Easybill

Memòria de Projecte Final de Màster

Desenvolupament de llocs i aplicacions web

Memòria final

Autor: Carles Canellas Crusellas

Consultor: Víctor Cuervo

Professor: Carlos Casado Martínez

4 de gener de 2020

Resum

Easybill és una aplicació full stack amb les funcionalitats bàsiques de facturació. Arquitectònicament està basada en un model client -servidor – gestor de base de dades. El client (o frontend) està programat en React, i fa crides de tipus REST API al servidor (o backend). Les dades s'emmagatzemen remotament en una base de dades de tipus MongoDB, que utilitza el format NoSQL. A part, inclou un petit servei que ofereix a l'aplicació una llista de municipis amb els seus codis postals.

Paraules clau: React, REST API, NodeJS, MongoDB, Fullstack

Abstract

Easybill is a fullstack application with basic billing features. It is arquitecturally based on a client – server - database manager model. The client (or frontend) is programmed in React, and makes REST API calls to the server (or backend). The data is stored remotely in a MongoDB database, which uses the NoSQL format. In addition, it includes a small service that offers the application a list of cities with their postal codes.

Keywords: React, REST API, NodeJS, MongoDB, Fullstack

Índex

Figures i taules

Llistat d'imatges, taules, gràfics, diagrames, etc., numerades, amb títols i les pàgines on apareixen.

Índex de figures

s

Taula de figures

1. Introducció

1.1 Context i justificació del treball

Com a autònom em trobo amb la necessitat de portar un registre de les diferents factures que faig pels clients, així com una forma ràpida de generar-les.
Inicialment, suposo que, més o menys com tothom, portava aquest registre en un full de càlcul, i generava les factures a partir d'una plantilla d'un document de text. Aquest mètode, però, té força inconvenients a mesura que la quantitat de factures va creixent. Per exemple:

- Necessita una sincronització manual entre les dades del full de càlcul i les factures emeses.
- Està subjecte a errors manuals de comptabilitat.
- És tediós en feines repetitives, com per exemple inserir les dades dels clients a la factura.
- No permet fer consultes personalitzades habituals.

És per això que m'ha semblat interessant fer un programa de facturació bàsic.

1.2 Objectius del Treball

Els objectius abarquen aspectes tant pràctics com d'aprenentatge professional:

- Crear una aplicació que permeti portar un control de clients i factures, i que generi factures per poder-les enviar i tenir-les registrades.
- Fer una aplicació segura.
- Aprofundir en les aplicacions client-servidor amb un *backend* que proporciona una API de tipus REST i un *frontend* que es comunica amb aquesta API tant per consumir com per modificar les dades.
- Aprofundir en l'ús de bases de dades de tipus NoSQL, en particular MongoDB.
- Aprofundir en la programació de frontend amb el *framework* React, així com altres tecnologies associades (per exemple, Redux).
- Aprofundir en els servidors basats en les tecnologies NodeJS i Express.
- Crear una aplicació PWA.
- Portar un control i gestió del treball a través de VCS.

1.3 Enfocament i mètode seguit

Per dur a terme el projecte, m'he inclinat per una arquitectura client-servidor, que es comuniquen entre ells mitjançant una REST API. Aquesta arquitectura ens proporciona una bona flexibilitat, ja que separa la part d'interfície d'usuari de la gestió de les dades. Això ens permet, per exemple, consumir les dades des de diferents interfícies, o fer canvis a la part del servidor sense interferir en la interfície d'usuari.

El *frontend* he decidit programar-lo en React. És un *framework* Javascript que està adquirint molta popularitat. Gràcies a la seva simplicitat i flexibilitat, avui en dia molts projectes estan realitzats en React, com Facebook, Airbnb, Github, Instagram... i les estadístiques mostren com la seva popularitat creix cada cop més.

Tal com es pot veure a l'informe de HackerRank https://research.hackerrank.com/developer-skills/2019, Javascript és el llenguatge més popular des del 2018. És, per tant, un llenguatge amb un futur molt sòlid en què val molt la pena aprofundir. Igualment, es veu que la popularitat de React respecte altres *frameworks*, com per exemple Angular, està augmentant any a any. Aquests factors argumenten com a encertada la decisió d'escollir Javascript com a llenguatge de programació tant pel *frontend* com el *backend*, i React com a *framework* pel *frontend*. Aquesta tendència també es pot veure a l'informe del 2020 del mateix web https://research.hackerrank.com/developer-skills/2020

