Desenvolupament d'una aplicació per a l'automatització i la gestió de la restauració

Víctor Sánchez Pérez
Desenvolupament d'aplicació mòbils
Àrea de treball final

Francesc D'Assís Giralt Queralt
Carles Garrigues Olivella

02/06/2021

FITXA DEL TREBALL FINAL

Títol del treball:	Desenvolupament d'una aplicació per a l'automatització de la restauració (Easy Order)
Nom de l'autor:	Víctor Sánchez Pérez
Nom del consultor/a:	Francesc D'Assis Giralt Queralt
Nom del PRA:	Carles Garrigues Olivella
Data de lliurament (mm/aaaa):	06/2021
Titulació o programa:	Desenvolupament d'aplicació mòbils
Àrea del Treball Final:	Treball final de màster DADM
Idioma del treball:	Català
Paraules clau	Aplicació multiplataforma, restaurants

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

Easy Order és una aplicació que pretén automatitzar i modernitzar restaurants. Es vol resoldre la problemàtica del servei que ofereixen alguns restaurants, permetent agilitzar el servei i ser més eficient. A més, actualment amb la pandèmia mundial que s'està vivint, és el moment adequat per a modernitzar aquest sector, i també reduir el contacte entre cambrers i clients. Es vol realitzar una aplicació multiplataforma, però per falta de temps es realitzarà només per a dispositius Android. Tot i això, s'implementarà un backend que realitzarà la majoria de processos perquè sigui senzill fer l'aplicació per a diferents dispositius. Aquest projecte es realitzarà seguint una metodologia àgil basada en tres iteracions, i marcant uns objectius assequibles a curt temps.

Abstract (in English, 250 words or less):

Easy Order is a Mobile app whose objective is to automate and modernize restaurants. The objective is to solve the problem of the service offered by some restaurants, allowing the service to be more efficient. In addition, with the currently global pandemic that is being experiences, it is the right time to modernize this sector. It allows no reduce the contact between the waiter and the client. It is pretended to do a multiplatform app, but due to lack of time it will be done only for Android devices. Even so, a backend server will be implemented that will perform most of the processes so that it makes easy to make the application for different devices. This project will be done following an agile methodology based on three iterations, and setting some objectives achievable in the short term.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

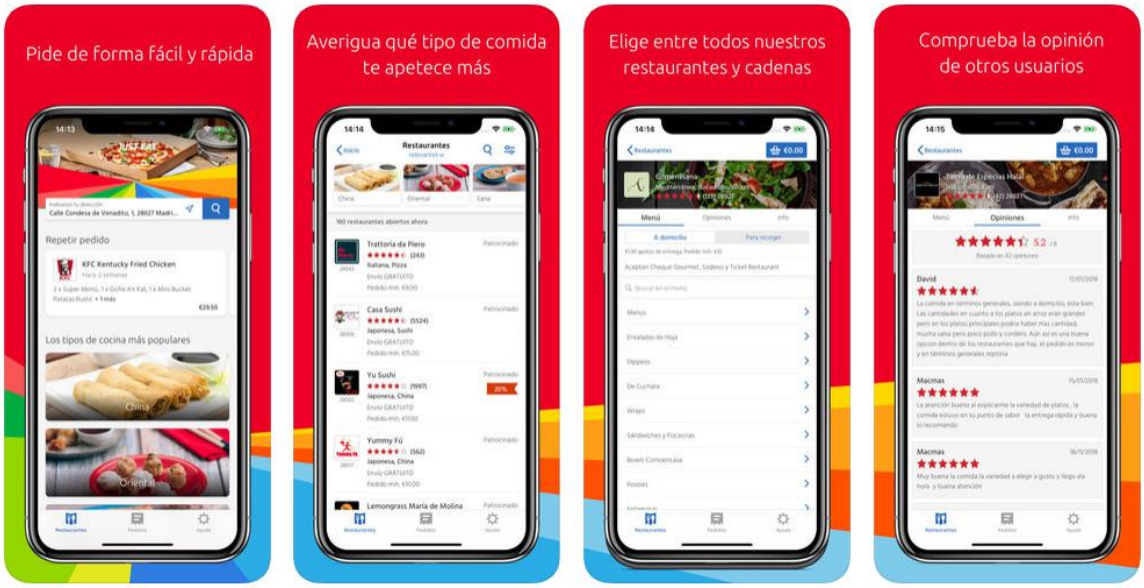
A causa de l'actual pandèmia de la Covid el sector de la restauració s'ha modernitzat una mica amb la introducció de codis QR per a la visualització dels menús dels restaurants. Tot i això, el sector de la restauració segueix estant antiquat i podria modernitzar-se molt més, permetent automatitzar alguns processos dels restaurants, com per exemple fer una comanda, o pagar el compte. Automatitzar aquests processos permet optimitzar i millorar el servei dels restaurants.
El sector de la restauració és un dels més grans del país i un dels principals motors econòmics. El servei donat als restaurants és igual a any enrere, s'ha modernitzat molt poc. Això provoca que si un restaurant vol oferir un servei òptim, necessiti una gran quantitat de cambrers (segons la grandària del restaurant). Qui no ha anat a un restaurant i ha estat més temps del necessari per a fer la comanda o perquè l'atenguin. Per això, automatitzant algun dels processos de la restauració milloraria molt la qualitat d'aquest sector.

Hi ha altres aplicacions de gestió de restaurants semblants a la solució que es vol implementar, però no són iguals, no són tan útils, ni aporten una modernització tan gran al sector. A continuació s'analitzen algunes aplicacions que s'assemblen:

- **ElTenedor Manager**: El tenedor manager és una aplicació exclusiva per a propietaris de restaurants que permet optimitzar el dia a dia del restaurant. Principalment permet gestionar les reserves en temps real, modificar i afegir noves. És una aplicació que permet optimitzar els restaurants, però a diferència de Easy Order és una aplicació que només està dirigida a propietaris dels restaurants. En canvi, Easy Order també està dirigida a clients de restaurants. Les funcionalitats de Easy Order són més àmplies que les de ElTenedor Manager.

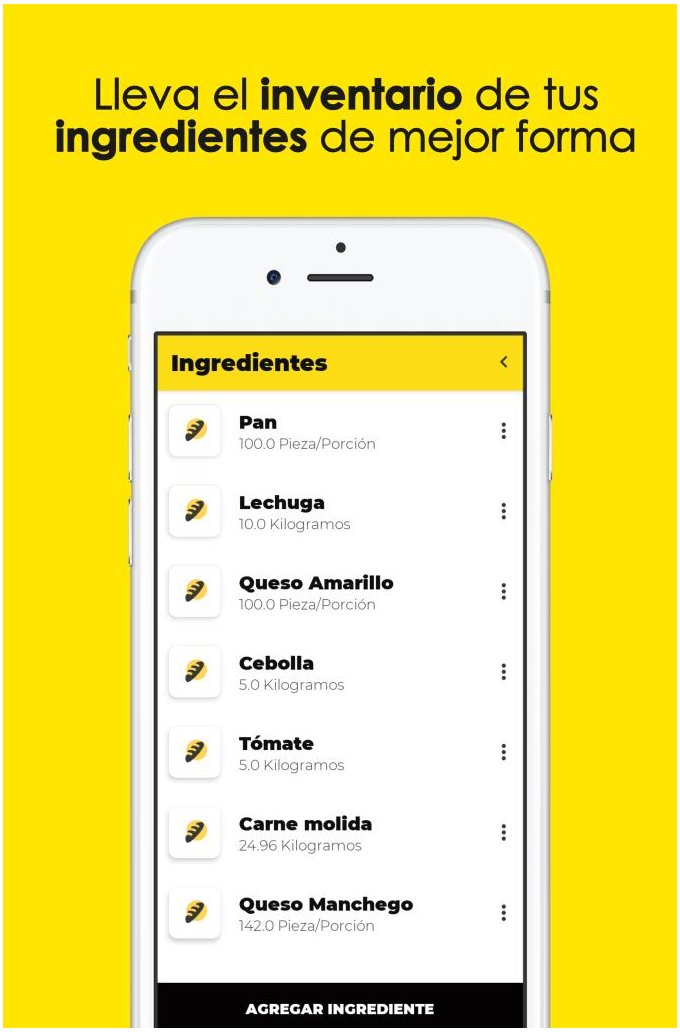
Ilustració 1 - Captura de ElTenedor Manager

- **Just-eat**: Just eat és una aplicació molt famosa, molt consolidada al mercat amb una gran quantitat d'usuaris. Tot i això, no és un competidor directe, ja que les funcionalitats que ofereix no són exactament les mateixes que ofereix Easy Order. Les funcionalitats de Just-eat estan orientades a demanar menjat a domicili, d'altra banda les funcionalitats d'Easy Order estan orientades a la gestió i l'automatització dels restaurants de forma presencial. Just eat permet buscar els restaurants que tenen servei a domicili a la teva direcció i permet fer una comanda perquè un repartidor ho porti al domicili de l'usuari.



Ilustració 2 - Captures de Just eat

- **Mesereando**: Aquesta aplicació serveix per a gestionar i administrar l'inventari i estoc d'un restaurant. Permet registrar dos tipus d'usuaris propietari o treballador. Permet visualitzar les comandes, crear la carta del restaurant, crear un inventari i que els ingredients vagin disminuint segons les comandes, visualitzar les estadístiques de vendes. Aquesta aplicació no és gaire famosa i en comparació amb les altres, tot i això, té més de 10 mil descàrregues. Està orientada als propietaris dels restaurants, ja que no hi ha opció per a clients. Aquesta és la principal diferencia entre Easy Order i Mesereando, ja que a Easy Order es permet als clients fer les comandes utilitzant l'app, mentre que a Mesereando són els treballadors els que han de fer tot. Mesereando és més una aplicació d'ús privat o intern a nivell de restaurant, d'altra banda Easy Order és d'ús públic. A part que hi ha varies diferències en les funcionalitats ofertes.



Ilustració 3 - Captura de Mesereando

Analitzant diferents aplicacions disponibles al mercat es pot observar que la majoria són d'ús exclusiu per a propietaris de restaurants, i no optimitzen gaire el procés d'un restaurant. En canvi, amb Easy Order es pretén estalviar temps als cambrers a l'hora de prendre la comanda, portar el compte... La nova aplicació estarà destinada tant a clients com a treballadors.

1.2 Objectius del Treball

L'objectiu principal d'aquest projecte és crear una aplicació que permeti gestionar, automatitzar i agilitzar el procés d'un restaurant. Es pretén que la mateixa aplicació pugui ser utilitzada pels clients a tots els restaurants que vagin, per això és important que una gran quantitat de restaurants vulguin registrar el seu restaurant a l'aplicació.

La interfície gràfica de l'aplicació ha de ser molt intuïtiva i fàcil d'usar, ja que els clients dels restaurants abasten un ampli rang d'edats i coneixements informàtics. L'aplicació ha de poder usada per tots els clients sense problemes.

El projecte és un projecte ambiciós i que es pretén que estigui en constant procés d'ampliació i modernització. Per això, es separen els requisits funcionals en tres grups, el primer grup són els indispensables, el següent grup són els requisits opcionals i l'últim grup són les funcionalitats que queden com a propostes de millora, per a realitzar en un futur.

També es definiran les funcionalitats segon si són per a usuaris treballadors de restaurants o per a usuaris clients.
Inicialment, es vol que tant els clients de restaurants com els treballadors es puguin crear un usuari i poder iniciar sessió per a poder accedir a les funcionalitats de cada tipus d'usuari.

Per part del restaurant, ha de poder introduir la carta del restaurant amb tota la informació dels plats (preu, al·lèrgies, ingredients, calories, descripció, fotografies…). A més, es permetrà afegir menús diaris, de grups, descomptes…

Cada taula d'un restaurant tindrà un número identificador i els treballadors dels restaurants han de poder observar quines taules estan ocupades, rebre alertes si una taula necessita assistència (opcional), rebre les comandes de les taules sense necessitat de prendre la comanda de manera manual. Això permet proporcionar un servei més ràpid, ja que els cambrers es poden centrar a servir les taules i no han d'anar preguntant taula per taula si ja saben que volen demanar.

Una funcionalitat que serà part de les propostes de millora és permetre als restaurants elaborar informes (diaris, setmanals i mensuals) utilitzant les dades dels plats més demanats, amb més puntuació, els menys demanats, i més informació amb l'opció de poder visualitzar la informació basant-se en l'edat dels clients. Això permet als restaurants potenciar els seus punts forts i millorar en els punts més febles, i d'aquesta manera millorar el servei donat al restaurant.

D'altra banda, per part dels clients dels restaurants han de ser capaços d'introduir el codi de la taula (o escanejant un codi QR, funcionalitat opcional) ocupar una taula del restaurant. Un cop un client ha ocupat una taula ha de poder fer una comanda sense haver d'esperar al cambrer. A més, si el client selecciona algun plat el qual porta algun ingredient del qual és al·lèrgic, li saltarà una notificació. A més, el client ha de poder afegir qualsevol mena d'observació sobre el plat demanat.

Com a funcionalitat opcional, seria interessant que un client fos capaç de demanar l'assistència d'un cambrer mitjançant un botó.

El client ha de ser capaç de poder pagar el compte sense necessitat d'un cambrer mitjançant l'aplicació. Es pretén proporcionar diferents mètodes de pagament: amb targeta, paypal, bizum, transferència (opcional) i com a proposta de millora, seria interessant permetre el pagament amb criptomones. Un requisit important a l'hora de pagar és que sigui molt segur, perquè els usuaris puguin confiar sense problema.

Una altra proposta de millora és fer un buscador de restaurants segons tipus de menjar, localització, valoració, preu, entre altres. Pel moment no s'implementarà aquesta opció, ja que hi ha diverses aplicacions que permeten fer-ho, tot i això, és una funcionalitat interessant per a implementar en un futur.

També pot ser interessant en un futur permetre buscar treballadors per als restaurants. I permetre als treballadors enviar el seu currículum. Per a poder concertar entrevistes i facilitar la contractació.

Per últim, com a proposta de millora es vol que sigui una aplicació multiplataforma i multi-idioma per tal d'arribar a la major quantitat de públic. Per la falta de temps a l'hora de realització del projecte, només es realitzarà amb un idioma i per a Android, però la manera d'implementar anirà orientada a poder ampliar en un futur.

Indispensables	**Opcionals**	**Propostes de millora**
Crear usuari	Client: Sol·licitar un cambrer amb un botó. Treballador: Rebre alertes si una taula necessita assistència.	Treballador: Poder elaborar informes (diaris, setmanals, mensuals...) amb les dades del restaurant.
Login	Client: Poder ocupar taula analitzant un codi QR. Treballador: Poder generar codis QR per a les taules del restaurant.	Client: Poder pagar amb criptomonedes
Client: Visualitzar menú. Treballador: Crear, modificar, eliminar menú del restaurant.	Client: Poder pagar amb diferents mètodes de pagament; targeta, bizum, transferència, paypal...	Client: Poder cercar restaurants segons diferents paràmetres.
Client: Ocupar taula. Treballador: Poder veure quines taules estan ocupades.	Client: Fer llista d'aliments preferits per a crear suggeriments personalitzats.	Treballadors: Poder buscar treball enviant currículum, concertar entrevistes...
Client: Fer la comanda. Treballador: Rebre les comandes.	Client: Reservar taula fent ús de l'app.	App: Multiplataforma
Client: Indicar els aliments als quals és al·lèrgic.		
Client: Alerta en cas de seleccionar un plat que contingui un aliment al		

qual és al·lèrgic		
Client: Poder pagar el compte sense necessitar un cambrer.		
Client: Poder pagar cadascú la seva part del compte, en el cas d'anar en grup, de manera senzilla		
Afegir/eliminar treballadors al restaurant		

1.3Enfocament i mètode seguit

S'ha decidit que el millor és desenvolupat un producte nou des de zero, ja que això permet una major flexibilitat, i és més fàcil per a obtenir un producte més semblant a l'esperat. S'implementarà un backend amb Java i Spring que es comunicarà amb Firebase. Això serà complementat per l'aplicació Android que s'alimentarà de la informació rebuda del backend i només haurà de presentar-la. En realitzar la majoria d'operacions en un servidor backend permet en un futur poder implementar l'aplicació per a dispositius iOS i l'aplicació web.

El projecte serà realitzat seguint una metodologia àgil utilitzant històries d'usuari i fixant objectius concrets i realitzables a curt termini. El software serà desenvolupat de forma iterativa amb iteracions curtes.

El primer sprint tindrà una durada de 15 dies, unes 77 hores de treball. Començarà el 01/04/2021 i finalitzarà el 15/04/2021. El següent sprint durarà 14 dies, unes 74 h de treball. Amb inici el 16/04/2021 i finalitza el 29/04/2021. Per finalitzar, l'últim sprint tindrà una durada de 13 dies, unes 71 hores. Inicia el 30/04/2021 i finalitza el 12/05/2021.

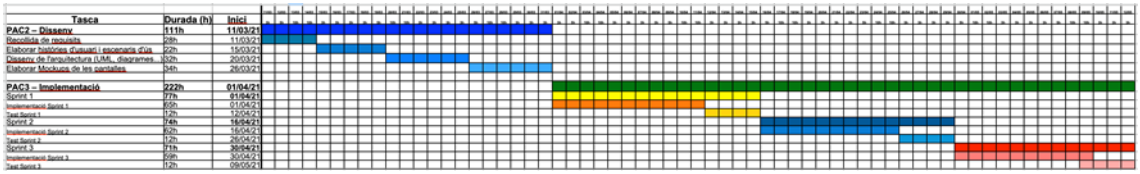
Es definiran unes històries d'usuari concretes per a cada sprint. Cada història s'haurà d'analitzar en detall, implementar, testejar i integrar a la resta d'aplicació en el mateix sprint. Aquestes històries d'usuari es definiran i es seguirà el seu desenvolupament utilitzant una eina que es diu Jira (on es guardarà el backlog).

Per al control de versions es farà ús de Github. Seguint una metodologia de branques GitFlow.

Per a tenir un feedback més real del software al final de cada iteració es donarà a potencials usuaris una versió de l'aplicació. Els usuaris actuaran com a testers i provaran l'aplicació. Finalment, hauran d'omplir una enquesta sobre les proves realitzades, respecte a l'aparença de l'aplicació, la dificultat d'ús, si han trobat algun bug o error i per últim, si tenen alguna proposta de millora.

1.4Planificació del Treball

Abans de començar amb la realització del diagrama de Gantt es defineixen les hores dedicades al projecte cada dia. Com que cada dia laborable treballo fins a les 18:00 h, els dies laborables només serà possible 3 h, excepte els divendres que seria possible dedicar 5h. D'altra banda els dies no laborables es dedicaran unes 10 h diàries. A partir d'aquesta planificació s'extrau el següent diagrama de Gantt.



Ilustració 4 - Diagrama de Gantt

1.5 Breu sumari de productes obtinguts

A continuació es mostra una llista amb els productes obtinguts en el projecte:

- Memòria del projecte
- Easy Order App Android
- Backend de l'aplicació Android (API REST)

1.6 Breu descripció dels altres capítols de la memòria

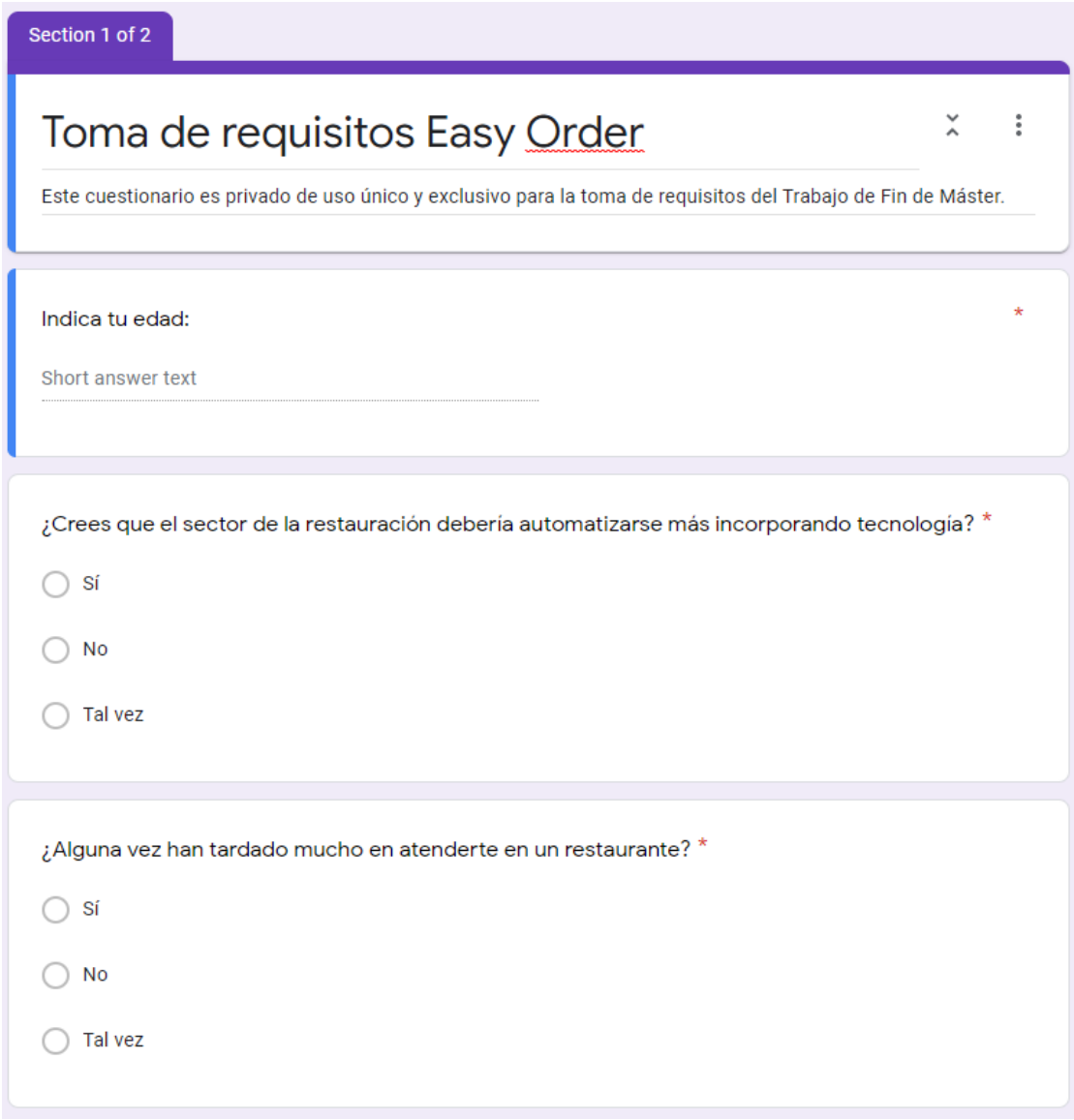
A continuació es farà una mena de resum dels següents capítols de la memòria. Inicialment s'explica detalladament el procés de la definició de requisits del projecte. A continuació, es defineixen els prototips d'usuaris de l'aplicació i els possibles escenaris. El tercer apart es tracta del prototipat de les pantalles de l'aplicació, on es mostren els wireframes tant a baix nivell com alt nivell. Seguidament es defineixen els casos d'ús de l'aplicació. El cinquè apartar s'explica l'arquitectura del sistema. Per finalitzar, els dos últims apartats s'explica el procés d'implementació amb les decisions que s'han pres a cada cas, i per últim s'explica el procés de testeig de l'aplicació.

2. Resta de capítols

2.1 Definició de requisits

Per tal de realitzar la presa de requisits s'elabora un qüestionari per tal que usuaris potencials de l'aplicació el responguin i per a poder definir els principals requisits.

