negocio-en-internet/el-metodo-lean-mobile/> [Consulta: 10 oct. 2017].

Material Design Guidelines [en línia]. Google, 2017. <https://material.io/guidelines/> [Consulta: 16 oct. 2017].

Human Interface Guidelines [en línia]. Apple, 2017. <https://developer.apple.com/ios/human-interface-guidelines/overview/themes/> [Consulta: 16 oct. 2017].

How to Send Data with POST Requests in Ionic 2. Josh Morony, 5 abril 2017. <https://www.joshmorony.com/how-to-send-data-with-post-requests-in-ionic-2/> [Consulta: 15 nov. 2017].

Coyier, Chris. *How SVG Line Animation Works*. CSS Tricks, 18 febrer 2014. <https://css-tricks.com/svg-line-animation-works/> [Consulta: 15 nov. 2017].

Oberlehner, Markus. *Creating a Pure CSS Animated SVG Circle Chart*. 15 juny 2017. <https://markus.oberlehner.net/blog/pure-css-animated-svg-circle-chart/> [Consulta: 15 nov. 2017].

Alex. *Layout the Cool Way: Using the Ionic 2 Grid Component*. Blog Ionic Framework, 31 agost 2016. <http://blog.ionicframework.com/layout-the-cool-way-using-the-ionic-2-grid-component/> [Consulta: 17 nov. 2017].

Strasser, Dominik. *Ionic - programmatically change active tab*. One Second. <https://www.one-second.me/tips-and-tricks/ionic-programmatically-change-active-tab> [Consulta: 25 nov. 2017].

ionic 2 call function from another controller. StackOverFlow, 17 desembre 2016. <https://stackoverflow.com/questions/41196819/ionic-2-call-function-from-another-controller> [Consulta: 8 des. 2017].

How to Create a Data Model in Ionic 2. Josh Morony, 5 abril 2017. <https://www.joshmorony.com/how-to-create-a-data-model-in-ionic-2/> [Consulta: 10 des. 2017]

Firebase how to send Topic Notification. StackOverFlow, 22 febrer 2017. <https://stackoverflow.com/questions/42384157/firebase-how-to-send-topic-notification> [Consulta: 16 des. 2017]

How to retrieve notification message intent.getExtras() when app is background FCM. StackOverFlow, 4 gener 2017. <https://stackoverflow.com/questions/41458359/how-to-retrieve-notification-message-intent-getextras-when-app-is-background-f> [Consulta: 16 des. 2017]

POST data to a URL in PHP. StackOverFlow, 20 juny 2010. <https://stackoverflow.com/questions/3080146/post-data-to-a-url-in-php> [Consulta: 18 des. 2017]

How to Use JSON Data with PHP or JavaScript. Tania Rascia, 21 gener 2017. <https://www.taniarascia.com/how-to-use-json-data-with-php-or-javascript/> [Consulta: 18 des. 2017]

How to Handle Hardware Back Button In Ionic 3. Prantik Vaghela, 17 novembre 2017. <https://pointdeveloper.com/handle-back-button-ionic-3/> [Consulta: 29 des. 2017]

Llibreries i Bases de dades

Sass (Syntactically Awesome StyleSheets). <http://sass-lang.com/documentation/file.SASS_REFERENCE.html>

Tabs. Ionic Framework. <https://ionicframework.com/docs/api/components/tabs/Tabs/>

CSS Utilities. Ionic Framework. <https://ionicframework.com/docs/theming/css-utilities/>

Responsive Grid. Ionic Framework. <https://ionicframework.com/docs/theming/responsive-grid/>

Ionic2-super-tabs. Github, publicat per Zyra Media. <https://github.com/zyra/ionic2-super-tabs>

Icons8 [en línia]. 2017. <https://icons8.com/>

Free Icons Designed by Smashicons [en línia]. Flaticon, 2017. <https://www.flaticon.com/authors/smashicons>

Firebase Cloud Messaging. Firebase. <https://firebase.google.com/docs/cloud-messaging/>

Moment.js [en línia]. 2017. <https://momentjs.com/>

Numeral.js. [en línia]. 2017. <http://numeraljs.com/>