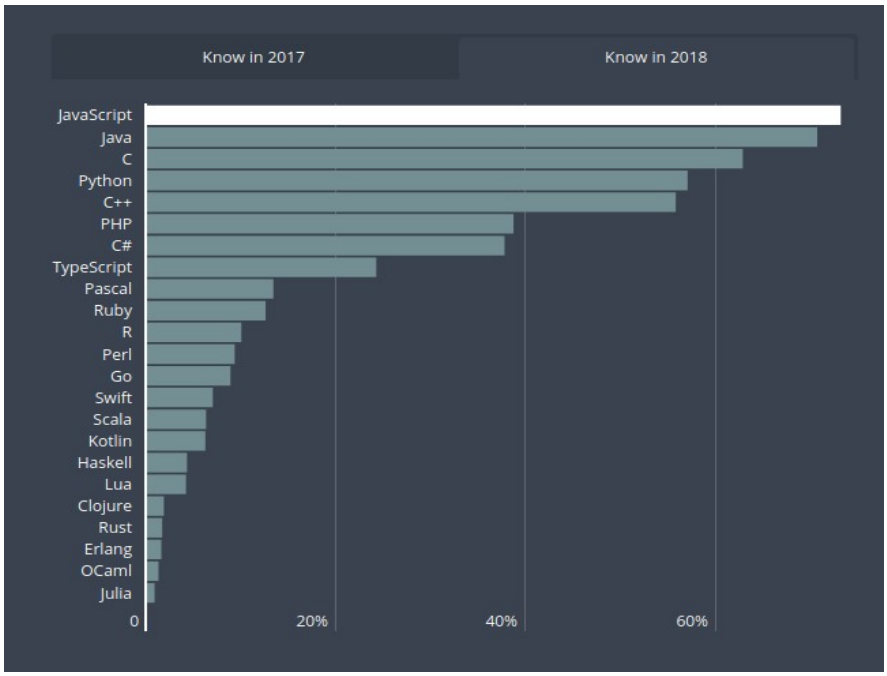


Figura 1: El 2018 Javascript ja era el llenguatge de programació més conegut. Font: https://research.hackerrank.com/developer-skills/2019

		2020	2019	2018
JavaScript		1	1	2
Java		2	2	1
C		3	3	3
Python		4	4	5
C++		5	5	4
C#	▲	6	7	6
PHP	▼	7	6	7
TypeScript		8	8	8
Pascal		9	9	9
R		10	10	10

Figura 2: El 2020 Javascript segueix essent el llenguatge de programació més conegut.
Font: https://research.hackerrank.com/developer-skills/2020

Figura 3: Els programadors de Javascript són els més buscats el 2020. Font:
https://research.hackerrank.com/developer-skills/2020

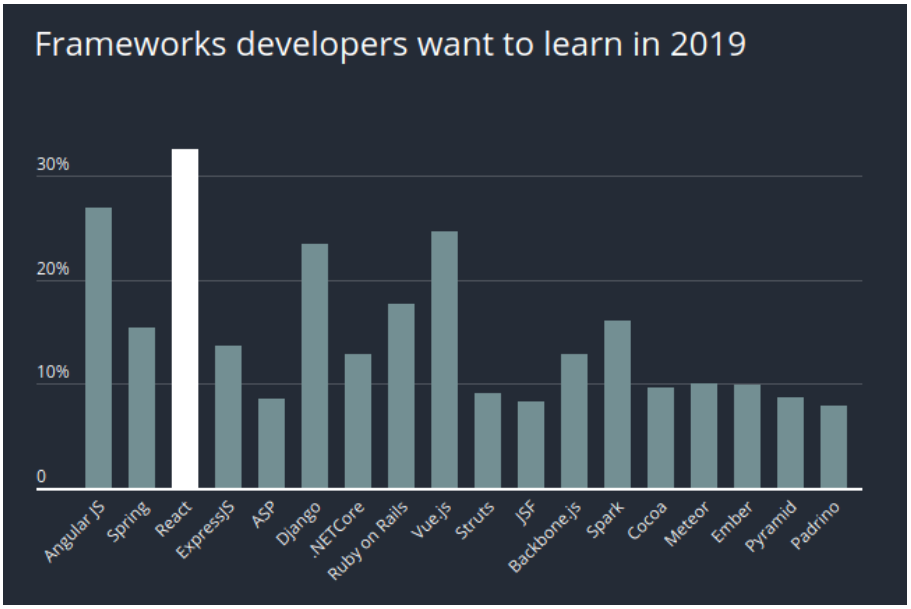


Figura 4: Els desenvolupadors volien conèixer preferentment React el 2019. Font: https://research.hackerrank.com/developer-skills/2019