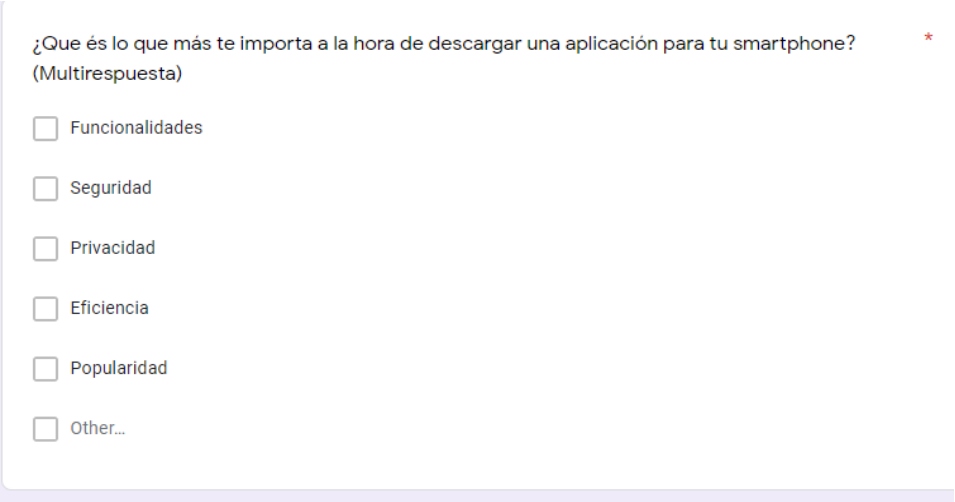
A continuació es mostren les preguntes del qüestionari. Té dues parts, una part comuna per a tothom (ja que tothom pot ser client d'un restaurant) i una part que només és per a treballadors de restaurants.



Ilustració 5 - Formulari requisits 1

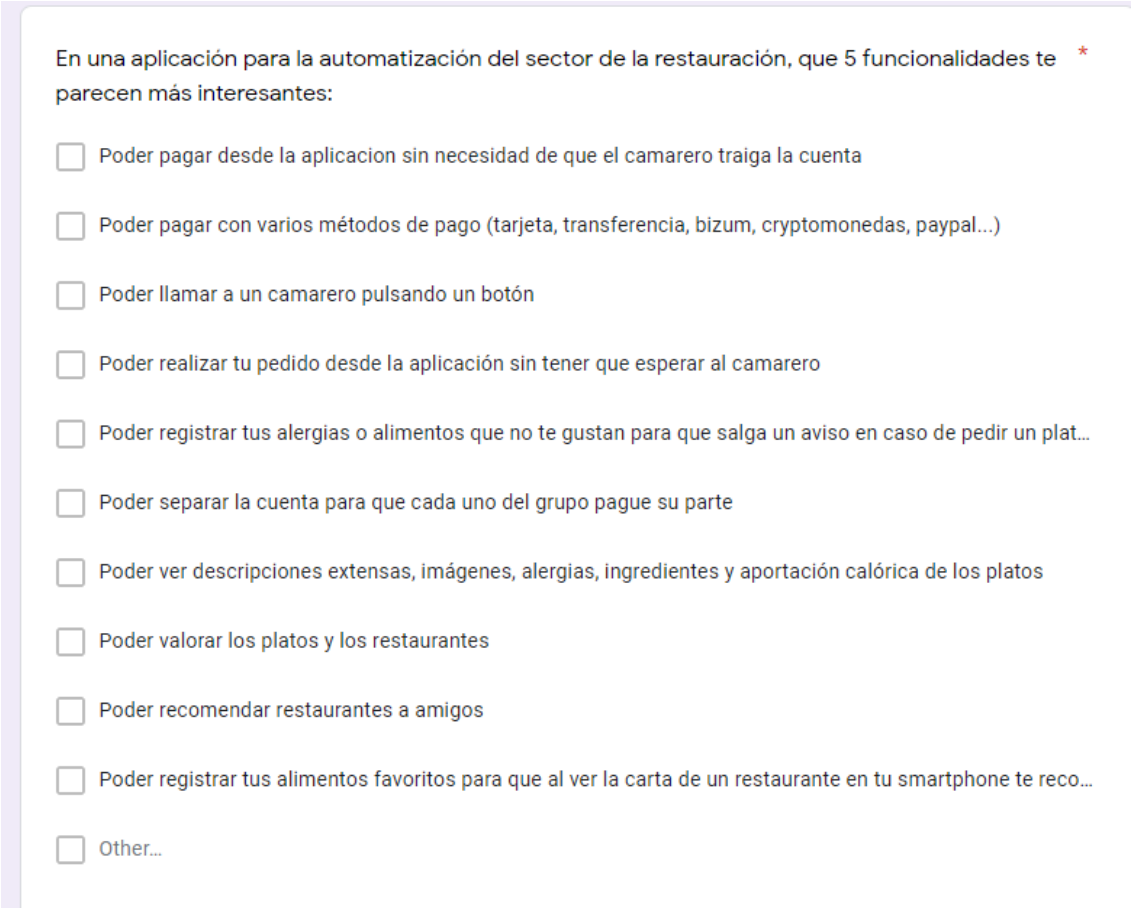
Inicialment es pregunta l'edat per a poder veure les necessitats o interessos segons l'edat de l'usuari i així poder separar una mica segons la demografia.

A continuació, es pregunta si es creu que el sector de la restauració hauria d'automatitzar-se amb la incorporació de tecnologia. Amb aquesta pregunta es pretén descobrir si els usuaris estan oberts a introduir la tecnologia a més contextos de la seva vida quotidiana com pot ser anar a un restaurant.
A la segona pregunta es pregunta si han tingut problemes amb el temps d'atenció en un restaurant, per a veure si l'aplicació satisfà una necessitat real que li passa a la majoria d'usuaris o no.



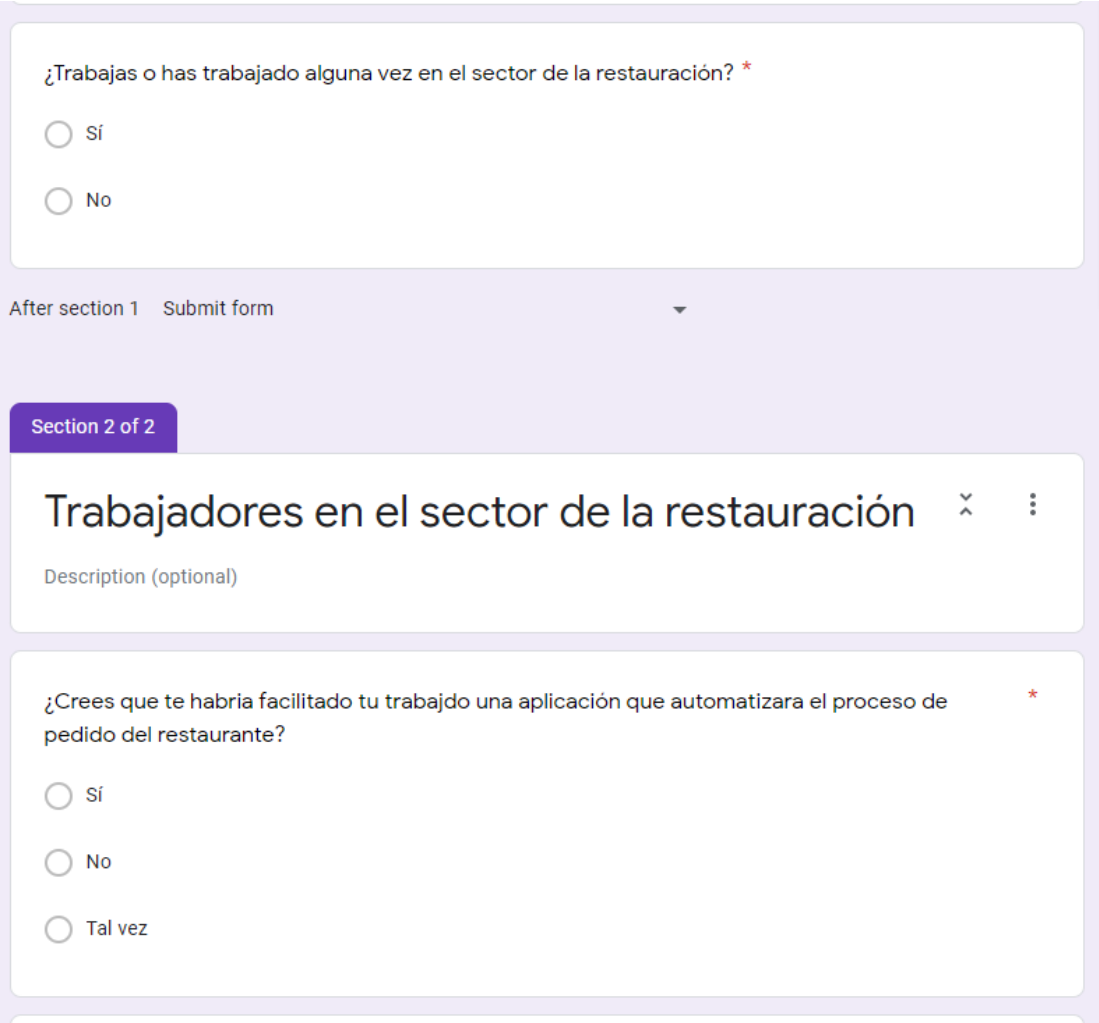
Ilustració 6 - Formulari requisits 2

Seguidament es pregunta què és el més important per a l'usuari a l'hora de descarregar una nova aplicació. Això permet definir els requisits no funcionals de l'aplicació, per a poder centrar-se a implementar funcionalitats innovadores, o centrar-se en la seguretat, la privacitat… A més, es permet a l'usuari introduir noves opcions.



Ilustració 7 - Formulari requisits 3

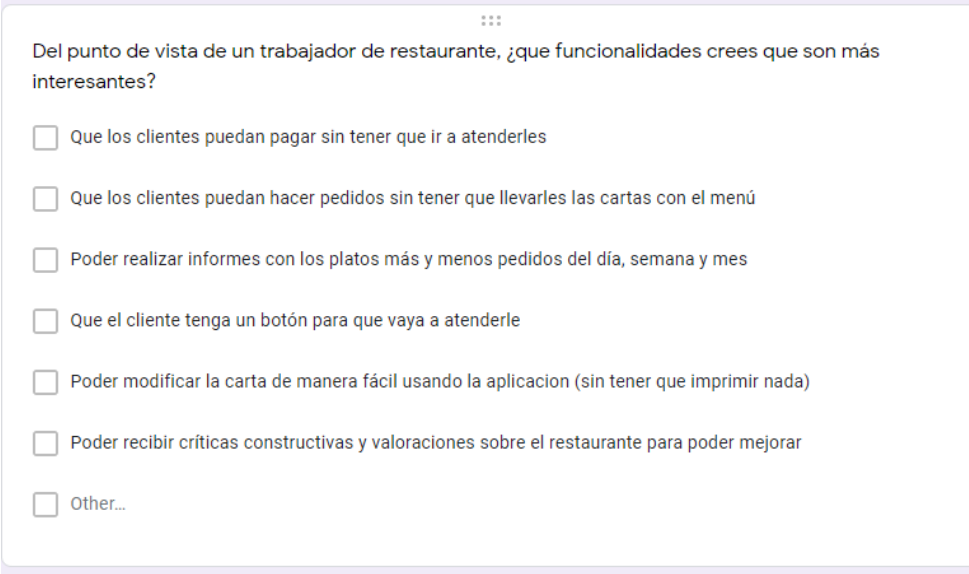
A continuació, es fa una de les preguntes més importants que serveix per a definir les preferències dels usuaris i permet prioritzar les funcionalitats. A l'usuari se li presenten diferents funcionalitats que pot incorporar l'aplicació i l'usuari ha de decidir les 5 més importants. A més es permet afegir alguna funcionalitat extra.



Ilustració 8 - Formulari requisits 4

L'última pregunta per als usuaris comuns es pregunta si han treballat al sector de la restauració. En cas de respondre negativament, s'acaba el formulari, en cas de respondre afirmativament es passa a la segona part del formulari, especifica per a treballadors.

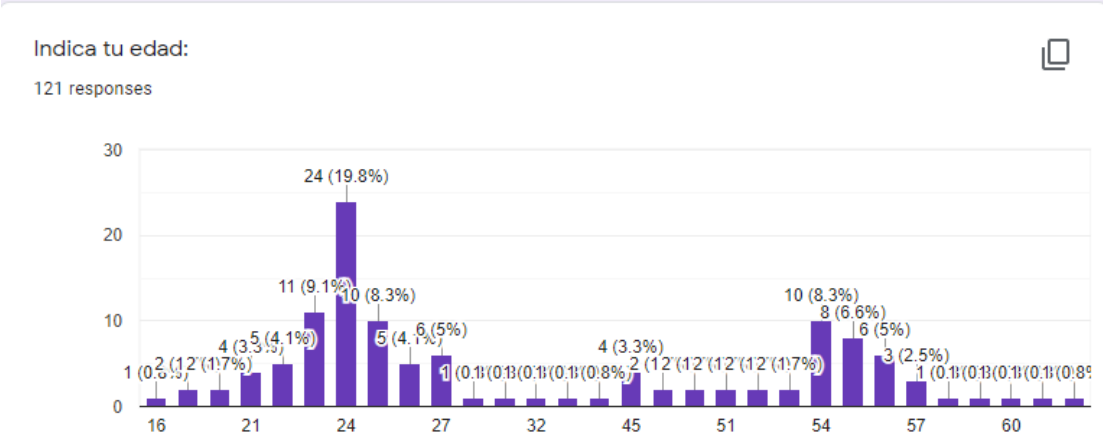
A l'inici de la segona part es pregunta al treballador si creu que una aplicació per a automatitzar el procés de comanda d'un restaurant hi hagués facilitat el seu treball. D'aquesta manera, es pot saber si l'aplicació a part de ser útil per als clients, ho és per als treballadors.



Ilustració 9 - Formulari requisits 5

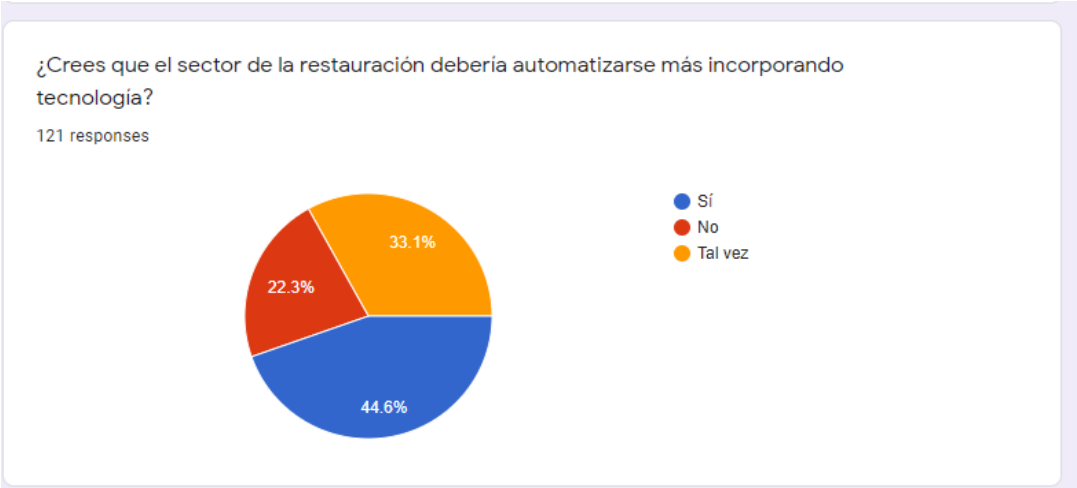
Per finalitzar, es pregunta als treballadors quines funcionalitats creu que serien més interessants per a facilitar el seu treball. A part d'unes predefinides es permet a l'usuari poder introduir alguna nova.

Aquest qüestionari s'ha entregat a diversos usuaris de diferents edats i sexe per a poder fer la presa de requisits. Després de 121 respostes s'han aconseguit els següents resultats.



Ilustració 10 - Respostes requisits 1

Es pot observar que hi ha respostes de totes les edats, però esta més concentrat en usuaris entre els 20 i 30 anys.

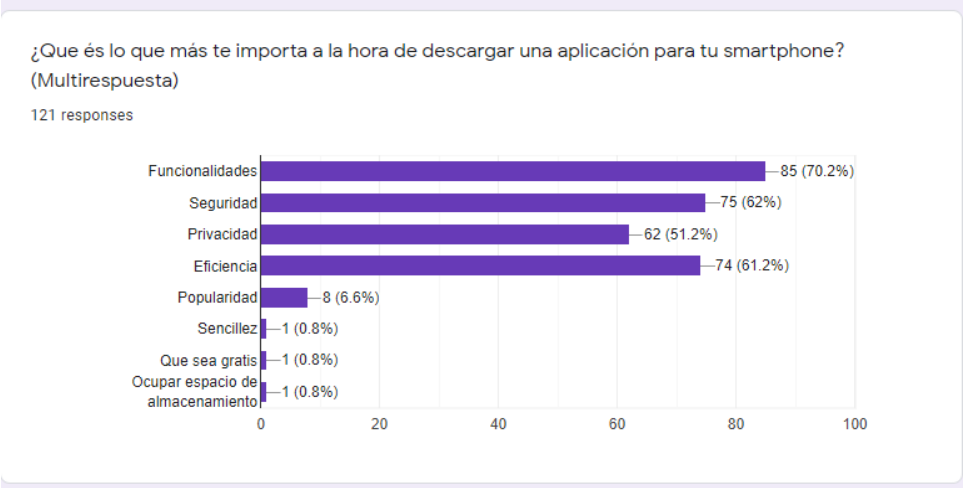


Ilustració 11 - Respostes requisits 2

Inicialment, es pot observar com la majoria d'usuaris creuen que s'ha d'incorporar tecnologia al sector de la restauració (44.6%). D'altra banda, un 22.3% creu que no. I un 33.1% pensa que pot ser, segons les funcionalitats que aporti. Analitzant amb més profunditat, s'ha pogut observar que la gran quantitat d'usuaris que han votat que no coincideix amb els usuaris més grans. Segurament per por a no saber utilitzar l'aplicació, per això es conclou que és important que l'aplicació sigui intuïtiva i fàcil d'usar sense importar l'experiència tecnologia de l'usuari, ja que serà una aplicació que l'usaran diferents tipus d'usuaris.

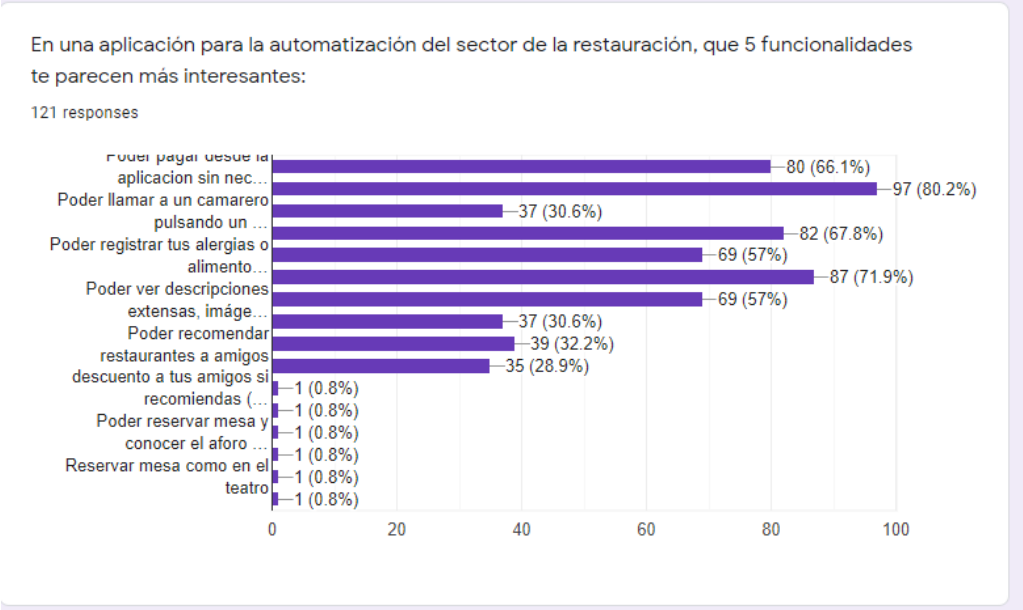
Ilustració 12 - Respostes requisits 3

A la pregunta de si han trigat molt a atendre'ls en un restaurant, es pot observar com la immensa majoria ha tingut alguna mala experiència en un restaurant. Un 95% dels usuaris preguntats, permeten afirmar que l'automatització d'aquest sector és una necessitat real.



Ilustració 13 - Respostes requisits 4

A la pregunta per definir els requisits no funcionals es pot observar com el més important són les funcionalitats que ofereix l'aplicació, la seguretat, l'eficiència i la privacitat. Alguns usuaris han afegit alguna resposta com que sigui senzilla i que sigui gratuïta. Dues coses importants a tenir en compte, ja que segurament la majoria d'usuaris no vulguin pagar per fer ús de l'aplicació, ni vulguin que sigui més complicat fer ús de l'aplicació que del mètode tradicional.

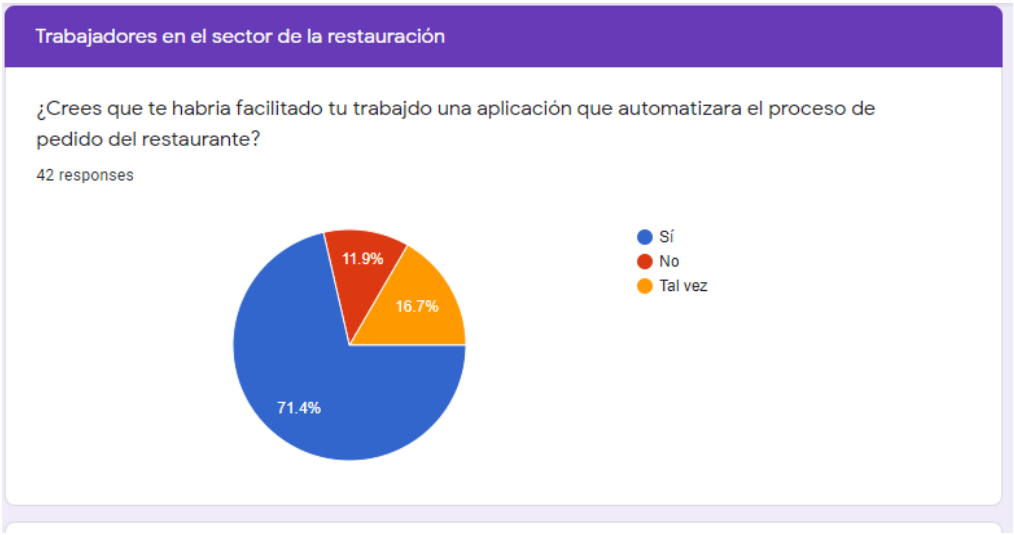


Ilustració 14 - Respostes requisits 5

Seguidament s'analitzen les dades obtingudes respecte a la pregunta de les funcionalitats. La funcionalitat més important és la de poder pagar amb diferents mètodes de pagament fent servir l'app, seguidament de poder separar el compte perquè cadascú pagui la seva part quan es va en grup. La tercera funcionalitat més demanada és poder fer la comanda utilitzant l'aplicació, seguida de poder pagar amb l'app sense necessitat d'esperar al cambrer. Aquestes quatre funcionalitats solucionen problemes reals que els usuaris es troben a l'hora d'anar a un restaurant.