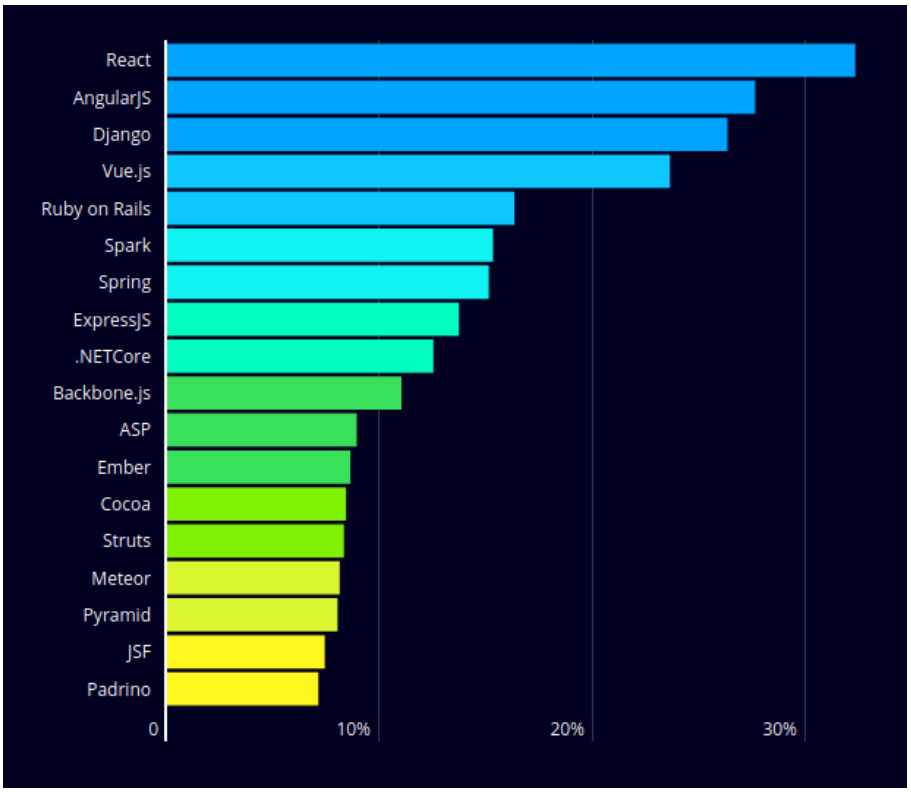


Figura 5: El 2020 React segueix essent el framework preferit. Font: https://research.hackerrank.com/developer-skills/2020

Dins de React, una de les llibreries més importants que he utilitzat és Redux. Permet gestionar les dades de l'aplicació al llarg del seu cicle de vida, centralitzant-les en un únic lloc. Així se simplifica el pas de dades entre components, i permet modificar de forma senzilla i centralitzada les estructures de dades. Un altre avantatge interessant en el procés de desenvolupament és que, gràcies a

l'extensió per navegadors Redux Devtools, podem fer un seguiment de les dades cada cop que es modifiquen.

La part del *backend* he decidit programar-la també en Javascript. El fet d'utilitzar el mateix llenguatge que el *frontend* simplifica la programació de tot el projecte. D'altra banda, NodeJS ofereix un servei web lleuger i ràpid, i gràcies a l'extensió Express permet programar REST API d'una forma bastant simple. A més, el procés d'instal·lació en servidor remot, així com la integració contínua (CI) són força senzills.

Pel que fa a la base de dades, m'he inclinat per MongoDB, que és de tipus NoSQL. En el tipus d'aplicació que estem parlant, aquesta decisió no té una justificació tècnica clara, ja que les dades d'un sistema de facturació són molt estructurades i un sistema SQL també compliria perfectament els requisits. A nivell d'aprenentatge, però, trobo interessant endinsar-me en una tecnologia que està creixent ràpidament, i que en el cas particular de MongoDB és prou flexible i segura per ésser utilitzada en multitud d'aplicacions. A més, gràcies a l'extensió Mongoose, s'integra a la perfecció amb NodeJS.

La distribució de l'aplicació és en forma de *Progressive Web Application* (PWA), de forma que es pugui instal·lar en qualsevol plataforma mòbil, a més d'utilitzar-se des d'un navegador web.

Voldria subratllar també que aquest tipus d'aplicacions solen ser optimitzades per treballar amb un dispositiu amb una pantalla gran, ja que presenten les dades en format taula, poc indicat per dispositius amb pantalla petita, com telèfons mòbils. No obstant, crec que cal fer un esforç important d'adaptació a aquest tipus de dispositius, ja que avui en dia són les portes d'accés més habituals a les aplicacions web. Aquesta aplicació, per tant, té molt en compte la usabilitat en dispositius mòbils.

Una part important del procés de desenvolupament són els tests, i en particular els tests unitaris. Tant React com NodeJS tenen extensions per poder fer tests automàtics. També es poden utilitzar eines com Postman per testar tots els punts d'entrada de l'API REST. És interessant utilitzar TDD (Test Driven Development) en els casos on sigui raonable, la qual cosa es tradueix en una aplicació més robusta i un procés de desenvolupament més ordenat i flexible.

Per gestionar el codi font del *fronted* i el *backend*, així mateix com la documentació, s'utilitza el sistema de gestió de versions Git. En concret, es desa en un repositori remot de Gitlab. A part de les funcions pròpies d'un VCS, aquesta plataforma en permet d'altres molt interessants com agrupació de projectes en grups, documentació tipus markdown avançat tant dels grups com dels projectes, boards tipus Kanban, gestió avançada d'*issues*, *merge requests*, *continous integration and delivery (CI/CD)*...

Finalment cal tenir en compte també el procés de documentació. Una part d'aquesta documentació ha d'estar reflectida al codi, i una altra en un document separat. És recomanable fer-la durant el desenvolupament i no pas al final del projecte. Així s'evita que hi hagi parts que no hagin estat correctament documentades a causa de l'oblit.

1.4 Planificació del Treball

La base per fer la planificació són les dates de lliurament definides al calendari de l'assignatura.

- 29 de setembre de 2020: lliurament PAC1
- 28 d'octubre de 2020: lliurament PAC2
- 6 de desembre de 2020: lliurament PAC3
- 4 de gener de 2021: lliurament final

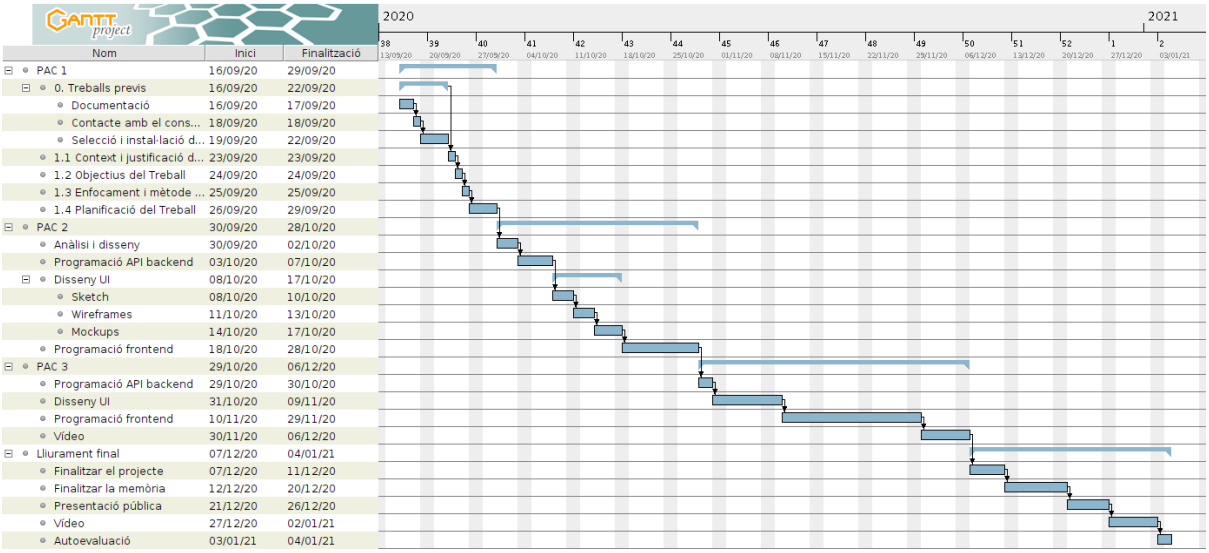
Per garantir aquests lliuraments, cal descomposar-los en altres tasques.

Deixant de banda aquestes dates clau, crec que és més convenient afrontar el projecte des d'un enfocament àgil, de forma que no calgui haver finalitzat una tasca per començar la següent. Així doncs, tot i que les tasques estan molt delimitades al diagrama de Gantt, en realitat ens servirà només de guia, ja que hi haurà solapaments entre elles.

Les fites són també les marcades per les PAC:

- **PAC1**: part de la memòria on s'introdueix el projecte, se'n defineixen els objectius, metodologies, funcionalitats... Es materialitza la planificació. S'instal·la tot l'entorn de treball en una màquina local.
- **PAC2**: consolidació dels requisits i les bases del projecte. A nivell d'implementació:
  - Es realitzarà l'anàlisi i el disseny, amb els respectius diagrames.
  - Es definirà la base de dades.
  - Es crearà el conjunt del *backend* (definició de l'API, controladors, lògica, tests...).
  - Es posarà la base per la interfície d'usuari, creant la majoria dels *sketch*, *wireframes* i *mockups*.
  - Es començarà la programació del *frontend*.
- **PAC3**: primera versió funcional del projecte, encara pendent de possibles modificacions. Realització d'un vídeo explicant el funcionament d'aquesta versió. Es reserven uns dies per fer les modificacions que calgui al *backend*. Se segueix amb el disseny de la interfície d'usuari, especialment la part de *mockups*. Es fa la major part de programació del *frontend*. Es realitza un vídeo explicant el funcionament de la primera versió de l'aplicació, que ha d'estar a punt al final d'aquesta fase.

- **Lliurament final**: versió definitiva de l'aplicació. Fer-ne el desplegament en un servidor remot per tal que sigui accessible al professorat. Tancament de la memòria. Presentació pública en vídeo. Autoevaluació.