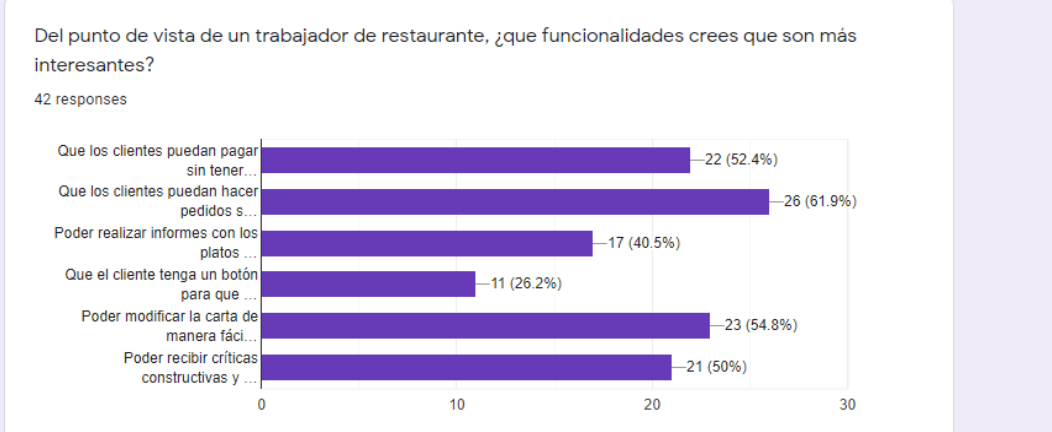
Seguidament, es demana que es pugui configurar les al·lèrgies que té un usuari per evitar demanar plats amb aquells ingredients, i que els plats estiguin descrits a fons, amb imatges i l'aportació calòrica.
Per últim, les funcionalitats menys demanades són poder avisar al cambrer usant un botó, poder valorar plats i restaurants i poder recomanar restaurants a amics (estil xarxa social). Aquestes són les funcionalitats menys prioritàries. Una funcionalitat extra que han demanat diversos usuaris és la de poder reservar taula des de l'aplicació. És una funcionalitat interessant que es tindrà en compte com una funcionalitat opcional i si dóna temps s'implementarà.

A continuació s'analitzen les preguntes realitzades als usuaris treballadors en restaurants.



Ilustració 15 - Respostes requisits 6

Inicialment es pregunta si els treballadors creuen que una aplicació per automatitzar el proces de comanda seria útil i facilitaria el seu treball. En aquest cas, la gran majoria creuen que sí (71.4%). Un 16.7% opina que potser, depenent de les funcionalitats. I per últim, només un 11.9% creu que no li seria útil. Això és molt important, ja que es busca que l'aplicació sigui útil tant pels treballadors del restaurant com per als clients.



Ilustració 16 - Respostes requisits 7

Per finalitzar, es demana als treballadors quines són les funcionalitats que creuen més útils. Principalment les tres funcionalitats més votades són les següents. Permetre als clients poder fer la comanda sense necessitat de què el cambrer l'atengui, poder crear i modificar la carta amb l'aplicació de manera senzilla i per últim, que els clients puguin pagar sense que el cambrer hagi d'atendre'l. Seguidament, hi ha altres funcionalitats menys votades, com pot ser, rebre crítiques constructives i valoracions, poder fer informes amb els plats més demanats i per últim, la menys útil és incorporar un botó per a poder sol·licitar assistència d'un cambrer. Aquesta última funcionalitat és més útil en casos de restaurants més grans on no es té visibilitat de totes les taules.

Partint dels resultats d'aquest qüestionari s'han definit els requisits tant funcionals com no funcionals de l'aplicació.

2.2 Definició dels usuaris i escenaris

Es definiran dos usuaris tipus d'exemple que podrien fer ús de l'aplicació.

Usuari 1:

Ilustració 17- Usuari 1

Manel

Professió: Estudiant d'arquitectura /Cambrer a un restaurant els caps de setmana
Ingressos: Baixos-mitjos. Els únics ingressos d'en Manel són els de treballat mitja jornada els caps de setmana de cambrer.
Edat: 20 anys. Una persona jove que sap utilitzar i confia en les noves tecnologies.
Ciutat: Barcelona. Una ciutat molt gran amb una gran concentració de gent, que provoca que en Manel tingui molt de treball al restaurant.
Aficions: Sortir a dinar entre setmana amb els seus amics de la universitat per a desconnectar dels estudis.
Objectiu: Reduir l'estrès al treball, ja que és un restaurant molt exitós a Barcelona i té molt de treball.

Cas d'ús de l'aplicació (Escenari):
En Manel estudia 5 dies a la setmana i els caps de setmana treballa 8 hores cada dia. Porta un nivell d'estrès molt alt. Al restaurant on treballa han decidit implantar l'aplicació Easy Order per a automatitzar el procés de les comandes, reduir el treball dels seus treballadors, ser més eficient amb el servei i facilitar les coses als clients. En Manel ara no ha d'atendre als clients repartint els menús quan arriben, no ha d'estar pendent quan els clients estan llestos per demanar, ni ha de prendre les comandes. Es pot centrar a servir les taules, ja que rep les comandes automàticament al seu smartphone. El seu nivell d'estrès al treball s'ha reduït considerablement, permetent oferir un servei més eficient i més bo, permetent-li ser més simpàtic amb els clients.

Usuari 2:

Ilustració 18 - Usuari 2

Cristina

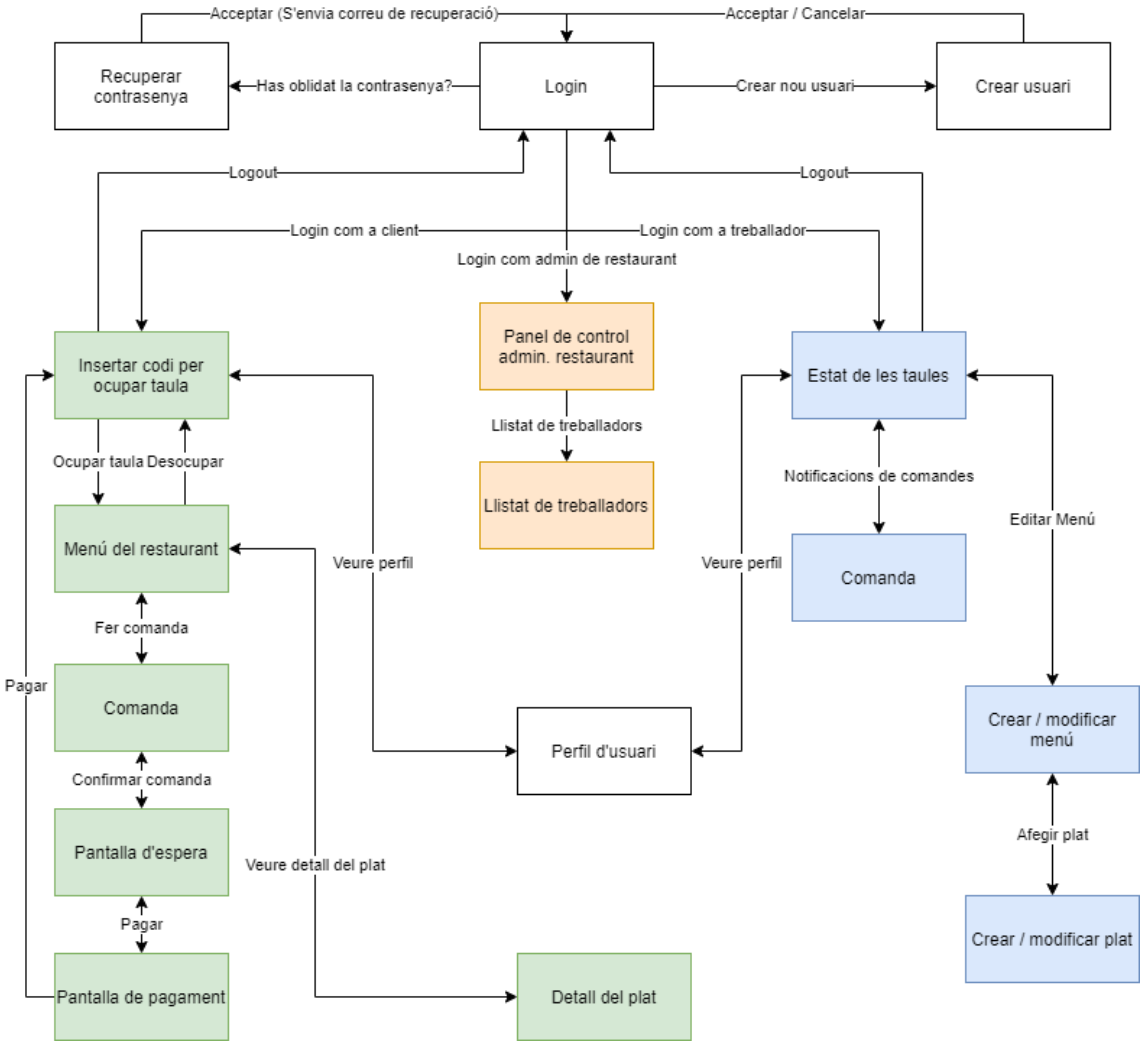
Professió: Periodista
Ingressos: Mitjos. Té un bon sou com a periodista.
Edat: 27 anys. Tot i ser jove, no és una experta en la tecnologia.
Ciutat: Barcelona. Una ciutat molt gran amb una gran concentració de gent, que provoca que en Manel tingui molt de treball al restaurant.
Aficions: Li encanta anar a dinar a restaurants amb els seus companys de feina en els descansos d'una hora que tenen en el treball.
Objectiu: Reduir el temps que es passa dinant, ja que hi ha restaurants que triguen més d'una hora.

Cas d'ús de l'aplicació (Escenari):
La Cristina s'ha descarregat l'aplicació d'Easy Order, ja que diversos restaurants de al voltant de la seva feina han implantat aquest sistema. A l'arribar al restaurant amb els seus companys es senten a una taula i introdueixen el codi de la taula per a ocupar la taula i poder iniciar la comanda. Automàticament l'aplicació mostra el menú del restaurant sense haver d'esperar a ser atesos per un cambrer. La Cristina i els seus companys fan la comanda del menjar i la beguda. Al fer la comanda, l'aplicació indica a la Cristina que ha demanat un plat que conté un aliment al qual és al·lèrgica, això permet a la Cristina canviar el plat. A l'instant d'acabar la comanda apareix el cambrer amb les begudes demanades. La Cristina i els seus companys respiren relaxats perquè veuen que el servei és ràpid i no arribaran tard al treball. Pocs minuts després el cambrer porta la resta de plats. Un cop han acabat amb els plats principals, la Cristina prem un botó per avisar al cambrer que ja han acabat i pot recollir els plats, a més fa la comanda de les postres. El cambrer recull els plats de la taula i procedeix a servir els postres. Finalment, la Cristina i els seus companys quan acaben de menjat volen procedir a pagar. Gràcies a Easy Order cadascú pot pagar el que s'ha demanat de manera senzilla sense haver de fer càlculs. Algun paga amb targeta, algun amb Bizum i

la Cristina decideix pagar amb Paypal. En 45 minuts han acabat de dinar i poden anar a donar un passeig abans de tornar al treball.

2.3 Prototipat

Abans de començar a fer els prototips de les pantalles, s'ha fet un arbre de navegació de les pantalles de l'aplicació per a poder deixar clares totes les pantalles que tindrà l'aplicació, i des de quines pantalles es podrà accedir.



Ilustració 19 - Arbre de navegació de les pantalles

Es poden veure 4 tipus de pantalles diferents. Les que estan representades en blanc són les que es poden accedir des de qualsevol mena d'usuari. Les verdes només poden accedir usuaris del tipus client, les blaves són per a usuaris treballadors de restaurants, i per últim les taronges són pantalles accessibles per a usuaris administradors de restaurant.

A partir d'aquest arbre de navegació s'han fet els primers esbossos de les pantalles de baixa fidelitat amb paper i llapis, i posteriorment s'han fet les pantalles d'alta fidelitat utilitzant Android Studio.

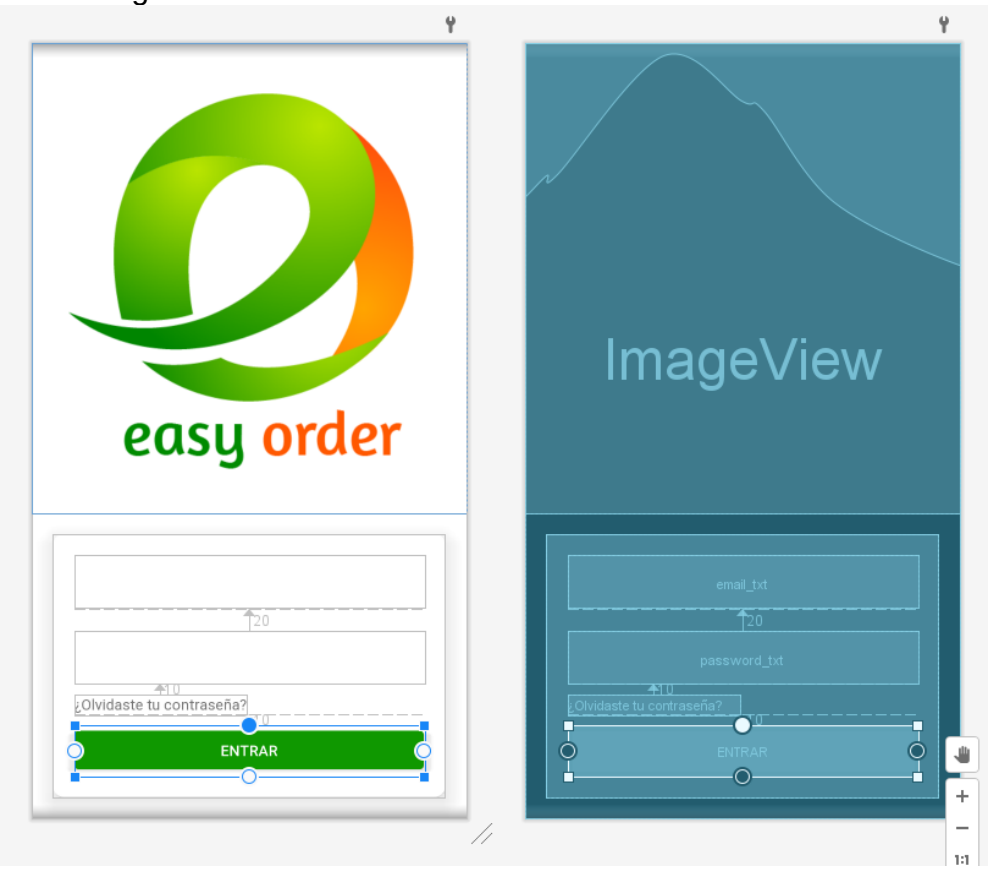
A continuació s'adjunta una imatge amb l'esbós de baixa fidelitat de les pantalles.

Ilustració 20 - Esbossos 1

Ilustració 21 - Esbossos 2

Partint d'aquests esbossos realitzats amb paper i llapis, s'han fet els wireframes d'alta fidelitat utilitzant Android Studio en un nivell més detallat.

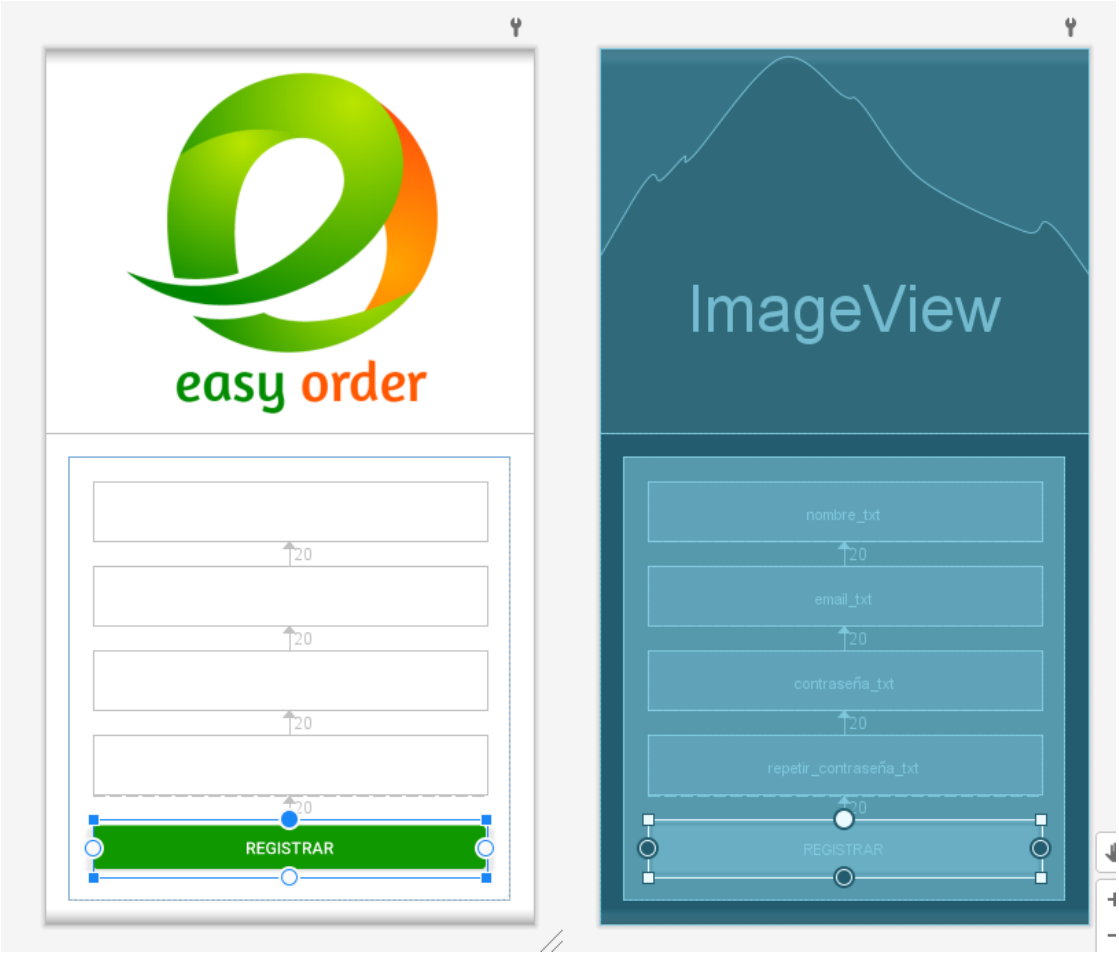
Pantalla de Login:



Ilustració 22 - Pantalla login

Aquest disseny es basa en la imatge del logo, dos camps de text per introduir l'email i la contrasenya, un enllaç per quan l'usuari ha oblidat la contrasenya i per últim el botó per a fer login.

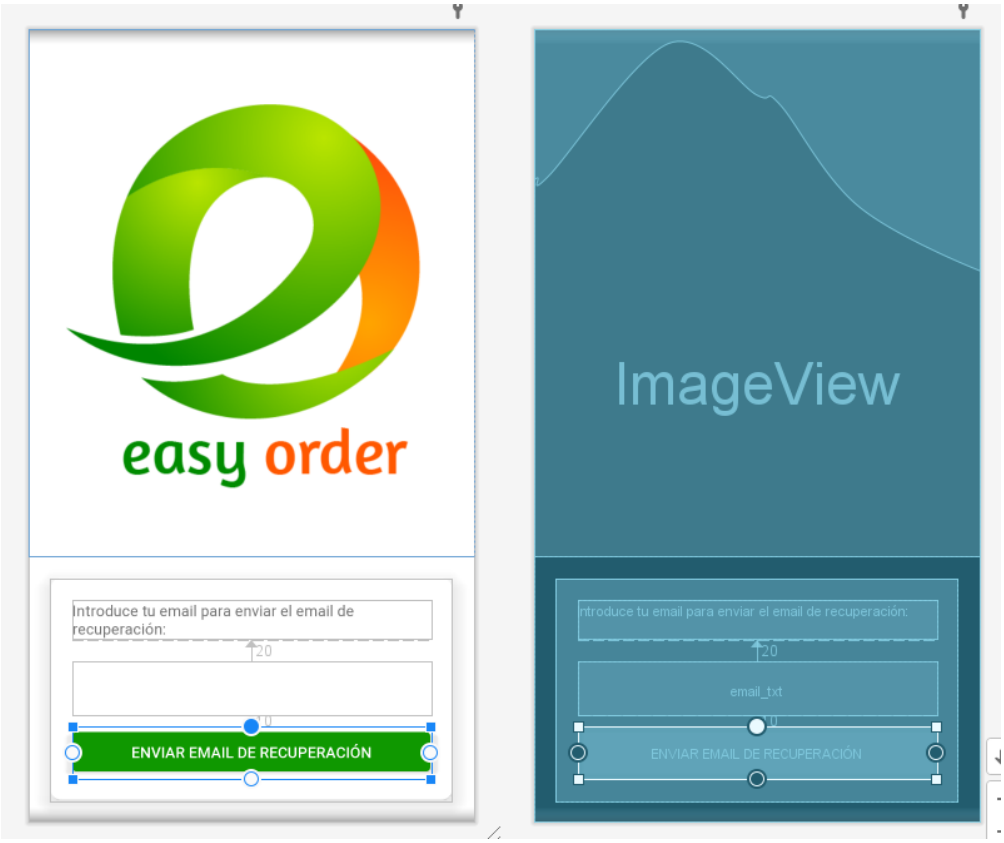
Pantalla Register:



Ilustració 23 - Pantalla register

El disseny d'aquesta pantalla consta del logo de l'aplicació, quatre camps de text (un pel nom de l'usuari, un altre per l'email, un per la contrasenya i l'últim per a repetir la contrasenya) i a baix de tot el botó per a registrar l'usuari.

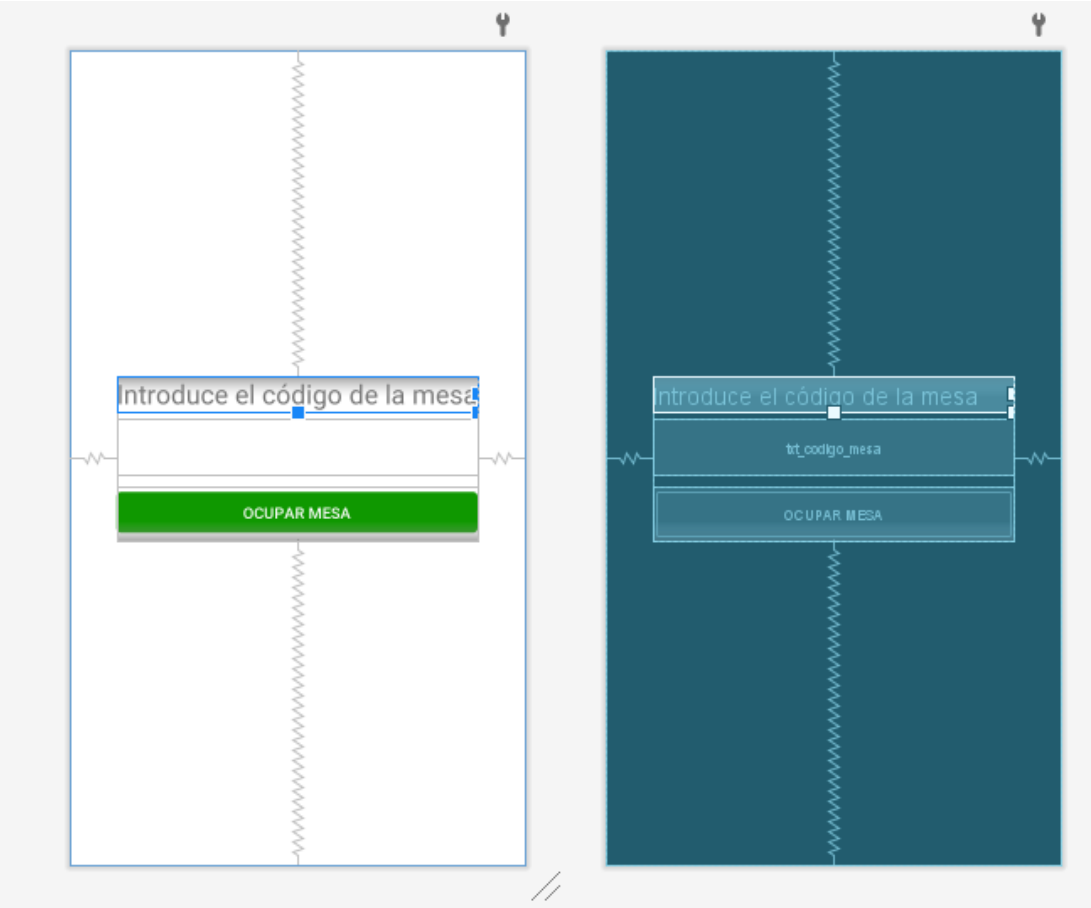
Pantalla recuperar contrasenya:



Ilustració 24 - Pantalla recuperació de contrasenya

Aquest disseny segueix la base de les altres dues pantalles, consta del logo de l'aplicació, un label indicant que s'ha d'introduir l'email per a enviar l'email de recuperació, el camp de text per a introduir l'email i per últim el botó per a enviar l'email de recuperació.