Nom	Inici	Finalització
PAC 1	16/09/20	29/09/20
0. Treballs previs	16/09/20	22/09/20
Documentació	16/09/20	17/09/20
Contacte amb el cons...	18/09/20	18/09/20
Selecció i instal·lació d...	19/09/20	22/09/20
1.1 Context i justificació d...	23/09/20	23/09/20
1.2 Objectius del Treball	24/09/20	24/09/20
1.3 Enfocament i mètode ...	25/09/20	25/09/20
1.4 Planificació del Treball	26/09/20	29/09/20
PAC 2	30/09/20	28/10/20
Anàlisi i disseny	30/09/20	02/10/20
Programació API backend	03/10/20	07/10/20
Disseny UI	08/10/20	17/10/20
Sketch	08/10/20	10/10/20
Wireframes	11/10/20	13/10/20
Mockups	14/10/20	17/10/20
Programació frontend	18/10/20	28/10/20
PAC 3	29/10/20	06/12/20
Programació API backend	29/10/20	30/10/20
Disseny UI	31/10/20	09/11/20
Programació frontend	10/11/20	29/11/20
Vídeo	30/11/20	06/12/20
Lliurament final	07/12/20	04/01/21
Finalitzar el projecte	07/12/20	11/12/20
Finalitzar la memòria	12/12/20	20/12/20
Presentació pública	21/12/20	26/12/20
Vídeo	27/12/20	02/01/21
Autoevaluació	03/01/21	04/01/21

2. Objectius

2.1 Principals

- Planificar i executar una aplicació web en totes les seves fases, des de la seva concepció fins a obtenir una aplicació completament funcional.
- Crear una aplicació *fullstack*, o sigui que englobi tant un *frontend* (aplicació que utilitzarà l'usuari des del seu dispositiu) com un *backend* (aplicació que gestionarà tant les dades com les peticions que vinguin del *frontend*).
- Crear una aplicació pràctica, per tant que no només sigui funcional sinó que sigui útil per un sector d'usuaris del món real.
- Que l'aplicació compleixi criteris d'usabilitat i que els potencials usuaris tinguin una bona experiència d'usabilitat en un ventall ampli de dispositius.
- Crear una aplicació amb una base de dades NoSQL, preveient-ho com a tecnologia a utilitzar en futures aplicacions (en aquest cas, MongoDB).
- Fer una aplicació segura.
- Realitzar un control de l'estat de l'aplicació a través de VCS.

2.2 Secundaris

- Crear una aplicació PWA, que es pugui instal·lar en un dispositiu mòbil com si fos una aplicació nativa i que permeti treballar-hi sense accés a xarxa.
- Utilitzar el paradigma TDD (Test Driven Development).

3. Escenari

La facturació és una necessitat de sectors molt diversos de la societat, des de persones físiques que facturen feines de tant en tant, autònoms que necessiten no només fer les factures sinó portar un registre de les que s'han generat, cobrat... fins a empreses petites i grans.

Tenint en compte aquesta necessitat, no és d'estranyar que al mercat hi hagi disponibles un ventall força ampli de programes informàtics que fan aquesta funció. Alguns, com Contasimple, són solucions que ofereixen funcionalitats basades només en la facturació (incloent pressupostos, factures, clients, pagaments de taxes...). D'altres, com Odoo, ofereixen la facturació com a part integrada d'un sistema ERP més complex.

El cost d'aquestes solucions també és molt variable, i de fet cadascuna sol oferir diferents plans amb costos segons les característiques de cada pla. Fins i tot hi ha solucions de codi obert que es poden utilitzar sense cap cost. Aquest és el cas d'Odoo i altres productes, que es poden descarregar, instal·lar en un servidor propi, configurar i utilitzar sense pagar. Generalment, però, fer-ho d'aquesta forma requereix un alt coneixement del programa, pel que algunes empreses ofereixen aquests programes a punt per ser utilitzats a canvi del pagament d'una quota.

Un dels avantatges de crear una aplicació de facturació és la gran flexibilitat de funcions que permet. N'hi ha algunes de bàsiques, que són les que fan referència als diferents elements que consten a la factura: usuari que la genera, client a la que va dirigida, línies que componen cada factura i impressió de la factura. A partir d'aquí, es pot complicar tant com es vulgui: gràfiques d'evolució de la facturació, creació de pressupostos, exportació i importació d'altres formats, integració amb altres sistemes (CMS, online shopping....), pagament de taxes... Això és molt interessant a nivell de disseny i programació, ja que requereix un sistema prou flexible per poder incorporar futures funcionalitats de manera prou ràpida i eficient.

D'altra banda, a l'haver-hi un mercat tant saturat, la pregunta és si té sentit crear una altra aplicació d'aquest tipus. El meu plantejament és, més enllà de l'ús personal que en pugui fer, adquirir l'experiència que em suposarà i poder tenir un projecte que pugui mostrar i que sigui representatiu de les meves habilitats com a desenvolupador d'aplicacions web.

4. Continguts

Els continguts de l'aplicació prioritzen la senzillesa per sobre de la complicació i la quantitat de funcionalitats.