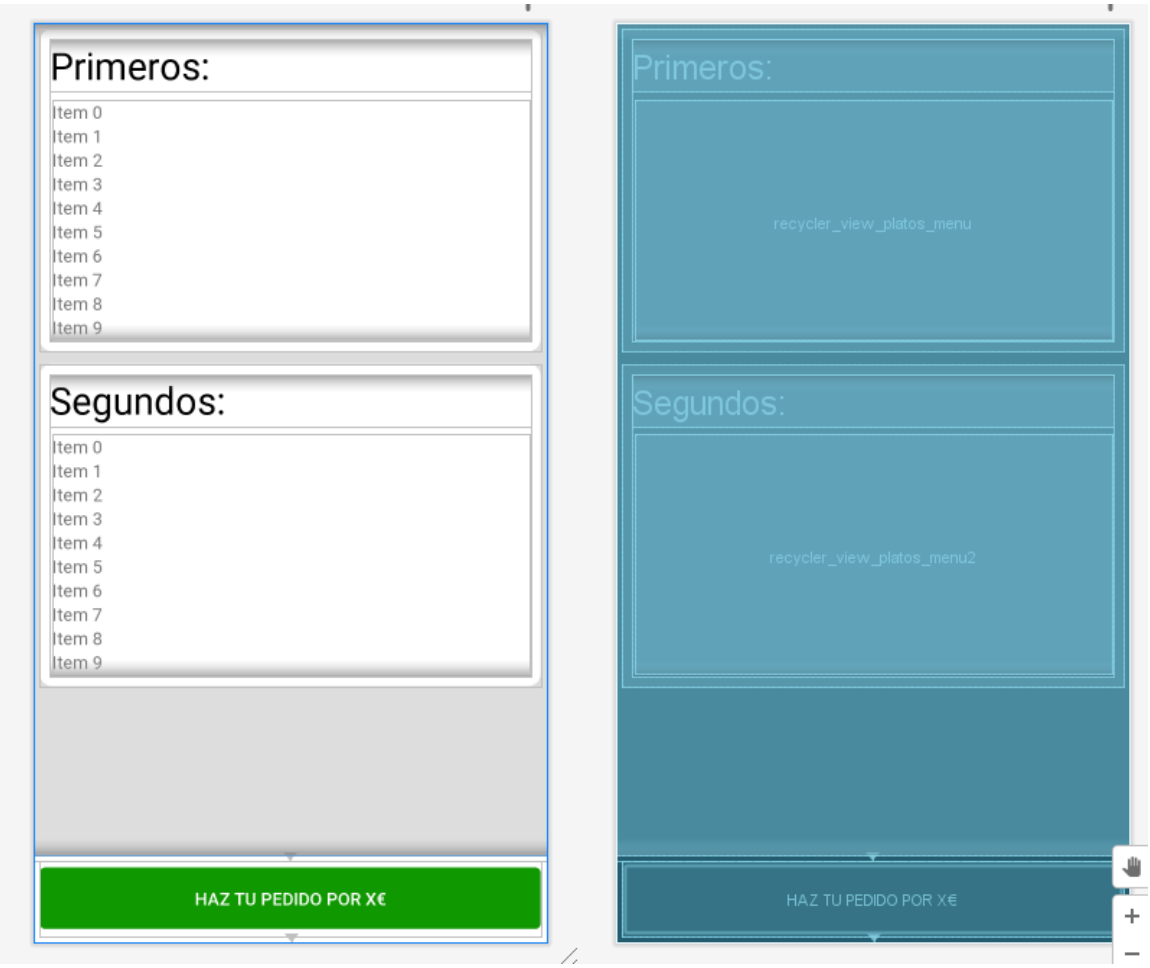
Pantalla ocupar taula:



Ilustració 25 - Pantalla ocupar taula

El disseny d'aquesta pantalla igual que el de totes les pantalles de l'aplicació s'ha fet molt senzill i perquè sigui molt intuïtiva, ja que ha de ser fàcil d'utilitzar sense importar l'edat de l'usuari o el nivell de coneixement de tecnologia. Per això aquest disseny només consta d'un camp de text on l'usuari ha d'introduir el codi de la taula que vol ocupar, i un botó per a ocupar-la.

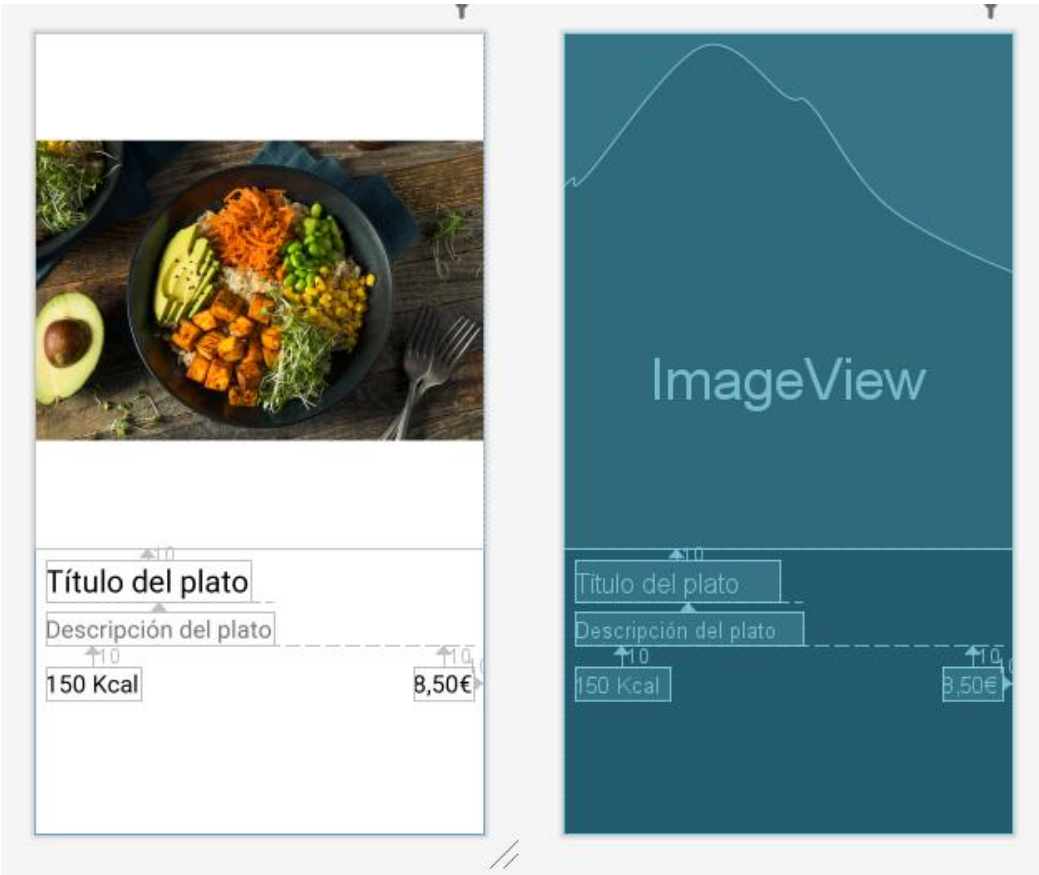
Pantalla menú:



Ilustració 26 - Pantalla menú restaurant

Aquesta pantalla dependrà del menú que hagi creat el restaurant. Es mostrarà un card_view per cada categoria (per exemple: una pels primers, una pels segons i un altre per a les begudes). Dins de cada card_view hi ha el títol de la categoria, i a sota un recycler view amb els plats d'aquesta categoria. L'usuari pot fer click a cada plat de la llista perquè s'obri un dialog preguntant si vol afegir el plat a la comanda. El botó de la part inferior anirà actualitzant el preu de la comanda cada cop que s'afegeixi un plat. Per finalitzar la comanda l'usuari ha de fer click al botó.
Aquest disseny s'ha basat en el disseny de l'app de Just Eat per a fer una comanda, ja que és una aplicació consolidada al mercat, i l'usuari ja sap com funciona, per això a l'hora d'utilitzar Easy order no haurà d'aprendre com funciona i sabrà utilitzar-la de manera intuïtiva.

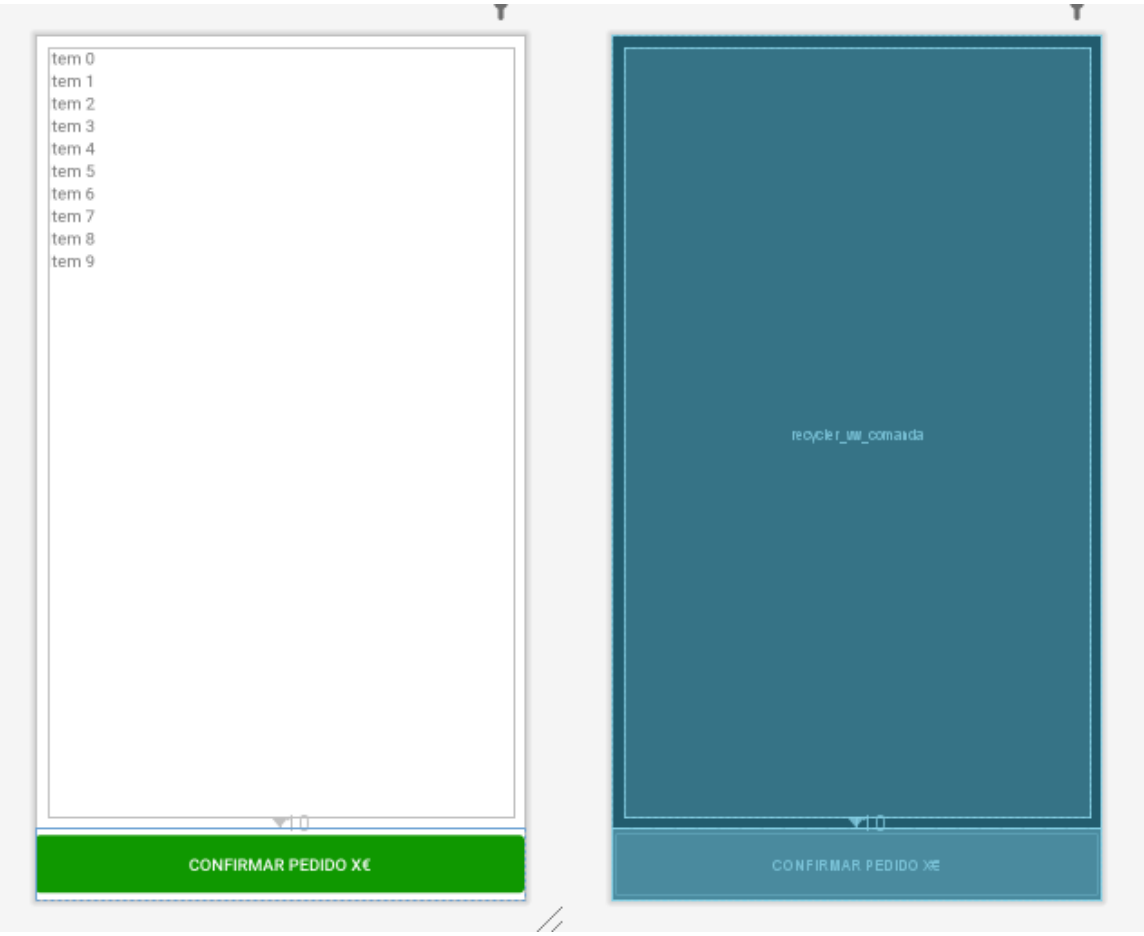
Pantalla detall plat:



Ilustració 27 - Pantalla detall plat

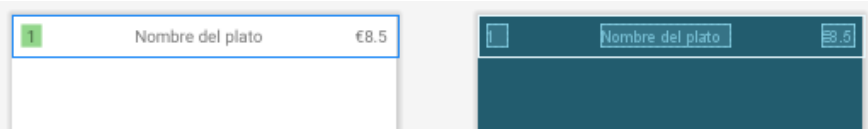
El disseny de la pantalla del detall del plat consta de la imatge del plat (si existeix), un label amb el títol del plat, un label menys destacat amb la descripció del plat i a la part inferior es mostren les calories del plat i el preu.

Pantalla comanda:



Ilustració 28 - Pantalla comanda

Aquesta pantalla és molt senzilla, es mostra una llista amb els plats demanats a la comanda i un botó per a confirmar la comanda on s'indica el preu de la comanda.



Ilustració 29 - Disseny llistat comanda

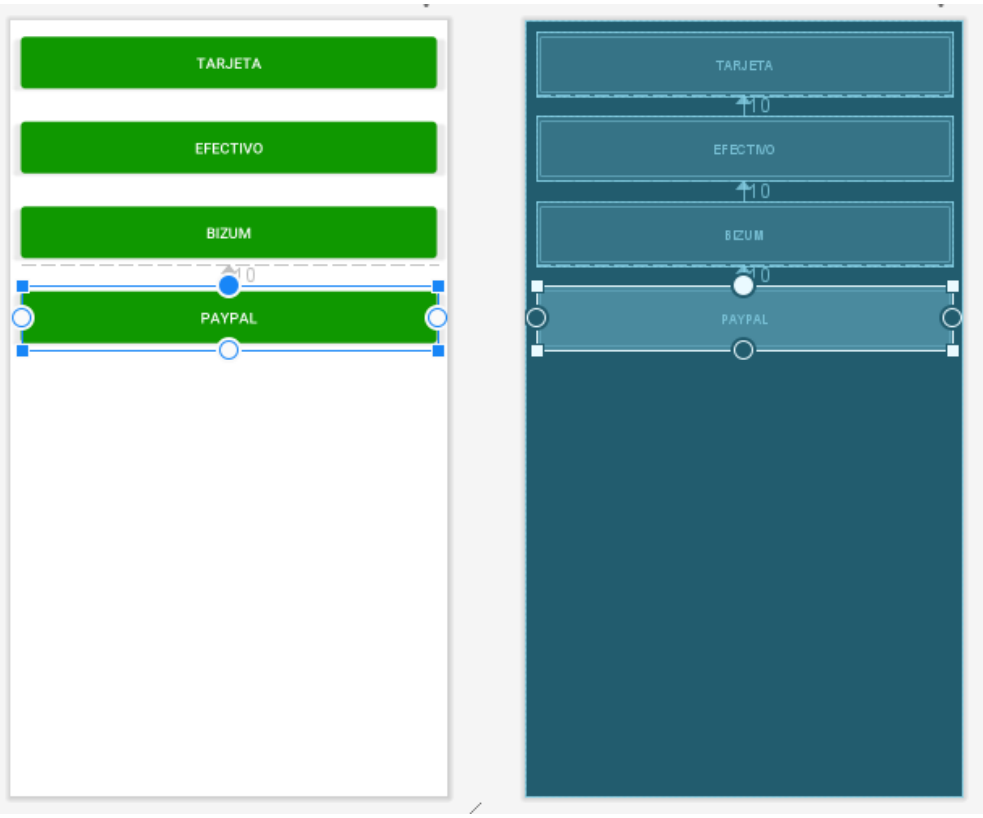
El format dels elements de dins de la llista és el que es veu a l'anterior imatge. Conté la quantitat de plats demanats, el nom del plat demanat i el seu preu.

Pantalla d'espera:

Ilustració 30 - Pantalla d'espera

Aquesta pantalla no té res especial, simplement és una pantalla d'espera mentre el restaurant prepara el menjar. Es mostra un gif cuinant i dos botons. El primer botó és per a editar la comanda, per si l'usuari vol afegir algun plat a la comanda. I l'últim botó és per quan l'usuari ha acabat de menjar, i accedir a la pantalla de pagament.

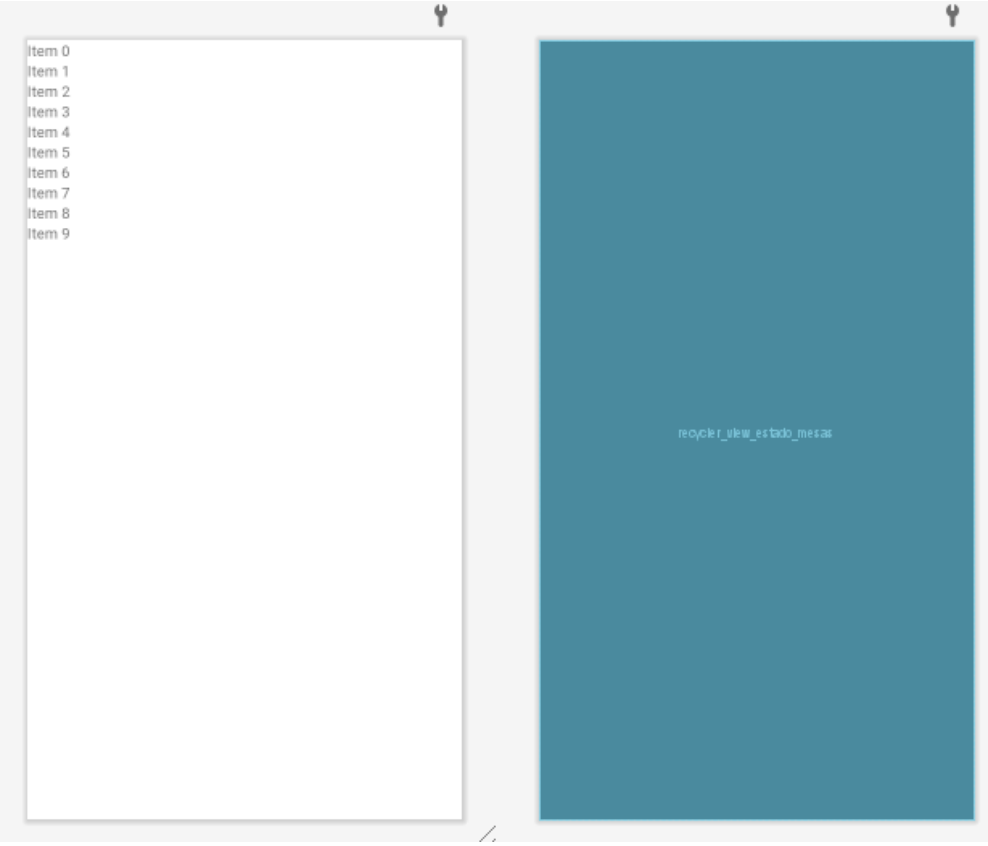
Pantalla métode de pagament:



Ilustració 31 - Pantalla de pagament

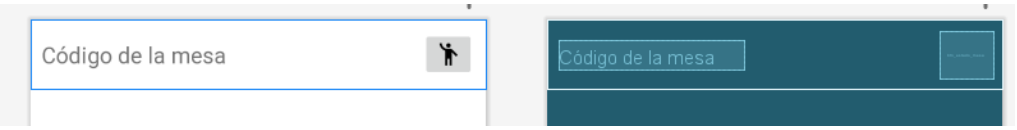
De nou és un disseny molt senzill que només conté un botó per a cada mètode de pagament. Segons el botó que premi el client s'obrirà un procés de pagament del botó que hagi premut.

Pantalla estat taules:



Ilustració 32 - Pantalla estat de taules

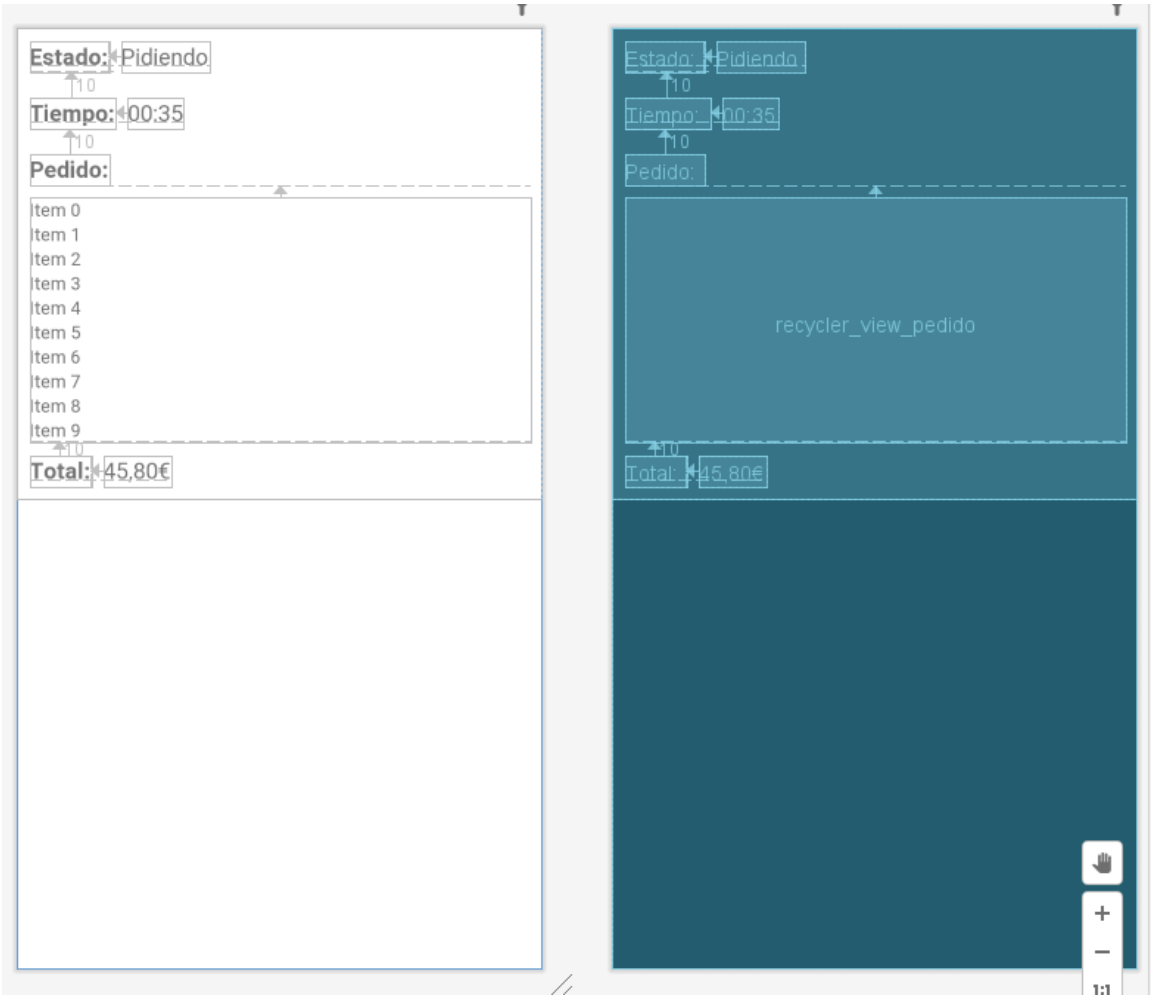
Aquest disseny consta de només un llistat on es mostraran les taules del restaurant. El disseny dels elements del llistat és el següent:



Ilustració 33 - Disseny llistat taules

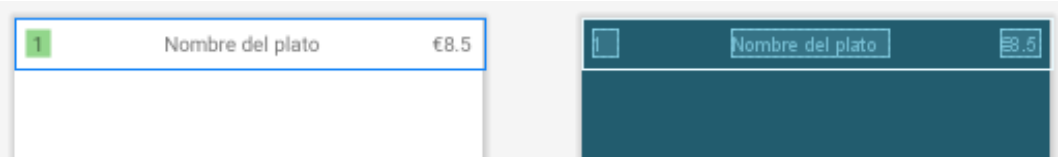
El disseny dels elements de la llista consta d'una etiqueta amb el codi de la taula i un botó amb una icona que canvia segons l'estat de la taula. A més, el color del fons de l'element també canvia segons l'estat perquè sigui més ràpid d'identificar quines taules necessiten assistència per part del treballador. Els estats que pot tindre una taula són: lliure, demanant, menjant o pagant.