- **Pàgina d'entrada**: entrar com a usuari registrat, registrar-se com a nou usuari i restaurar contrasenya en cas d'oblit.
- **Llistat de factures**: llistat de totes les factures creades actualment amb la informació bàsica i possibilitat de filtratge. Accés a edició i impressió de cadascuna de les factures i creació d'una factura nova.
- **Edició de factura**: funcions CRUD sobre una factura. Canvi de les dades en cas que la factura estigui activa. Seguiment del cobrament i el pagament de les taxes associades. Emissió, anul·lació i eliminació d'una factura. Llistat de línies que conté la factura. Resum de la factura.
- **Edició de línies de factura**: funcions CRUD sobre una línia de factura.
- **Llistat de clients**: llistat de tots els clients creats actualment amb la informació bàsica i possibilitat de filtratge. Accés a edició de cada client i creació d'un client nou.
- **Edició de client**: funcions CRUD sobre un client.
- **Dades personals**: edició del perfil d'usuari.
- **Impressió**: vista prèvia de la factura tal com quedarà impresa.

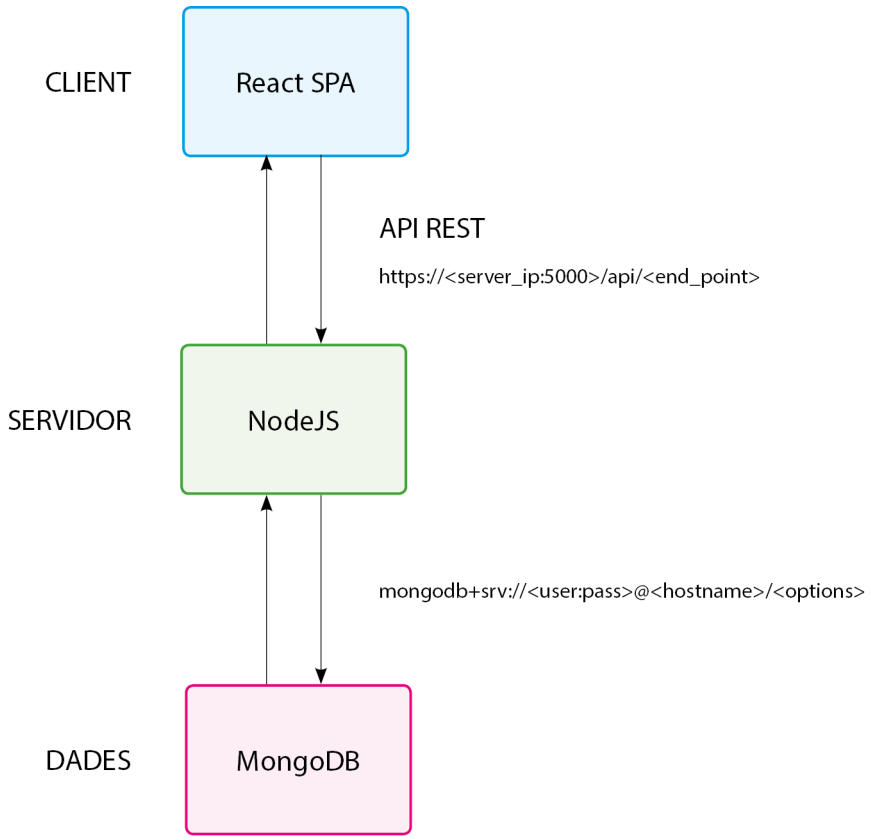
5. Metodologia

La metodologia emprada es deriva de les següents premisses:

1. Cal complir estrictament amb els terminis de lliurament de les PAC, amb els requisits que s'hi demanen
2. Tenint en compte que cal complir el punt 1, utilitzar una metodologia àgil, és a dir posar-se fites que corresponguin a diferents funcionalitats de l'aplicació.
3. Tenint en compte que cal complir el punt 1, i tenint en compte també el punt 2, seguir la planificació inicial de forma orientativa.
4. Dissenyar i crear l'arquitectura de l'aplicació. Instal·lar els elements bàsics que la componen per tal de poder iniciar el treball de desenvolupament. Alguns elements de l'arquitectura poden rebre modificacions al llarg del desenvolupament.
5. A nivell de programació, crear primer les estructures sobre les quals en depenguin d'altres. Per exemple, crear i testar abans el backend bàsic per tal que es pugui desenvolupar el frontend amb agilitat.
6. Refactoritzar el codi.
7. Documentar el procés a mesura que es va desenvolupant l'aplicació.
8. Treballs finals (vídeo de defensa, autoevaluació...).

6. Arquitectura de l’aplicació

L’arquitectura de l’aplicació és un sistema client-servidor, amb comunicació entre ells a través d’una REST API.
El client és una aplicació web SPA programada en React.
El servidor és una aplicació NodeJS que envia i rep dades per mitjà d’uns punts d’accés REST API a través del port 5000.
La comunicació entre client i servidor, en els casos que requereixi autenticació, es fa mitjançant un Json Web Token (JWT), amb un temps de vida d’1 dia.
Les dades s’’emmagatzemen remotament a MongoDB Atlas, una base de dades distribuïda oferta per MongoDB. La comunicació amb el servidor es fa de forma segura mitjançant una adreça web, un usuari i una contrasenya. Les funcions CRUD es fan a través de la llibreria de NodeJS Mongoose, que permet treballar amb models de dades i disposa de funcions avançades per dur a terme fàcilment les tasques més comunes.