Pantalla comanda taula treballador:



Ilustració 34 - Pantalla comanda treballador

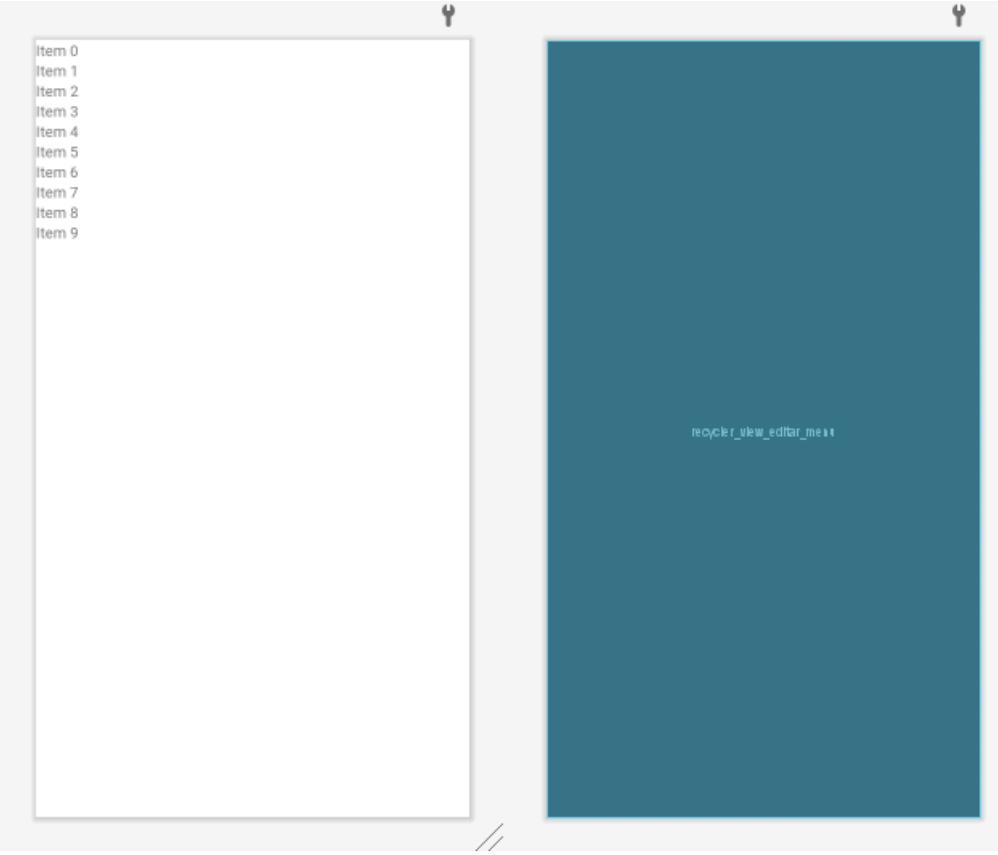
El disseny d'aquesta pantalla mostra l'estat de la taula, el temps des que la taula va ser ocupada, un llistat amb tots els plats demanats i per últim, el preu total de la comanda.



Ilustració 35 - Disseny llistat comanda treballador

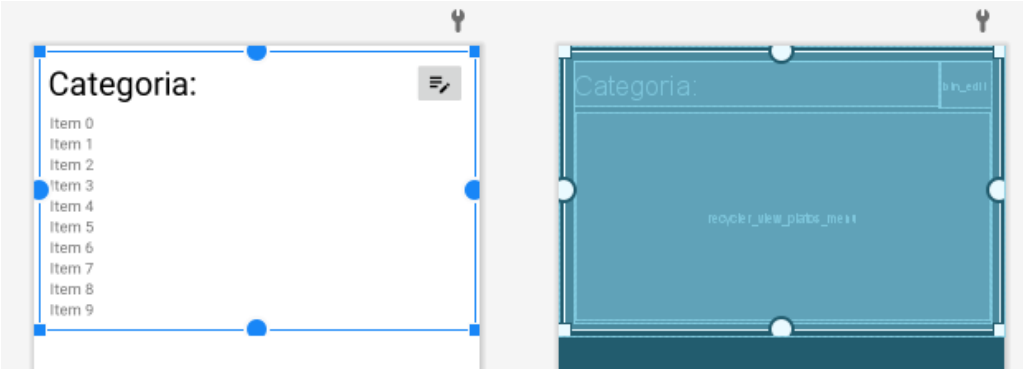
El disseny dels elements del llistat és igual que el del llistat de la pantalla de comanda, on es mostra la quantitat, el nom del plat i el preu.

Pantalla editar menú:



Ilustració 36 - Pantalla editar menú restaurant

La pantalla per a editar el menú consta d'una llista que mostra les categories que té el menú (per exemple: primers, segons, begudes i postres). A més, a la barra superior de la pantalla hi haurà una icona amb un + per a poder afegir noves categories i nous plats. El disseny de cada element de la llista s'explica a continuació:



Ilustració 37 - Disseny llistat categories editar menú

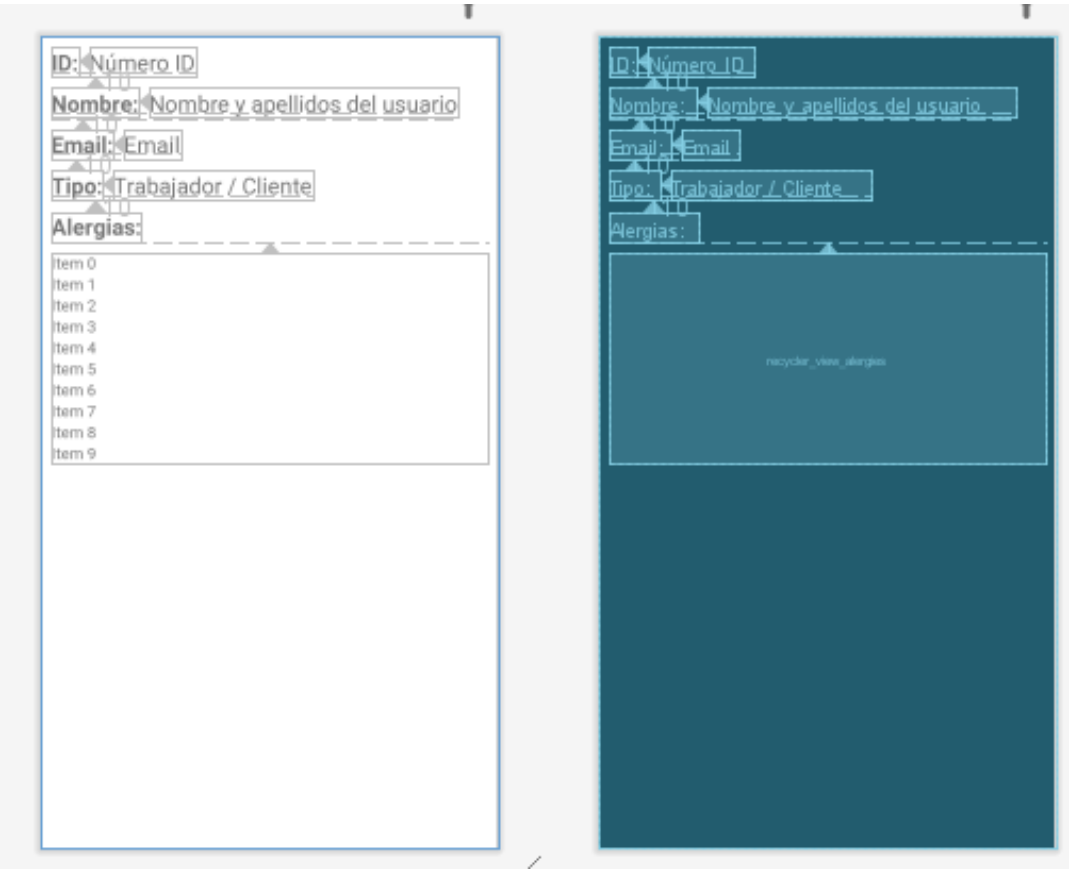
Cada element és una categoria del menú, per tant el disseny consta del títol de la categoria, a la dreta del títol hi ha un botó que permet editar la categoria, i a baix un llistat amb els noms dels plats que té aquesta categoria.

Pantalla crear plat:

Ilustració 38 - Pantalla crear plat

Aquest disseny consta d'un camp de text per a introduir el nom del plat, un altre per a la descripció del plat, un botó per afegir una imatge del plat, un camp de text per a les calories, un altre per al preu i per últim, un desplegable amb totes les categories que hi hagi al menú, que serà on s'afegirà el plat. Per finalitzar, a la part inferior es mostren dos botons un per cancel·lar i l'altre per guardar el plat.

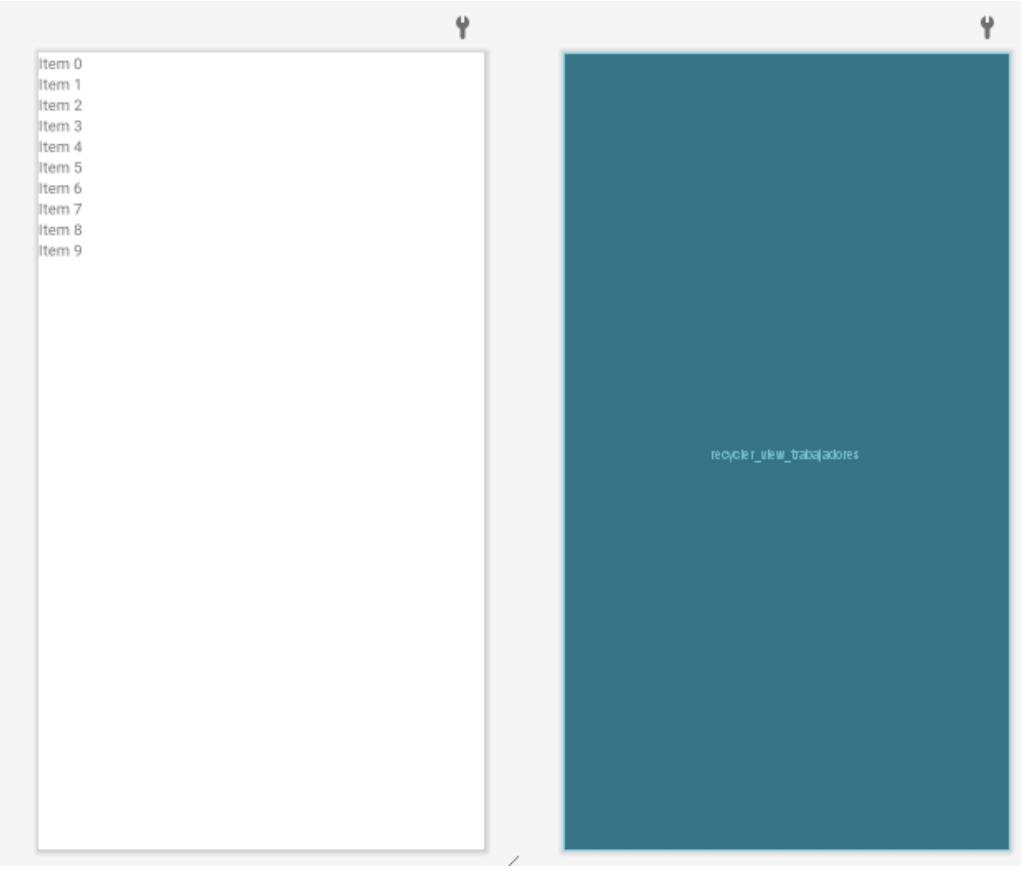
Pantalla perfil:



Ilustració 39 - Pantalla perfil

A la pantalla del perfil es mostra l'Id de l'usuari, el nom i cognoms, l'email, el tipus d'usuari i un llistat amb les al·lèrgies que l'usuari hagi introduït. El llistat amb les al·lèrgies pot ser modificat des d'aquesta mateixa pantalla. A la barra superior es mostrarà un botó per a editar les al·lèrgies.

Pantalla llistat treballadors restaurant:



Ilustració 40 - Pantalla llistat treballadors

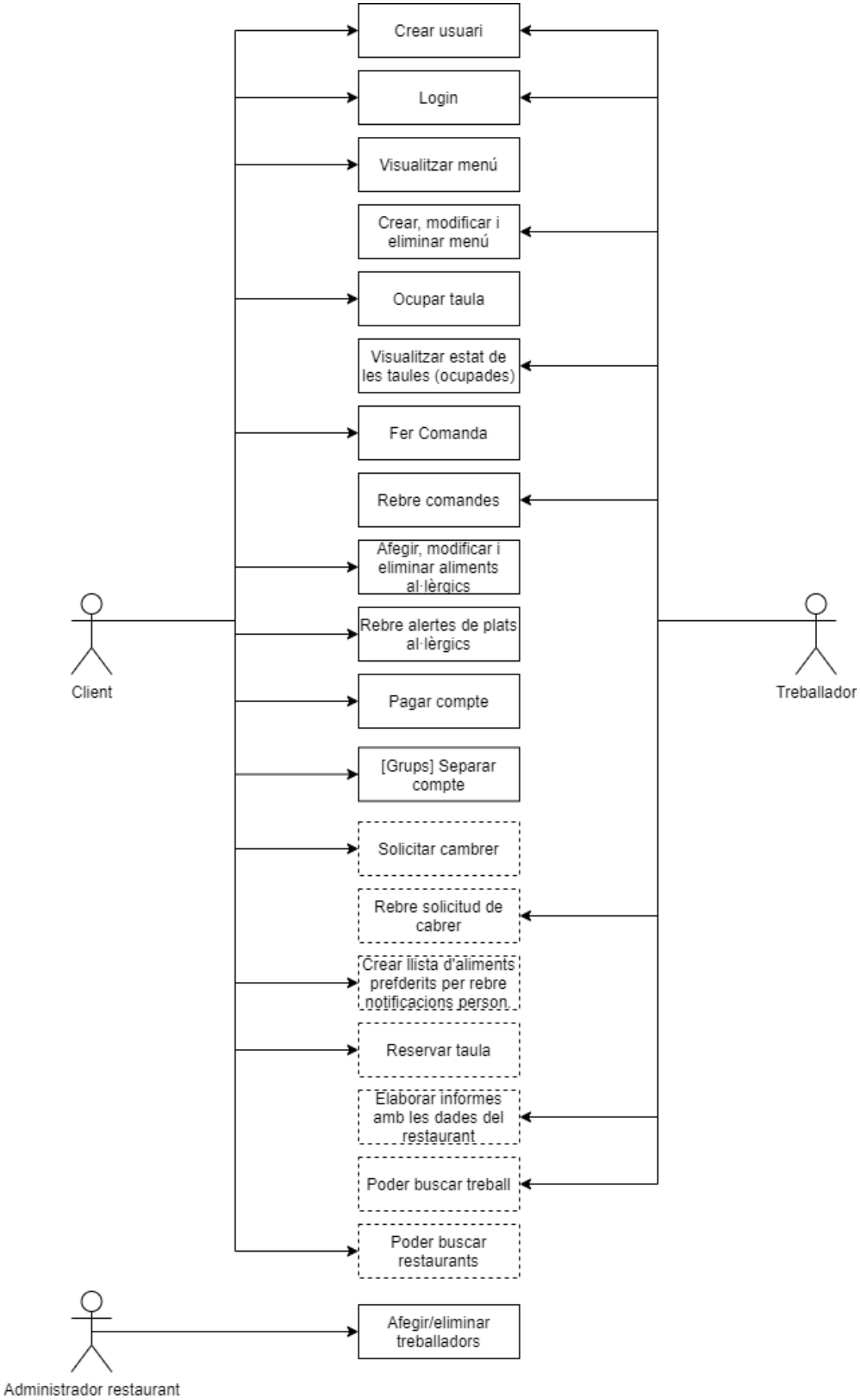
La pantalla amb el llistat de treballadors només conta d'un llistat on es mostren els noms i id de cada treballador. A la barra superior apareix un botó per a poder afegir un treballador al llistat. Aquest botó obrirà un dialog on s'ha d'afegir el codi del treballador. A més, per a eliminar un treballador es pot prémer a sobre del treballador i s'obrirà un dialog per a confirmar que es vol eliminar.

Com es pot observar tots els wireframes tenen un disseny molt senzill i en gran part dels casos basats en altres aplicacions del mercat que ja estan consolidades. Fent això es busca que sigui una aplicació molt fàcil d'usar, que sigui intuïtiva i que no necessiti un llarg procés d'aprenentatge.

Per a fer aquests dissenys s'han fet una sèrie d'iteracions per a anar editant contínuament els dissenys. I és possible que algun disseny pateixi algun canvi en un futur per a solucionar algun problema que no hagi estat contemplat.

2.4 Definició dels casos d'ús

A continuació s'adjunta el diagrama amb els casos d'ús i els diferents actors que interactuen amb l'aplicació.


Crear usuari
Login
Visualitzar menú
Crear, modificar i eliminar menú
Ocupar taula
Visualitzar estat de les taules (ocupades)
Fer Comanda
Rebre comandes
Afegir, modificar i eliminar aliments al·lèrgics
Rebre alertes de plats al·lèrgics
Pagar compte
[Grups] Separar compte
Solicitar cambrer
Rebre solicitud de cabrer
Crear llista d'aliments prefderits per rebre notificacions person.
Reservar taula
Elaborar informes amb les dades del restaurant
Poder buscar treball
Poder buscar restaurants
Afegir/eliminar treballadors
Client
Treballador
Administrador restaurant

Ilustració 41 - Diagrama casos d'ús

Seguidament es definiran tots els casos d'ús de l'aplicació junt amb les precondicions, postcondicions, actors...

Cas d'ús	**Crear usuari**			**CU01**
Actors	Client, treballador, administrador			
Tipus	Indispensable			
Precondicions	-			
Postcondicions	L'usuari ha sigut creat correctament i pot utilitzar l'aplicació			
Propòsit	Crear un usuari per a utilitzar l'aplicació			
Seqüència normal	**Pas**	**Acció**		
	1	En la pantalla principal prem el botó "Crear Usuari"		
	2	L'usuari escolleix el tipus d'usuari que vol crear, omple les dades demanades i prem el botó de Registrar.		
	3	El sistema crea el nou usuari.		
Excepcions	2	E.1	La contrasenya no es suficientment segura. L'usuari ha d'ingressar un altra contrasenya.	
	2	E.2	El correu ingressat ja és utilitzat per un altre usuari. L'usuari ha de fer ús d'un altre correu.	

Cas d'ús	**Login**			**CU02**
Actors	Client, treballador, administrador			
Tipus	Indispensable			
Precondicions	L'usuari ha de tenir un usuari creat			
Postcondicions	L'usuari ha sigut loguejat correctament i pot utilitzar l'aplicació			
Propòsit	Fer login per a utilitzar l'aplicació.			
Seqüència normal	**Pas**	**Acció**		
	1	En la pantalla principal ingressa el seu usuari i contrasenya, i prem del botó d'Entrar.		
	2	El sistema vàlida que sigui un usuari correcte.		
	3	Ingressa a l'usuari i mostra la pantalla segons quin tipus d'usuari sigui.		
Excepcions	2	E.1	La contrasenya o l'usuari no és correcte. El sistema demana tornar a introduir les dades correctament.	

Cas d'ús	**Ocupar taula**			**CU03**
Actors	Client			
Tipus	Indispensable			
Precondicions	L'usuari ha d'haver fet login			
Postcondicions	La taula passa a estat "ocupat" i l'usuari accedeix al menú del restaurant			
Propòsit	Ocupar la taula per a poder fer la comanda			
Seqüència normal	**Pas**	**Acció**		
	1	A la pantalla d'ocupar taula l'usuari ha d'introduir el codi de la taula		
	2	El sistema vàlida que la taula no estigui ocupada		
	3	El sistema mostra el menú del restaurant		
Excepcions	2	E.1	El codi de la taula no existeix o la taula està ocupada. L'usuari ha d'introduir un altre codi.	

Cas d'ús	**Visualitzar menú**	**CU04**
Actors	Client	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login i haver ocupat una taula	
Postcondicions	L'usuari ha pogut veure el menú del restaurant	
Propòsit	Veure el menú amb l'objectiu fe fer la comanda	

Seqüència normal	Pas	Acció
	1	L'usuari ocupa una taula d'un restaurant.
	2	El sistema mostra el menú del restaurant.

Cas d'ús	Crear menú	CU05
Actors	Treballador	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login com a treballador	
Postcondicions	El sistema guarda el menú del restaurant per a que els clients puguin visualitzar-lo	
Propòsit	Crear un menú del restaurant per a que els clients puguin visualitzar-lo	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari accedeix a la pantalla de creació de menú
	2	A la pantalla de creació el treballador prem el botó per a afegir una categoria del menú.
	3	L'usuari introdueix el nom de la categoria.
	4	L'usuari prem el botó per a afegir plats.
	5	L'usuari crea el plat i selecciona la categoria a la qual vol afegir-lo
	6	L'usuari prem el botó de guardar i el sistema guarda el menú

Cas d'ús	Modificar menú	CU06
Actors	Treballador	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login com a treballador	
Postcondicions	El sistema guarda els canvis del menú del restaurant perquè els clients puguin visualitzar-lo	
Propòsit	Modificar el menú del restaurant perquè els clients puguin visualitzar-lo	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari accedeix a la pantalla de modificació de menú
	2	A la pantalla de creació el treballador prem el botó per a afegir una categoria del menú.
	3	L'usuari introdueix el nom de la categoria.
	4	L'usuari prem el botó per a afegir plats. El codi de la taula no existeix o la taula esta ocupada. L'usuari ha d'introduir un altre codi.
	5	L'usuari crea el plat i selecciona la categoria a la qual vol afegir-lo
	6	L'usuari prem el botó de guardar i el sistema guarda el menú

Cas d'ús	Veure estat de les taules	CU07
Actors	Treballador	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login com a treballador	
Postcondicions	El sistema mostra l'estat de les taules del restaurant	
Propòsit	Veure l'estat de les taules del restaurant	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari accedeix a la pantalla d'estat de les taules
	2	Al llistat es mostra l'estat de cada taula, però l'usuari prem la taula que vol veure
	3	El sistema mostra la pantalla de la taula amb l'estat d'aquesta

Cas d'ús	Veure comanda	CU08
Actors	Treballador	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login com a treballador	
Postcondicions	El sistema mostra la comanda de la taula	

Propòsit	Veure la comanda de la taula per a poder servir	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari des del llistat de taules accedeix a la pantalla de la taula
	2	A la pantalla de la taula es mostra l'estat de la taula i la comanda realitzada

Cas d'ús	**Fer comanda**		**CU09**
Actors	Client		
Tipus	Indispensable		
Precondicions	L'usuari ha d'haver fet login com a client i haver ocupat una taula		
Postcondicions	El sistema guarda la comanda i l'envia als treballadors		
Propòsit	Enviar la comanda perquè els treballadors la puguin preparar		
Seqüència normal	**Pas**	**Acció**	
	1	L'usuari des de la pantalla del menú del restaurant	
	2	L'usuari va afegint els plats que vulgui a la comanda	
	3	L'usuari prem el botó de Realitzar la comanda	
	4	El sistema mostra la pantalla amb la comanda realitzada i el preu d'aquesta	
	5	L'usuari prem el botó de Confirmar comanda	
	6	El sistema guarda la comanda i l'envia perquè la preparin	
Excepcions	3	E.1	Si l'usuari ha afegit a la comanda un plat que conté un aliment al·lèrgic per a l'usuari. El sistema mostra una alerta sobre que un plat conté un aliment al·lèrgic, i pregunta si vol continuar igualment.

Cas d'ús	**Visualitzar perfil**	**CU10**
Actors	Client, treballador	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login	
Postcondicions	El sistema mostra el perfil de l'usuari	
Propòsit	Visualitzar el perfil de l'usuari	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari prem el botó de visualitzar perfil
	2	El sistema mostra el perfil de l'usuari amb el seu nom, tipus d'usuari, llista d'aliments al·lèrgics.

Cas d'ús	**Afegir aliment al·lèrgic**	**CU11**
Actors	Client	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login	
Postcondicions	El sistema guarda l'aliment en el llistat d'al·lèrgies	
Propòsit	Guardar l'aliment perquè surti un avís en cas de demanar-lo	
Seqüència normal	**Pas**	**Acció**
	1	L'usuari prem el botó d'editar perfil des de la pantalla del perfil
	2	L'usuari introdueix un aliment al qual es al·lèrgic i prem el botó de guardar
	3	El sistema guarda l'aliment al llistat, i genera alertes en cas de demanar l'aliment

Cas d'ús	**Eliminar aliment al·lèrgic**	**CU12**
Actors	Client	
Tipus	Indispensable	
Precondicions	L'usuari ha d'haver fet login i tindre algun aliment a la llista d'al·lèrgies	
Postcondicions	El sistema elimina l'aliment en el llistat d'al·lèrgies	
Propòsit	Eliminar l'aliment de la llista perquè no surtin més notificacions en cas de demanar-lo	

Seqüència normal	Pas	Acció
	1	L'usuari prem el botó d'editar perfil des de la pantalla del perfil
	2	L'usuari selecciona l'aliment al qual es al·lèrgic i prem el botó d'eliminar
	3	El sistema elimina l'aliment del llistat, i deixa de generar alertes en cas de demanar l'aliment

Cas d'ús	Pagar compte		CU13
Actors	Client		
Tipus	Indispensable		
Precondicions	L'usuari ha d'haver fet login i haver acabat de menjar		
Postcondicions	El client paga el compte, i la taula que estava ocupada deixa d'estar-ho		
Propòsit	Pagar el compte per a poder abandonar la taula		
Seqüència normal	Pas	Acció	
	1	L'usuari prem el botó de pagar de la pantalla d'espera	
	2	El sistema mostra els diferents mètodes de pagament	
	3	L'usuari segueix les instruccions segons el mètode de pagament seleccionat, i paga la comanda.	
	4	Els diners són descomptats del client, i els rep el restaurant. La taula que estava ocupada pel client passa a estar lliure.	

Cas d'ús	Sol·licitar cambrer		CU14
Actors	Client, treballador		
Tipus	Opcional		
Precondicions	L'usuari ha d'haver fet login i haver ocupat una taula		
Postcondicions	El client envia una notificació a un treballador per a ser atès		
Propòsit	Avisar a un treballador per a ser atès		
Seqüència normal	Pas	Acció	
	1	L'usuari prem el botó de sol·licitar assistència de la pantalla d'espera	
	2	El sistema envia una notificació a un treballador	

Cas d'ús	Crear llista amb aliments preferits		CU15
Actors	Client		
Tipus	Opcional		
Precondicions	L'usuari ha d'haver fet login		
Postcondicions	El client crea una llista amb els seus aliments preferits		
Propòsit	Crear llista amb els aliments preferits per a rebre recomanacions personalitzades		
Seqüència normal	Pas	Acció	
	1	L'usuari accedeix al seu perfil	
	2	Prem el botó d'editar	
	3	Introdueix el nom dels aliments preferits i prem el botó de guardar	
	4	El sistema guarda el llistat	

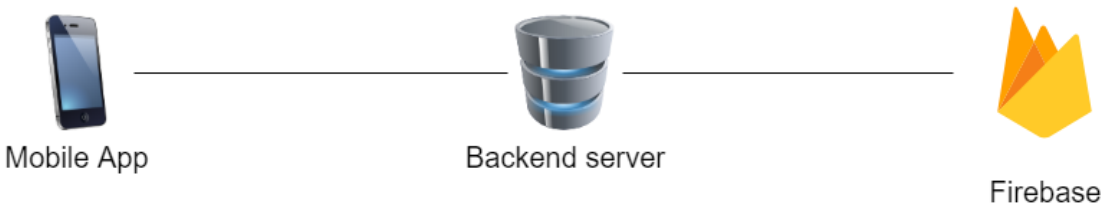
Cas d'ús	Reservar taula online		CU16
Actors	Client		
Tipus	Opcional		
Precondicions	L'usuari ha d'haver fet login		
Postcondicions	El client reserva taula per a un restaurant		
Propòsit	Reservar taula a un restaurant de manera online		
Seqüència normal	Pas	Acció	
	1	L'usuari entra a la pantalla de reserves	
	2	Introdueix el codi del restaurant	
	3	El sistema mostra les taules lliures segons les hores	
	4	L'usuari prem a sobre de la taula lliure que vulgui reservar i	

		selecciona l'hora
	5	L'usuari prem el botó reservar

Cas d'ús	**Afegir/eliminar treballador**		**CU17**
Actors	Administrador, treballador		
Tipus	Indispensable		
Precondicions	L'usuari ha d'haver fet login com a administrador i el treballador no ha de treballar a cap altre restaurant		
Postcondicions	El treballador passa a ser treballador del restaurant		
Propòsit	Afegir treballador al restaurant per a poder començar a treballar		
Seqüència normal	**Pas**	**Acció**	
	1	L'administrador accedeix al llistat de treballadors	
	2	Prem el botó d'afegir treballador	
	3	Introdueix l'ID del treballador i prem el botó de guardar	
	4	El sistema registra el treballador	
Excepcions	3	E.1	Si el treballador afegit no està disponible o no existeix, el sistema mostrarà un avís indicant que no és un treballador disponible

2.5 Arquitectura

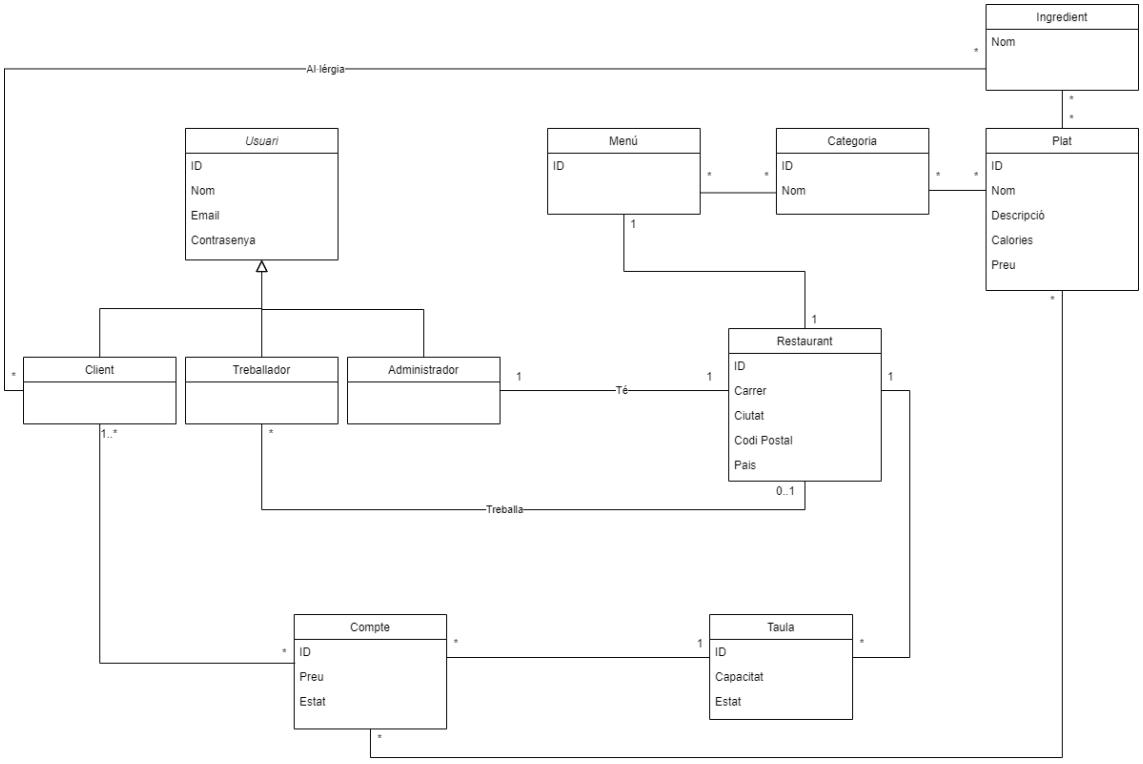
L'arquitectura de Easy order té tres parts. L'aplicació Android (per a iOS queda fora de l'abast del projecte, com a proposta de millora), el servidor backend (per a facilitar fer l'aplicació multiplataforma en un futur) i la base de dades de Firebase.



Ilustració 42 - Diagrama arquitectura Easy Order

El servidor backend és on es realitzaran la majoria de processos i transformacions de les dades, que seran consumides per l'aplicació mòbil mitjançant peticions REST. Firebase conté la base de dades on s'emmagatzemen les dades i també gestiona els usuaris de l'aplicació.

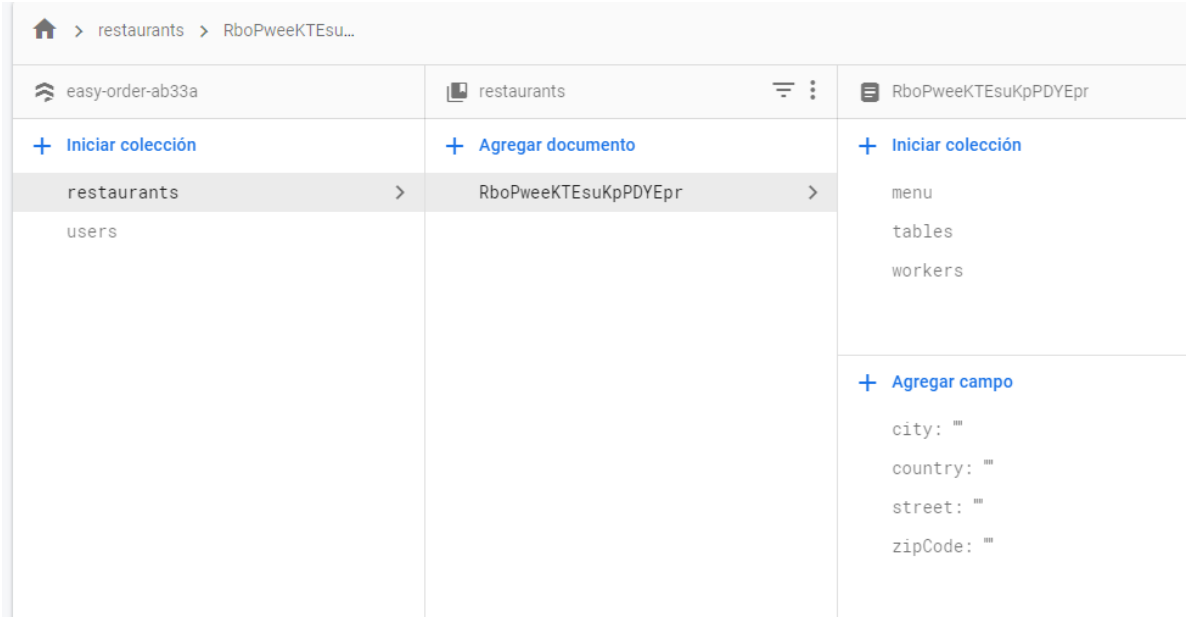
A partir dels casos d'usos s'ha fet un diagrama de classes amb les entitats que s'utilitzaran per a gestionar els diferents processos que realitza l'aplicació. S'adjunta a continuació.



Ilustració 43 - Diagrama de classes

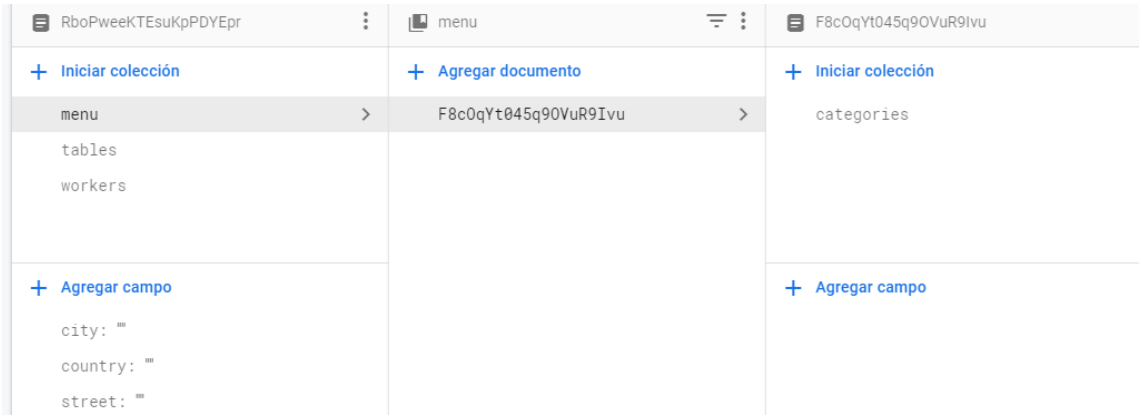
Al diagrama es pot observar com hi ha tres tipus d'usuaris diferents. Els clients són els que poden ocupar una taula i per tant crear un compte que tindrà diversos plats. A més els clients poden tenir un llistat amb els ingredients als quals són al·lèrgics. L'administrador té un restaurant. I el treballador treballa a un restaurant. Pot haver-hi més d'un treballador per no més d'un administrador. Un restaurant pot tindre més d'una taula. A més un restaurant té un menú, que conté diverses categories i cada categoria conté diversos plats.

La base de dades utilitzada serà NoSQL, segueix una estructura de clau-valor amb col·leccions. Per tant es mostrarà com s'ha decidit presentar amb captures reals de la base de dades de Firebase.



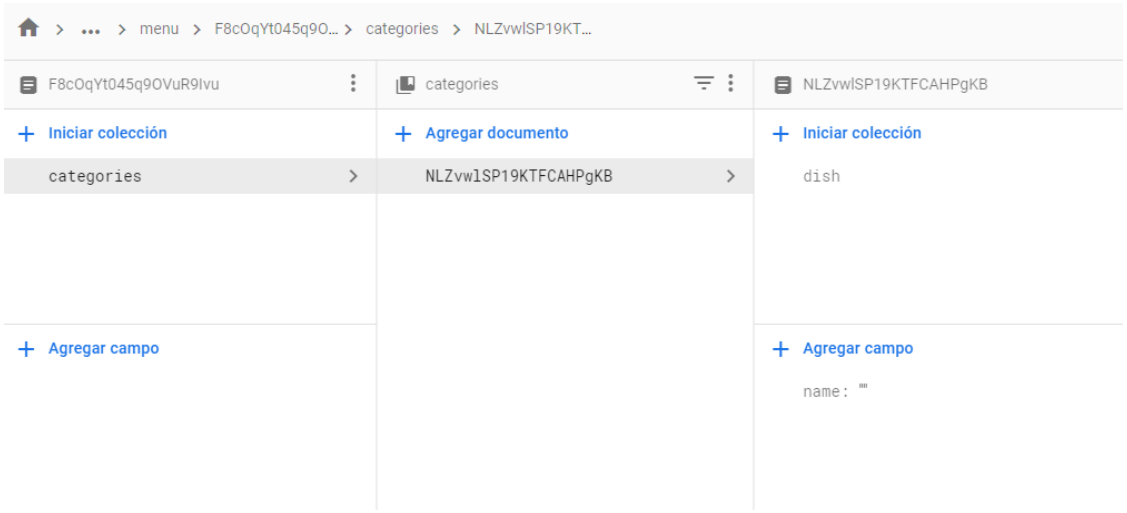
Ilustració 44 - Base de dades Easy Order 1

Com es pot observar té dues col·leccions principals que són els restaurants i els usuaris. S'iniciarà analitzant la col·lecció dels restaurants. Dins de la col·lecció de restaurants hi ha un document per cada restaurant registrat a l'aplicació (amb el seu ID). A més, cada restaurant té la ciutat, país, carrer i codi postal com a atributs. Seguidament dins de cada restaurant hi ha tres col·leccions el menú, les taules i els treballadors.



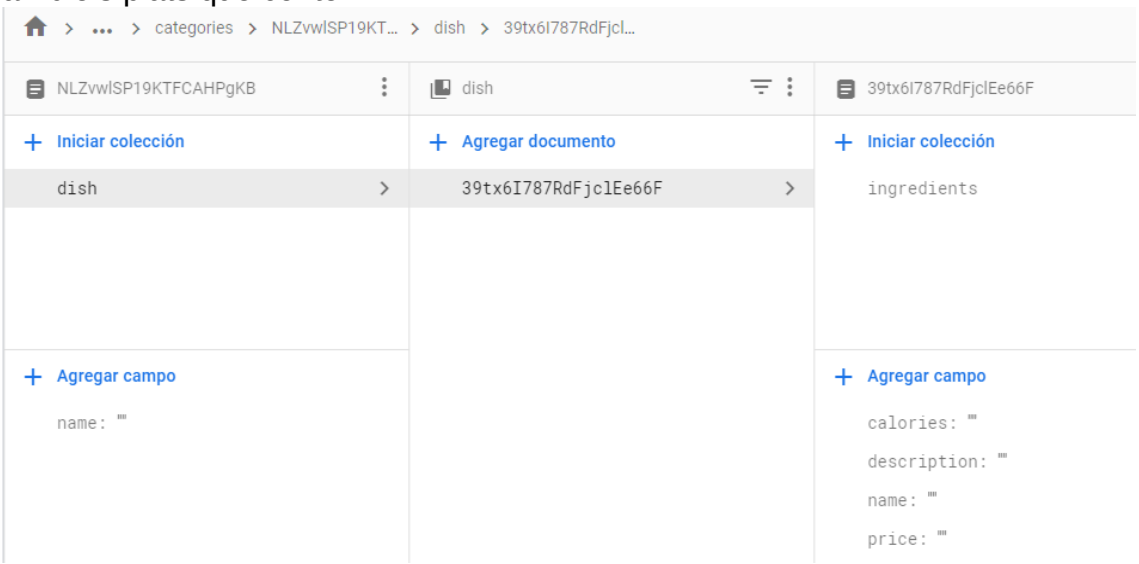
Ilustració 45 - Base de dades Easy Order 2

Dins de menú hi ha l'ID del menú, i una col·lecció amb les categories que té el menú.



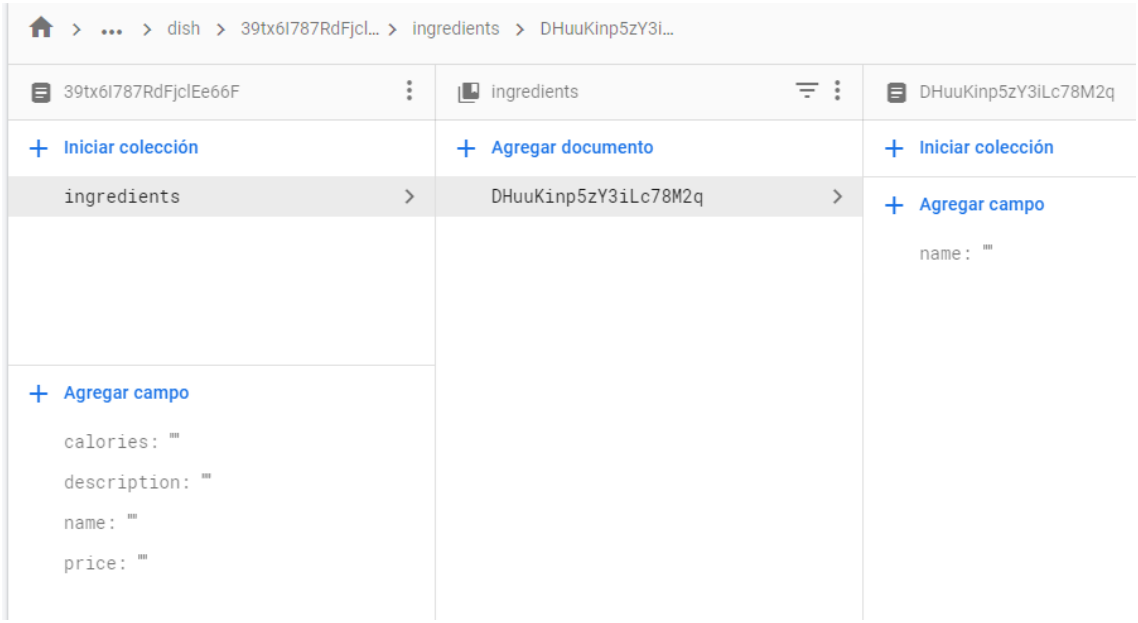
Ilustració 46 - Base de dades Easy Order 3

Cada categoria té un ID i un nom. I dins de cada categoria hi ha una col·lecció amb els plats que conté.



Ilustració 47 - Base de dades Easy Order 4

Cada plat té un id, el nombre de calories, una descripció, un nom i un preu. A més, d'una col·lecció amb els ingredients del plat.