7. Plataforma de desenvolupament

A nivell de hardware, s'ha utilitzat un portàtil HP Folio 1040 amb un processador Intel corei7, 8 GB de RAM i disc dur SSD de 256 GB. El sistema operatiu instal·lat és un Ubuntu 18.04 LTS.

A nivell de software, s'han utilitzat recursos locals i remots.

La llista de programari local és:

- Com a plataforma de programació, Microsoft Visual Studio Code.
- NodeJS per l'aplicació servidor, amb diverses llibreries de suport (destacar sobretot Express i Mongoose).
- React per l'aplicació client, amb diverses llibreries de suport (destacar Redux i Material).
- Diversos navegadors web, majoritàriament Mozilla Firefox, però també Google Chrome.
- Compass per gestionar l'estat de la base de dades remota.
- Postman per testejar les peticions HTTP a la REST API del servidor.
- Dia, per fer els diagrames UML.
- Com a editor de wireframes, DrawIO.
- LibreOffice per fer la memòria.
- Gantt per generar els diagrames de Gant.
- GIMP com a editor d'imatges.
- PowerPoint per les presentacions.
- Adobe Premiere per editar els vídeos.

La llista de programari remot és:

- Com a repositori VCS, Gitlab.
- Com a gestor de base de dades, MongoDB.

8. Backend

El *backend* ofereix un servei REST API que escolta el port 5000. Està programat en Javascript sobre NodeJS i es connecta a una base de dades remota MongoDB a través de la llibreria mongoose.

Els *endpoints* que ofereix l'API són els següents:

End point	Acció	Headers	Body	Retorna
POST api/auth/login	Entrar al sistema	--	Email, password	JWT
POST api/auth/register	Registre al sistema	–	Totes les dades de l'usuari	JWT
GET api/invoices	Llegir llista de factures	JWT	–	Llista de factures
GET api/invoices/:**number**	Llegir una factura segons un número	JWT	–	Factura cooresponent a **number**
POST api/invoices	Afegir una factura	JWT	Dades de la factura	Factura amb identificador **id** afegida
POST api/invoices/emit/:***id***	Emetre una factura	JWT	–	Factura amb identificador **id** emesa
POST api/invoices/cancel/:**id**	Anul·lar una factura	JWT	–	Factura amb identificador **id** anul·lada
PUT api/invoices/:**id**	Modificar una factura	JWT	Dades de la factura	Factura amb identificador **id** actualitzada
DELETE api/invoices/:**id**	Eliminar una factura	JWT	–	–
GET api/lines/:**invoice_id**	Llegir les línies d'una factura	JWT	–	Llista de línies corresponent a la factura **invoice_id**
GET api/lines/:invoice_id/:**line_id**	Llegir una línia de factura	JWT	–	Línia de factura amb identificador **line_id**
POST api/lines/:**invoice_id**	Afegir una línia a la factura **invoice_id**	JWT	Dades de la línia de factura	Línia de factura afegida
PUT api/lines/**:id**	Modificar una línia de factura	JWT	Dades de la línia de factura	Línia de factura modificada
DELETE api/lines/:**id**	Eliminar una línia de factura	JWT	–	–
GET api/clients	Llegir llista de clients	JWT	–	Llista de clients
GET api/clients/:**nif**	Llegir les dades d'un client	JWT	–	Client corresponent al **nif**
POST api/clients	Afegir un client	JWT	Dades del client	Client afegit
PUT api/clients/:**id**	Modificar el client identificat per **id**	JWT	Dades del client	Client modificat
DELETE api/clients/:**id**	Eliminar un client	JWT	–	–
GET api/users	Llegir les dades de l'usuari registrat	JWT	–	Dades de l'usuari registrat
PUT api/users	Modificar l'usuari registrat	JWT	Dades de l'usuari	Usuari modificat
DELETE api/users	Eliminar l'usuari registrat	JWT	–	--

Els *endpoints* sense *headers* d'autorització (login, register) són públics. Els que requereixen JWT són privats.

Les dades d'entrada es validen abans de ser processades. En cas d'error, aquest es retorna amb el codi corresponent.