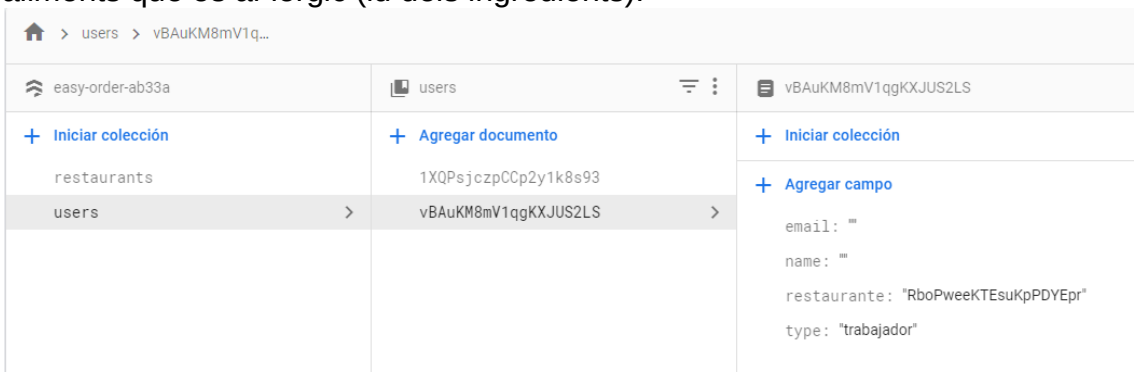


Ilustració 48 - Base de dades Easy Order 5

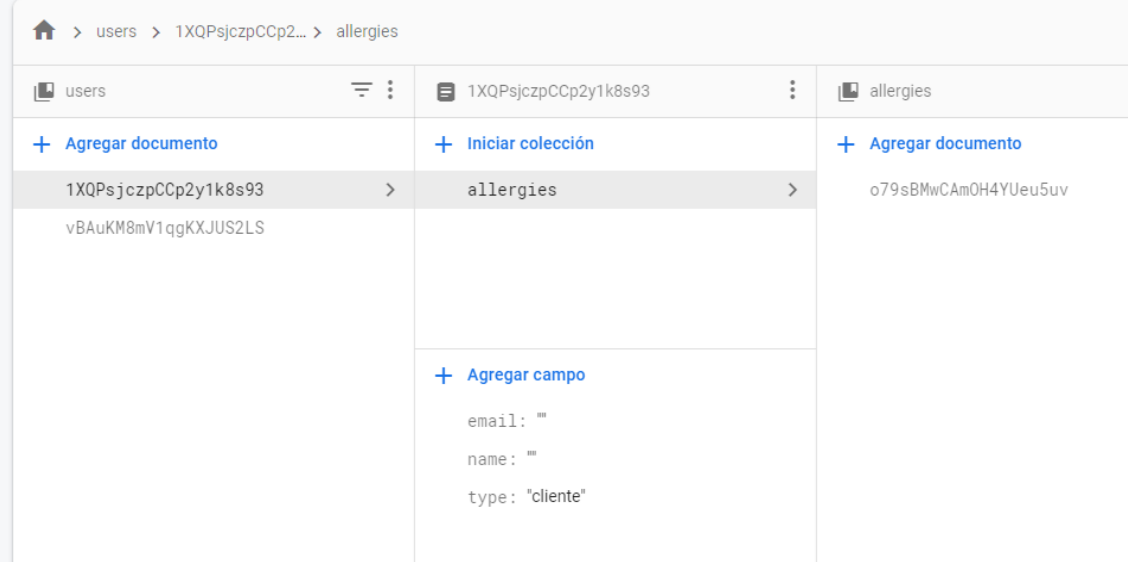
Per últim, cada ingredient del plat té un identificador únic i un nom. Aquests ingredients serviran per activar les notificacions d'al·lèrgies dels usuaris.

D'altra banda hi ha l'altra col·lecció principal que és la dels usuaris.

Cada usuari té un email, un nom i el tipus d'usuari. En el cas en el qual l'usuari sigui un treballador tindrà també un atribut amb el codi identificador del restaurant. En el cas en el qual l'usuari sigui un client, tindrà un llistat amb els aliments que és al·lèrgic (id dels ingredients).



Ilustració 49 - Base de dades Easy Order 6



Ilustració 50 - Base de dades Easy Order 7

El servidor backend és on es faran la majoria de processos i s'enviaran les dades en format JSON a l'aplicació Android (Frontend). Des de l'aplicació Android s'hauran de fer peticions Rest a l'API del servidor.

El servidor es farà amb Java i Spring. El backend té tres tipus de components bàsics. Controllers que és on es defineixen els endpoints. Services que són cridats pels controllers, i s'encarreguen de fer els processos necessaris i retornen les dades solicitades. Dao que són cridats pels services i són els que interactuen amb la base de dades. A més, de les classes del diagrama de classes mostrat anteriorment que forment part del model.

Per últim, per la part del frontend (Android App). Es seguirà el patró de ModelView – View- Model. Les vistes mostren les interfícies i informen a les següents capes sobre les accions que realitzen els usuaris. ViewModel exposa la informació a la vista perquè pugui presentar-la. Per finalitzar, el model rep la informació del datasource i es passa al viewModel.

2.6 Implementació

A continuació es mostraran les principals decisions a l'hora d'implementar l'aplicació. Inicialment, es concretaran algunes variacions que hi ha hagut respecte a la planificació inicial, degut a diversos motius de salut, entre altres, no s'ha pogut seguir la planificació al complet i s'ha hagut de prioritzar algunes històries d'usuari i altres s'han hagut de deixar com a opcions de millora.
Les principals històries d'usuari que estaven planificades com a indispensables i no s'han pogut implementar són les que permet als usuaris pagar el compte fet ús de l'aplicació i la que permet als usuaris indicar els aliments als quals és al·lèrgic per a evitar que els pugui demanar.
A més, hi ha altres que eren opcionals com per exemple afegir imatges dels plats a la carta, que tampoc s'ha pogut implementar i es deixa com a proposta de millora.

A causa dels imprevistos no s'ha pogut realitzar una aplicació amb tantes funcionalitats com es desitjava, però s'han pogut solucionar els imprevistos fins a obtenir un producte usable.

2.6.1 Sistema

Easy Order consta de tres parts principals. El front-end, el back-end i Firebase que s'usa com a base de dades i sistema d'usuaris. El motiu per el qual s'ha escollit fer ús d'un servidor back-end entremig del front i Firebase és per a poder fer l'aplicació multiplataforma més senzilla en un futur. Actualment, el Front-end realitzat és Android (Kotlin), però en un futur es vol fer l'aplicació disponible per a iOS, i d'aquesta manera és més senzill. Ja que s'ha intentat que el front-end realitzi el mínim de càlculs, i només es dediqui a presentar la informació recollida des del servidor back-end, que és on es fan les tasques i qui es connecta amb el back-end.
D'aquesta manera, en un futur quan es vulgui fer l'aplicació per a altres plataformes, o la pàgina web, només caldrà comunicar-se amb el servidor back-end mitjançant peticions REST i presentar la informació.

2.6.2 Firebase

La base de dades de Firebase ja està definida a l'apartat 2.5 d'arquitectura. Però de Firebase també es fa ús del seu sistema d'usuaris (Authentication). Actualment, només s'ha habilitat l'opció de registrar amb correu i contrasenya, però un dels motius pels quals s'ha decidit utilitzar aquest sistema, és perquèè permet habilitar opcions per a registrar-se fent ús dels comptes de Google, Twitter, Facebook... I són opcions molt interessants per a futures millores.

A més, en fer login amb un usuari Firebase genera un Token que té una durada de 1h. Aquest token és el que s'utilitzarà per a fer les peticions al back-end. Cada petició REST des del Front cap al Back ha de portar aquest Token, i el back el primer que farà és validar amb Firebase que aquest Token sigui correcte. En cas afirmatiu, executarà el codi necessari, però en cas negatiu es

tornarà un missatge d'error al Front indicant que el Token no es correcte. Això, proveeix d'un sistema de seguretat per a evitar que qualsevol persona pugui modificar dades de l'aplicació.

A més, des de Firestore (base de dades de Firebase) es permet definir unes regles per a millorar la seguretat. S'ha implementat la següent regla que permet llegit i escriure informació només als usuaris que estiguin autentificats.

```
1  rules_version = '2';
2  service cloud.firestore {
3    match /databases/{database}/documents {
4      match /{document=**} {
5        allow read, write: if request.auth != null;
6      }
7    }
8  }
```

Ilustració 51 - Reglas base de dades Firesotre

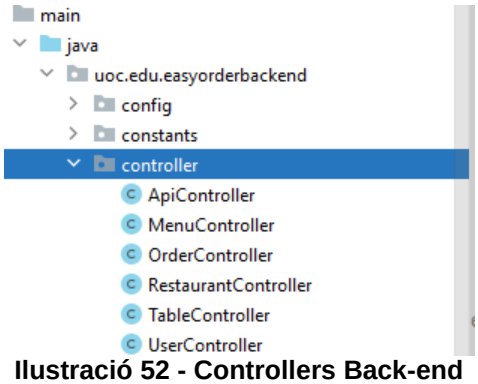
Tot i tenir aquesta regla, hi ha altres regles que proveeixen d'una major seguretat, i de cara al futur s'estudiarien la millor manera de garantir la màxima seguretat possible.

2.6.3 Servidor Back-end

Per a la implementació del servidor Back-end s'ha fet servir l'IDE IntelliJ amb el llenguatge Java. El codi s'ha anat pujant a un repositori Git per a portar el control de versions del codi.

S'ha fet ús de Spring Framework, ja que proporciona moltes facilitats a l'hora d'implementar un servidor back-end. Permet manejar de manera senzilla la injecció de dependències, proporciona seguretat (Spring-Security), permet realitzar una API per a consumir els recursos seguint una arquitectura MVC de manera molt senzilla.

L'arquitectura del servidor segueix el patró de MVC, fent servir els recursos que proporciona Spring. Les crides al servidor entren per les classes Controller.



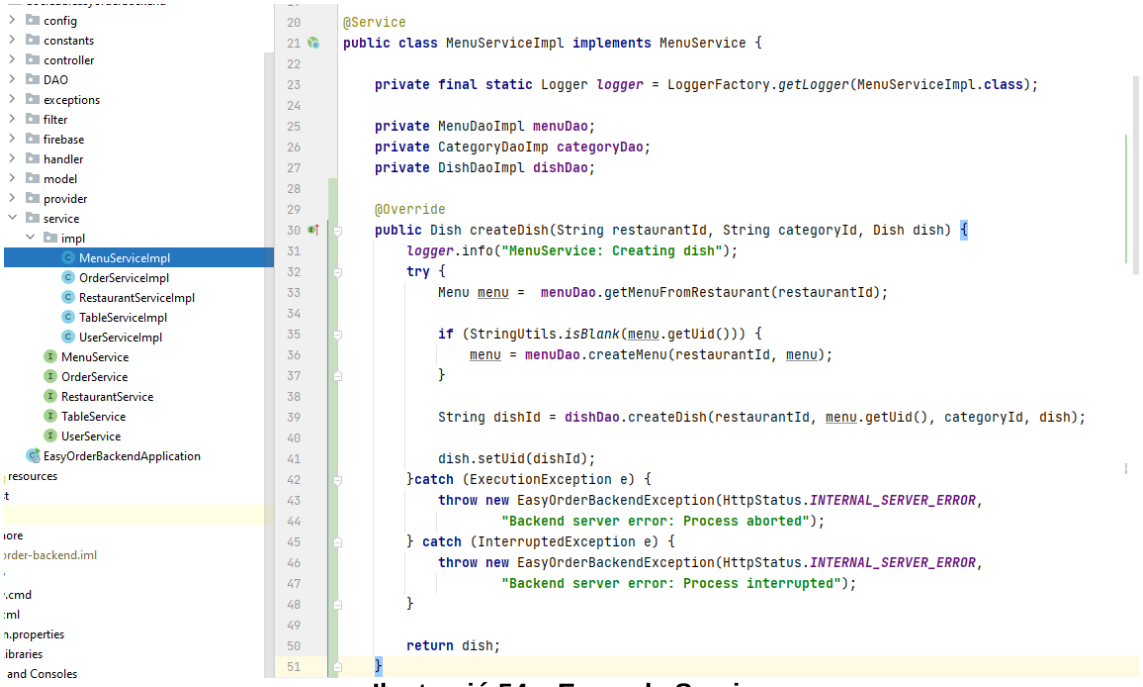
Ilustració 52 - Controllers Back-end

Les classes Controller només reben la petició (get, post, put, delete...), i fan la criada al Service. El Service retorna la informació pertinent i el Controller retorna una ResponseEntity amb la informació desitjada.

```
@RestController
@RequestMapping(UrlEasyOrderConstants.menuUrl)
public class MenuController {
    private final static Logger logger = LoggerFactory.getLogger(MenuController.class);

    private MenuService menuService;

    @GetMapping(UrlEasyOrderConstants.getFromRestaurant)
    public ResponseEntity<Menu> getMenuFromRestaurant(@PathVariable String restaurantId) {
        logger.info("MenuController: Get Menu from restraunt");
        ResponseEntity<Menu> response;
        if (StringUtils.isNotBlank(restaurantId)) {
            Menu menu = menuService.getMenuFromRestaurant(restaurantId);
            response = new ResponseEntity<>(menu, HttpStatus.OK);
        } else {
            throw new EasyOrderBackendException(HttpStatus.BAD_REQUEST, "Invalid ID");
        }

        logger.info("MenuController: Giving response");
        return response;
    }
```

Ilustració 53 - Exemple Controller

Les classes Service fan els processos necessaris i les crides als Daos necessaris per a obtenir la informació demanada.

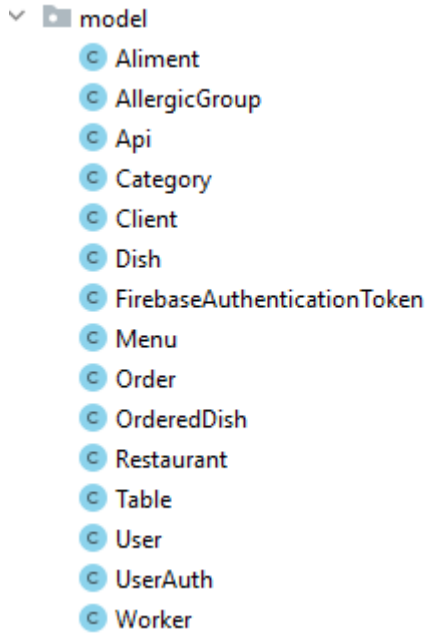


Ilustració 54 – Exemple Service

Des dels Services es fan les crides les classes DAO, que són les úniques que tenen interacció amb la base de dades (Firebase). D'aquesta manera es redueix al mínim l'acoblament. S'ha creat un DAO per a cada Entitat.

Ilustració 55 - Exemple Dao

A l'aplicació es fa ús de diverses classes que representen una entitat en el model de negoci.

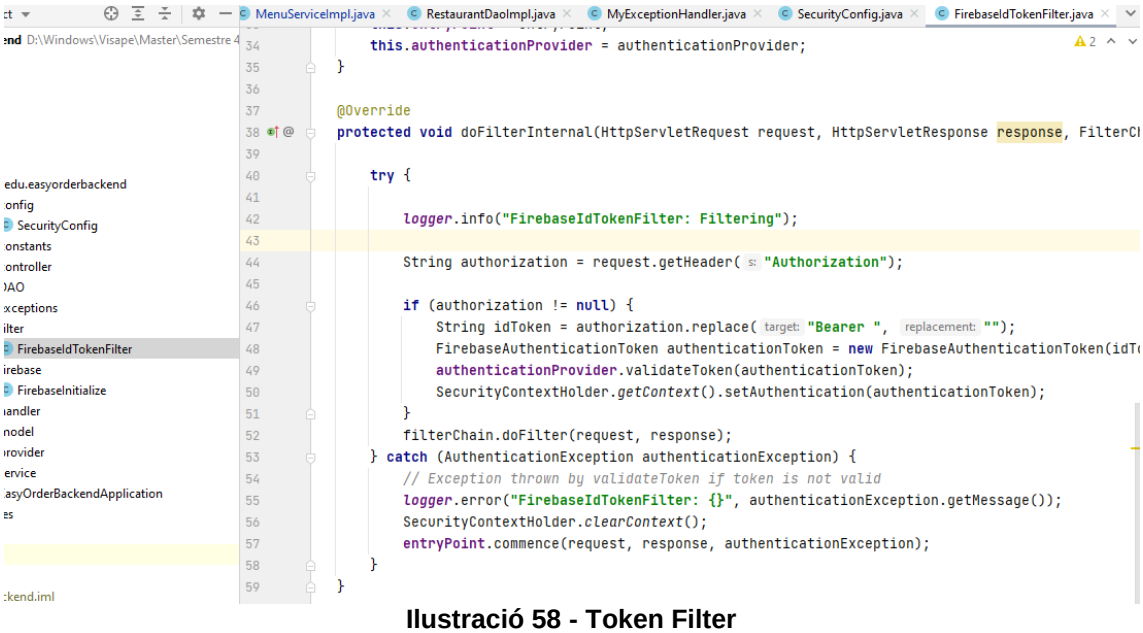


Ilustració 56 - Classes de Model

Per al control de les excepcions s'ha creat un Handler que tracta les excepcions i retorna un ResponseEntity amb un EasyOrderException amb un missatge que indica quin ha sigut el problema.

Ilustració 57 - Exception handler

Com s'ha comentat anteriorment, des del back-end les peticions el primer que fan és passar un filtre per a verificar que el Token passat a la petició és correcte, fent ús de la classe FirebaseIdTokenFilter.



Ilustració 58 - Token Filter

Al servidor hi ha algunes classes més que permeten configurar la seguretat, guardar constants, crear excepcions... Però les principals són les que ja s'han mostrat.
En el servidor Back-end s'ha prioritzat programar un servidor que sigui fàcil de mantenir i de modificar.

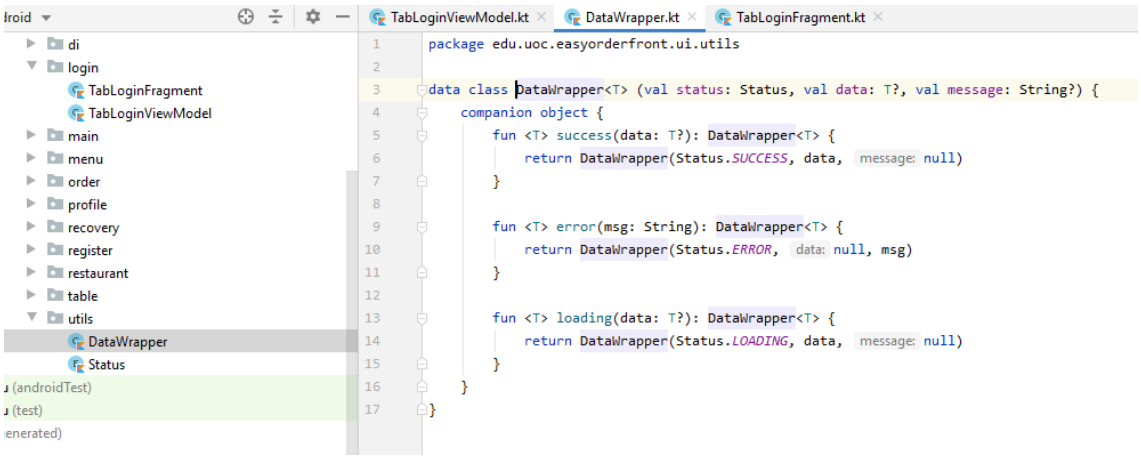
Per accedir al servidor s'ha fet servir Heroku que és una plataforma de Cloud que permet publicar una aplicació. S'ha escollit Heroku perquè té una versió gratuïta i en cas de voler més prestacions és fàcil d'escalar amb una versió de pagament. A més, s'integra molt bé amb Github i permet fer el deploy del Back de manera senzilla.
La URL de l'aplicació és https://easy-order-backend.herokuapp.com/

2.6.4 Front-end

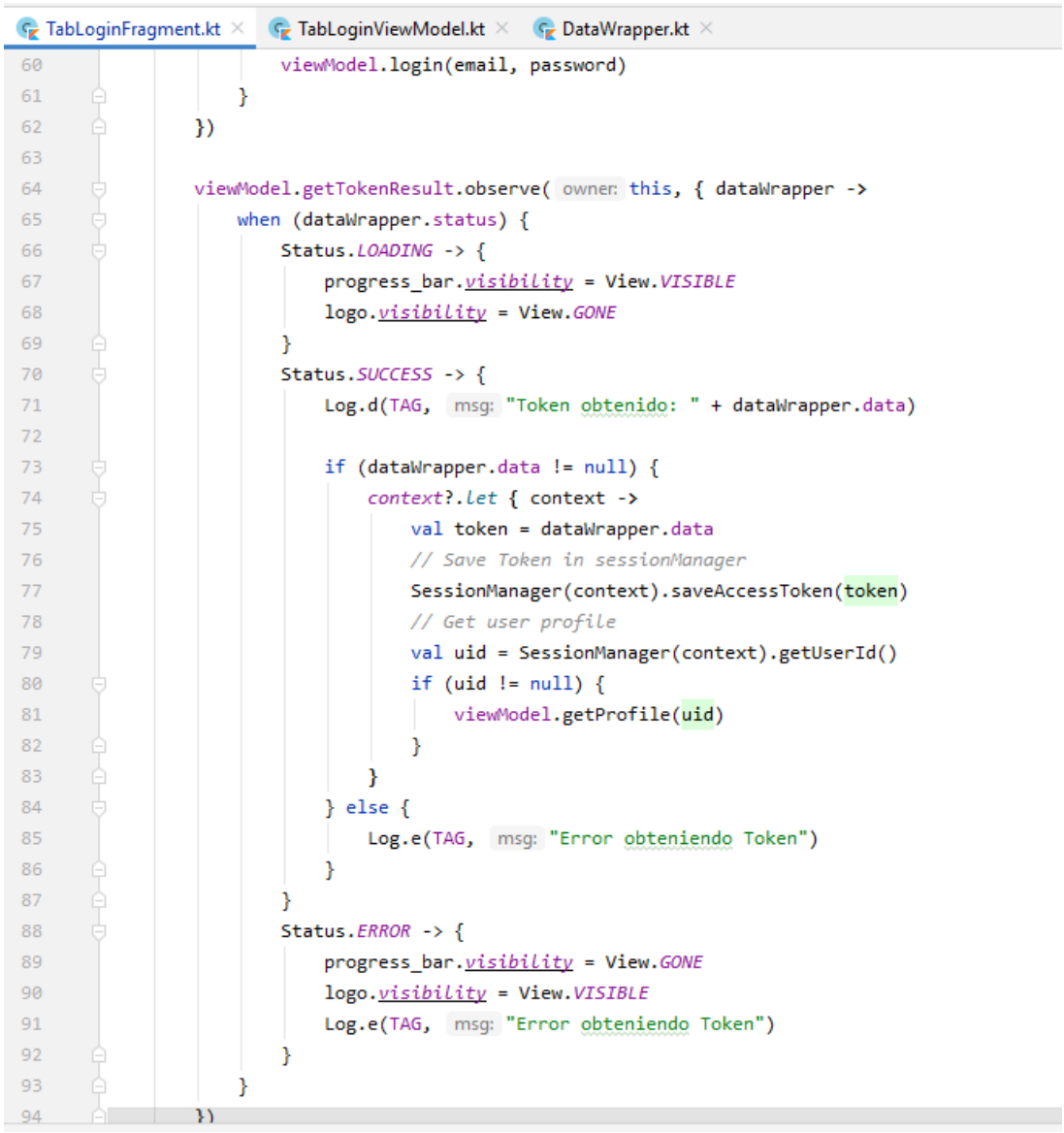
El front-end del sistema es basa en una aplicació Android, fent ús del llenguatge Kotlin.
El patró arquitectònic utilitzat per la implementació del Front és MVVM. On hi ha unes classes que són les vistes que presenten la informació, el model que representa les entitats del negoci i dades, i per últim les classes de ViewModel que són les que actuen com a intermediari entre les Vistes i el Model i conté la lògica de presentació de les dades.

A les classes de Vistes i ViewModel es fa ús d'objectes MutableLiveData que permet que la vista observi l'objecte i actualitzi la pantalla en cas que pateixi una modificació. A més, s'ha creat una classe anomenada DataWrapper, que permet informar a la vista en quin estat està la petició que ha realitzat al ViewModel (Success, Error, Loading), i així modificar la vista.

A continuació es mostra el codi fent ús de la classe DataWrapper amb el MutableLiveData.



Ilustració 59 - DataWrapper class

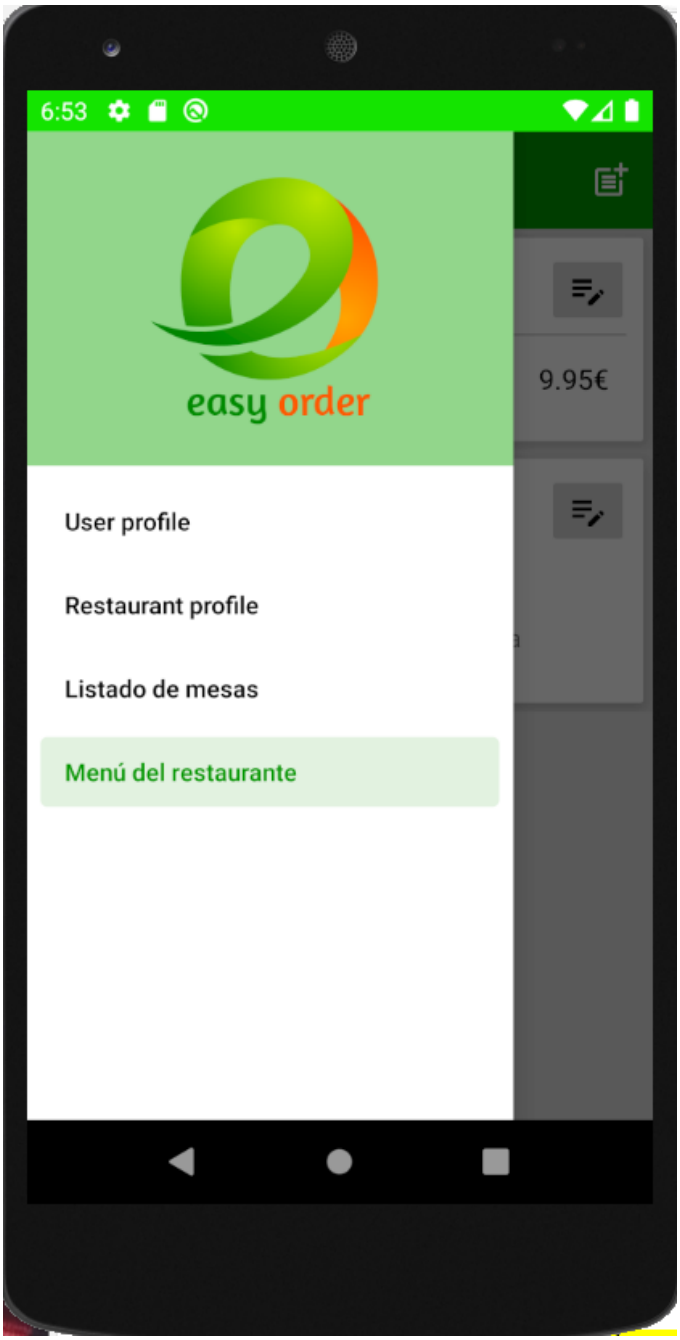


Ilustració 60 - DataWrapper example (1)

```
class TabLoginViewModel (private val repository: AuthenticationRepository,
                         private val profileRepository: ProfileRepository) : ViewModel() {
    val isLogged = MutableLiveData<DataWrapper<FirebaseUser?>>()
    val getTokenResult = MutableLiveData<DataWrapper<String?>>()

    var userProfile = MutableLiveData<DataWrapper<User?>>()

    private val TAG = "TabLoginViewModel"

    fun getTokenId() {
        viewModelScope.launch { this: CoroutineScope
            try {
                getTokenResult.postValue(DataWrapper.loading( data: null))

                repository.getIdToken().let { tokenResponse ->
                    Log.d(TAG,  msg: "Register: $tokenResponse")
                    getTokenResult.postValue(DataWrapper.success(tokenResponse))
                }
            } catch (e : Exception) {
                Log.e(TAG, e.toString())
                getTokenResult.postValue(DataWrapper.error(UIMessages.ERROR_GENERICO))
            }
        }
    }
```

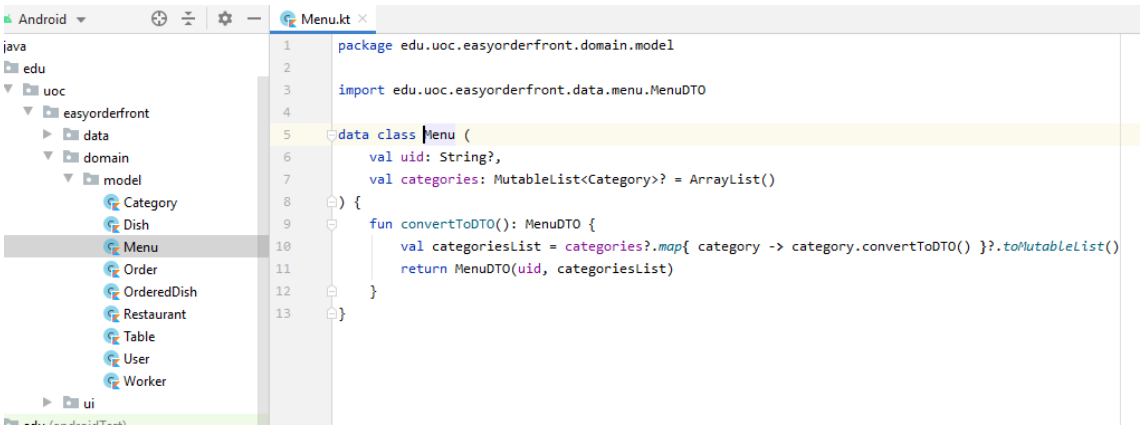
Ilustració 61 - DataWrapper example (2)

A l'aplicació es fa ús de fragments, ja que s'ha decidit fer un menú "d'hambuguesa", ja que permet navegar per l'aplicació de manera molt senzilla i intuïtiva.



Ilustració 62 - Menú de l'aplicació

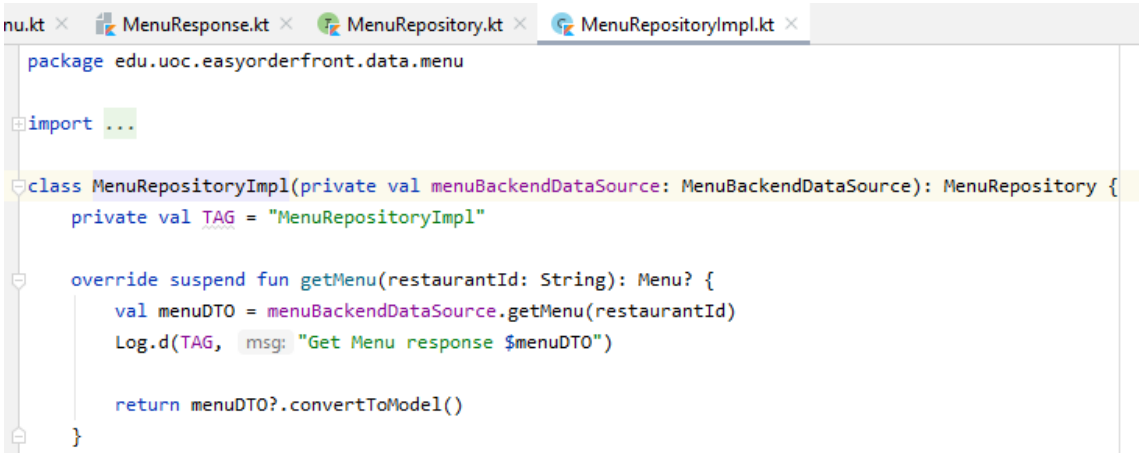
Igual que al Back-end, al Front s'ha creat les classes amb les entitats de l'aplicació. Aquestes entitats s'utilitzen al Front, d'altra banda per a la comunicació amb el Back s'han creat una classe per a cada classe de Model però que és un DTO, que permet serialitzar-se en la petició.



Ilustració 63 - Front entity example

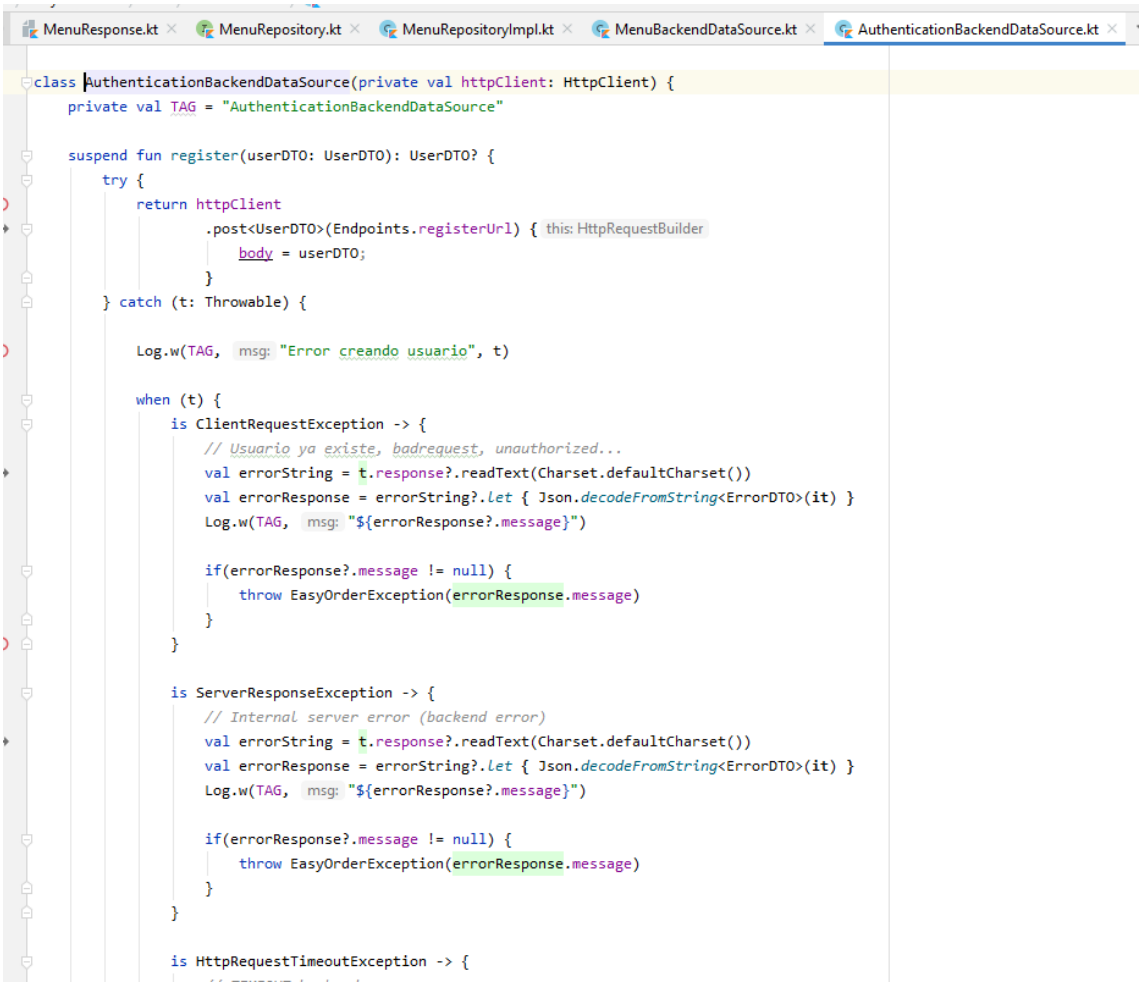
Ilustració 64 - Front DTO example

Des de les classes de ViewModel es fan les crides a les classes Repository, fent ús de les entitats, i des del Repository es fa la tranformació a DTO i es fa la crida al DataSource.



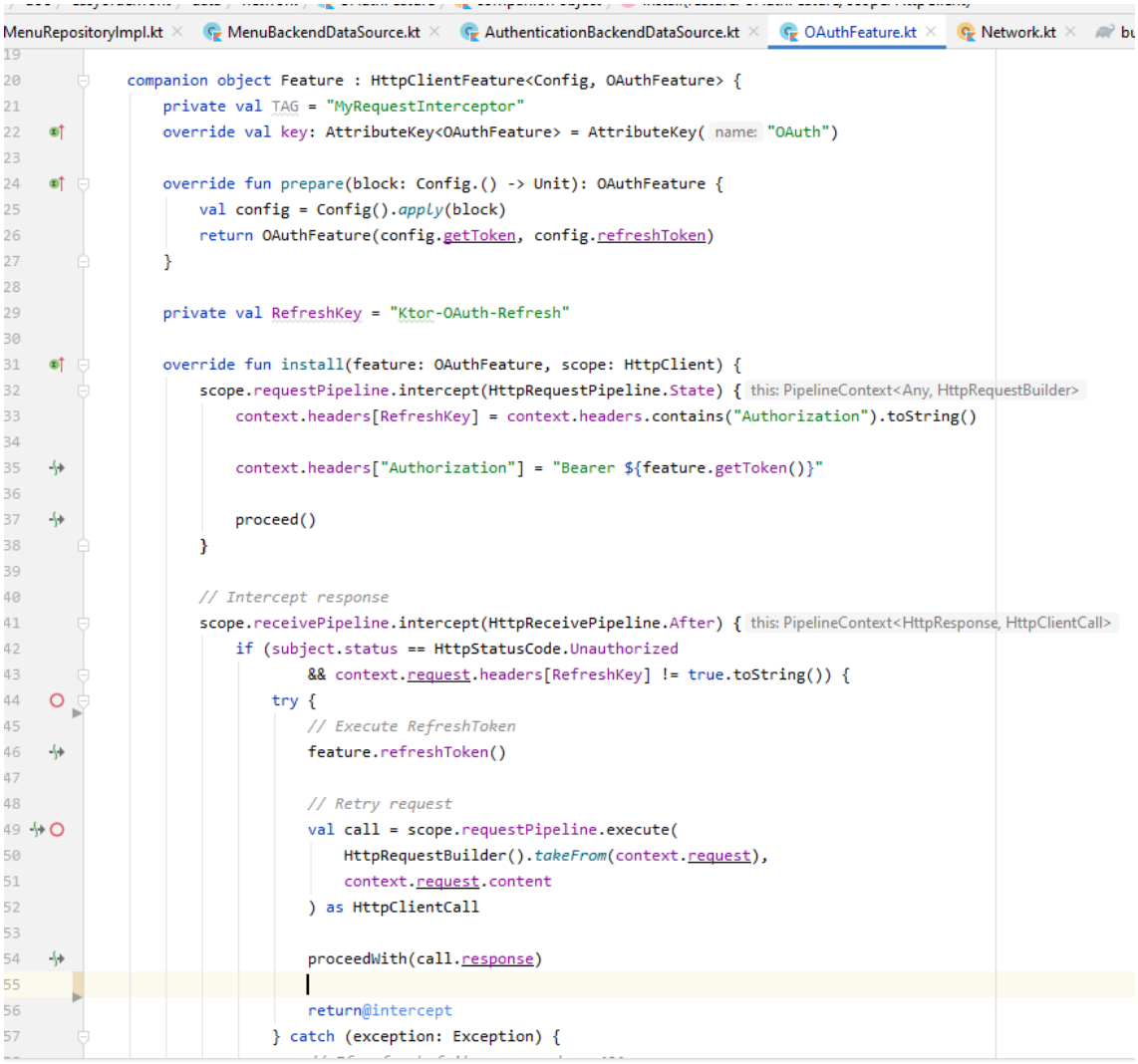
Ilustració 65 - Repository example

Per últim, des del DataSource es fa la petició al backend amb el DTO.



Ilustració 66 - DataSource example

Per a fer les peticions s'ha fet ús de Ktor. I s'ha creat un OAuthFeature que posa el token a les peticions, i en cas que el back retorni que el Token està caducat, l'OAuthFeature fa la petició a Firebase per a renovar el Token.



Ilustració 67 - OAuthFeature

Una de les llibreries utilitzada és QRGenerator, que permet generar el codi QR de les taules del restaurant.

```
val tableID = viewModel.table.value?.data?.tableRef
val qrgEncoder = QRGEncoder(tableID,  bundle: null, QRGContents.Type.TEXT, dimen)

val bitmap = qrgEncoder.encodeAsBitmap()

val getQrCodeActivity = GetQrCodeActivity(bitmap)
```

Ilustració 68 - QRGenerator example

D'altra banda, també s'ha fet ús de la llibreria Koin per a la gestió d'injecció de dependències.

UIModule.kt DataModule.kt

```kotlin
package edu.uoc.easyorderfront.data.di

import ...

val dataModule = module { this: Module
    factory { Network.createHttpClient(androidContext()) }

    single { SessionManager(get()) }

    // DATASOURCE
    single { AuthenticationBackendDataSource(get()) }
    single { RestaurantBackendDataSource(get()) }
    single { ProfileBackendDataSource(get())}
    single { TableBackendDataSource(get()) }
    single { MenuBackendDataSource(get()) }
    single { OrderBackendDataSource(get()) }

    single { FirebaseDataSource() }

    // REPOSITORIES
    single<AuthenticationRepository> { AuthenticationRepositoryImpl(get(), get())}
    single<RestaurantRepository> { RestaurantRepositoryImpl(get())}
    single<ProfileRepository> { ProfileRepositoryImpl(get())}
    single<TableRepository> { TableRepositoryImpl(get())}
    single<MenuRepository> { MenuRepositoryImpl(get())}
    single<OrderRepository> { OrderRepositoryImpl(get()) }

}
```

Ilustració 69 - Koin example

El Front es comunica amb dos serveis. Un és el servidor Back-end que és el que realitza la gran majoria de processos. L'altre servei amb el qual es comunica és amb Firebase, peró unicament per a obtenir el Token de l'usuari utilitzat per a les peticions a Back. L'aplicació de Front no es comunica més amb Firebase, ja que es busca reduir al màxim l'acoblament.

2.7 Testing

Pel que fa al testatge de l'aplicació s'han fet 2 tipus de tests.

El primer tipus de test ha sigut fet pel desenvolupador dissenyant plans de proves per a cada funcionalitat posant especial èmfasis en els casos més crítics (absència de dades, o dades errònies…) Es realitzaven les proves i en cas d'haver algun error, es solucionava abans de tancar el Sprint.

D’altra banda també s’han fet ús de potencials usuaris per a testejar. Al final de cada sprint s’ha donat l’aplicació a diferents usuaris perquè realitzin diverses proves de les històries d’usuaris (End to end).
A causa de la falta de temps, i a diferents imprevistos són els únics tests que s’han pogut realitzar. No s’han pogut fer proves unitàries ni de cap altre tipus. Tots els tests han sigut de manera manual, executant les diferents històries d’usuari, amb un pla de proves per a cadascuna, sempre buscant i tenint en compte els punts més crítics on l’aplicació podia fallar.

Al fi del procés de testatge, es feia preguntes als usuaris testers a veure si havien trobat algun error, o tenien alguna proposta de millora.

3. Conclusions

La planificació inicial no s'ha pogut seguir al 100% a causa de diversos motius, tot i això s'ha pogut anar adaptant i modificant la planificació sobre la marxa per a poder aconseguir la majoria d'objectius fixats.

El principal objectiu s'ha pogut aconseguir, que era poder obtenir una aplicació final totalment funcional per a automatitzar el procés d'un restaurant. Tot i no poder implementar totes les funcionalitats plantejades inicialment. L'aplicació és totalment funcional.

A més, s'ha aconseguit un software flexible i adaptable, que permet realitzar modificacions de manera senzilla. I s'han separat els processos més complexos a un servidor backend, que és consumit per l'aplicació Android. D'aquesta manera és més senzill en un futur poder realitzar l'aplicació per a iOS, en aquest cas només caldrà consumir la informació del backend i presentar la informació.

Tot i que els principals objectius s'han obtingut, hi ha diverses funcionalitats que no s'han pogut implementar, igual que no s'han pogut realitzar les proves unitàries que es volien implementar. Tot i que el software s'ha pogut testejar de manera correcta gràcies a diversos usuaris que s'han utilitzat per a testejar les històries d'usuari.

El motiu de no poder assolir tots els objectius que s'havien estipulat és que no s'ha pogut complir la planificació que havia degut a problemes de salut. No s'ha pogut dedicar el temps que s'havia fixat inicialment, i això ha provocat no poder implementar tot el que s'havia planificat.

La planificació inicial era l'adequada, encara que hagués sigut millor guardar una mica de temps per a poder rectificar en cas de possibles imprevistos, com el que ha passat. I d'aquesta manera no s'hagueren hagut de reduir les funcionalitats.

Han quedat bastants propostes de futur, les que ja s'havien planificat inicialment, més les funcionalitats que a causa de la falta de temps no s'han pogut implementar.
La principal funcionalitat que no s'ha pogut implementar i que queda com a proposta de millora és la implementació de diferents mètodes de pagament per al compte. Actualment només s'ha implementat el pagament en efectiu (demanar el compte al cambrer amb l'app).
Un altra funcionalitat que es volia implementar era guardar els aliments de cada plat i que l'usuari tingués un llistat amb els aliments que és al·lèrgic, i així evitar que els pogués demanar.
D'altra banda de les funcionalitats que s'havien plantejat com a opcionals s'ha implementat una, ja que facilitava molt el procés d'ocupació de les taules del restaurant. Per això, s'ha implementat la funcionalitat per a poder generar el codi QR per a les taules del restaurant, i la funcionalitat per a escanejar el codi QR i ocupar la taula.

Hi ha diverses propostes de millora que fan que sigui un projecte amb molt treball de futur per a explotar. Tot i això, l'aplicació actual és molt útil pels restaurants.

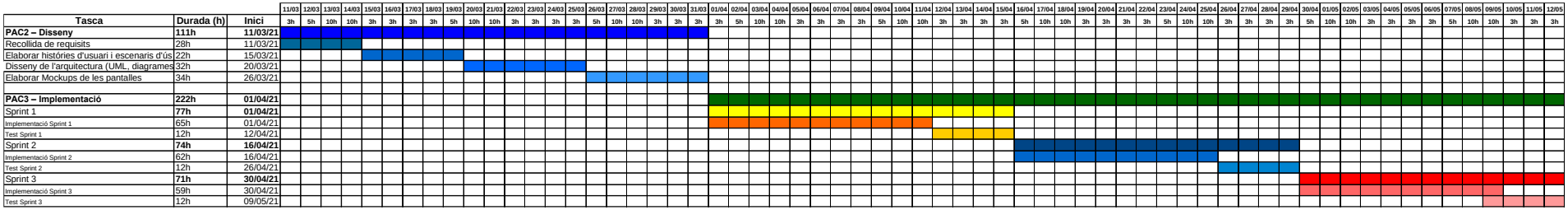
4. Glossari

- **Easy Order**: Nova aplicació mòbil per la gestió i optimització de restaurants, disponible per a clients i propietaris.

5. Bibliografia

1. Documentación | Firebase: https://firebase.google.com/docs/ (darrera consulta 25/05/2021)
2. Firebase Authentication for Spring Boot Rest API: https://thepro.io/post/firebase-authentication-for-spring-boot-rest-api-5V (darrera consulta 01/05/2021)
3. Heroku | Login: https://id.heroku.com/login (darrera consulta 01/06/2021)
4. BusinessInsider: https://www.businessinsider.es/ha-reinventado-sector-restauracion-durante-pandemia-764157 (darrera consulta 01/04/2021)
5. ¿Cómo afronta la hostelería este 2021? | Estar donde estés: https://estardondeestes.com/movi/es/articulos/como-afronta-la-hosteleria-este-2021 (darrera consulta 01/04/2021)
6. Formularios de Google: https://docs.google.com/forms/u/0/ (darrera consulta 31/03/2021)
7. Tablero EO: tablero àgil | Jira: https://easy-order.atlassian.net/jira/software/projects/EO/boards/1 (darrera consulta 01/06/2021)
8. Medium: https://medium.com/firebase-tips-tricks/how-to-work-synchronously-with-firebase-coroutines-livedata-mvvm-clean-architecture-b2f72638ef61 (darrera consulta 01/06/2021)

6. Annexos



Annex 1- Diagrama de Gantt