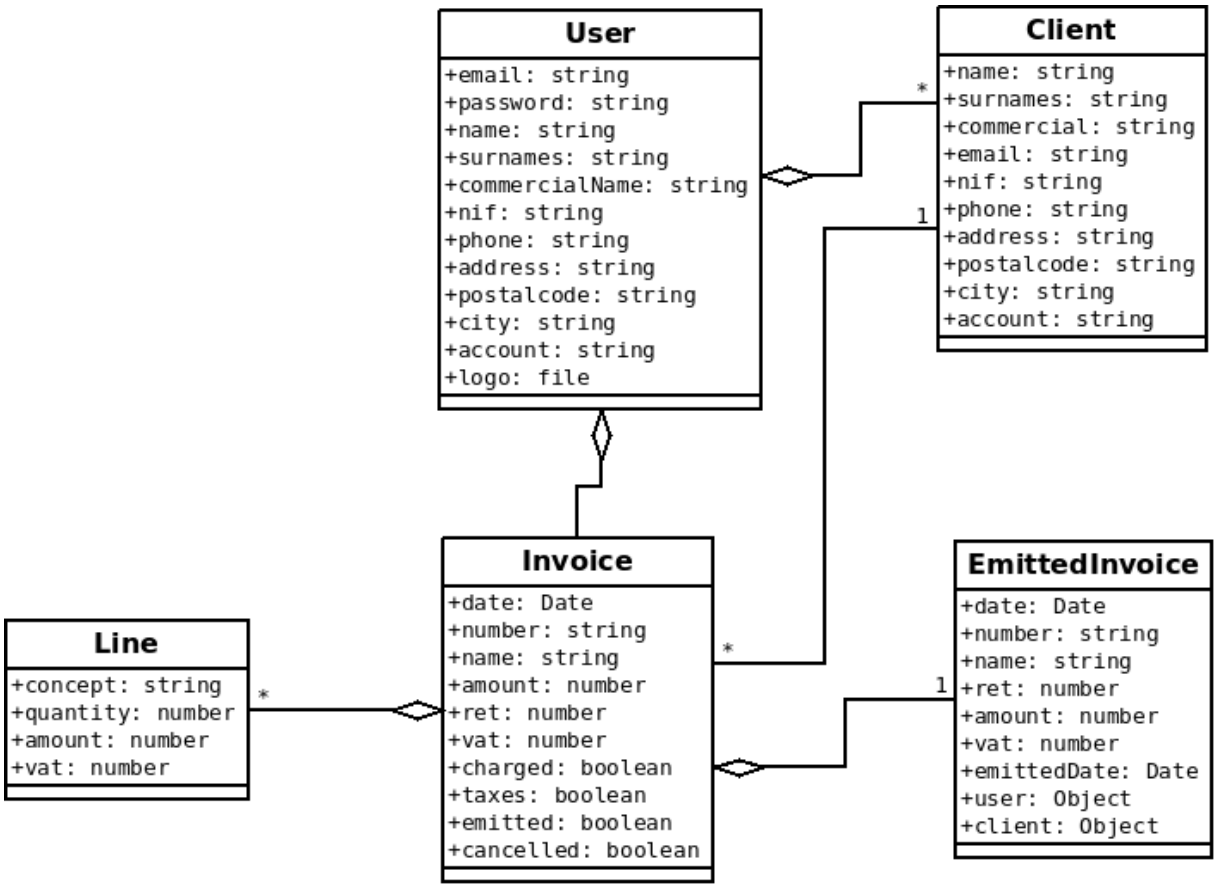
Per poder facilitar a l'usuari l'entrada del municipi i el codi postal de l'usuari i els seus clients, s'ha creat un nou servei API només per aquesta funció. El motiu de crear-lo a part és que, el més lògic seria extreure les dades d'un servei existent. Com que no he trobat cap servei que les ofereixi, he decidit fer-lo jo mateix però a part del backend, per tal que si convé la font de dades es pugui canviar fàcilment en un futur. Aquesta API escolta el port 5100, és pública i només té un endpoint.

End point	Acció	Headers	Body	Retorna
GET api/cities/**:substr**	Llegir llista de municipis el nom dels quals conté la cadena **substr**	--	--	Llista de municipis amb el seu codi postal

Les dades dels municipis estan extretes de la pàgina https://postal.cat

9. Diagrames UML

9.1 Diagrama estàtic



Tenim 4 classes:

User

Usuari del sistema. Qualsevol es pot registrar al sistema, sempre que no hi hagi 2 usuaris amb el mateix correu electrònic.

Invoice

Classe corresponent a les factures. Cada usuari té les seves factures i no pot veure ni modificar les de la resta d'usuaris.
L'atribut vat és derivat i correspon a la suma dels imports de l'IVA de les línies de comanda associades.
L'atribut amount és derivat i correspon a sumar per cada línia de factura **import * quantitat**.
Les factures no es poden modificar quan estan emeses o cancel·lades.

Line

Línies de factura agregades a una factura.

EmittedInvoice

Classe corresponent a les factures que ja han estat emeses. Desa la informació que existia en el moment d'emetre la factura, tant de la factura en sí, com de l'usuari i el client.

Client

Classe corresponent als clients. Cada usuari té els seus clients i no pot veure ni modificar els de la resta d'usuaris.

10. Prototips

Wireframes

S'han generat 2 grups de wireframes a baixa resolució, un per pantalla petita (mobile) i l'altra per pantalla gran (desktop). Els 2 estan en els fitxers adjunts mobile.drawio i desktop.io. A part de la disposició dels diferents elements a cada pantalla, incorporen la interacció i un arbre de navegació.

Algunes mostres de wireframes (es poden veure sencers a l'annex 4):

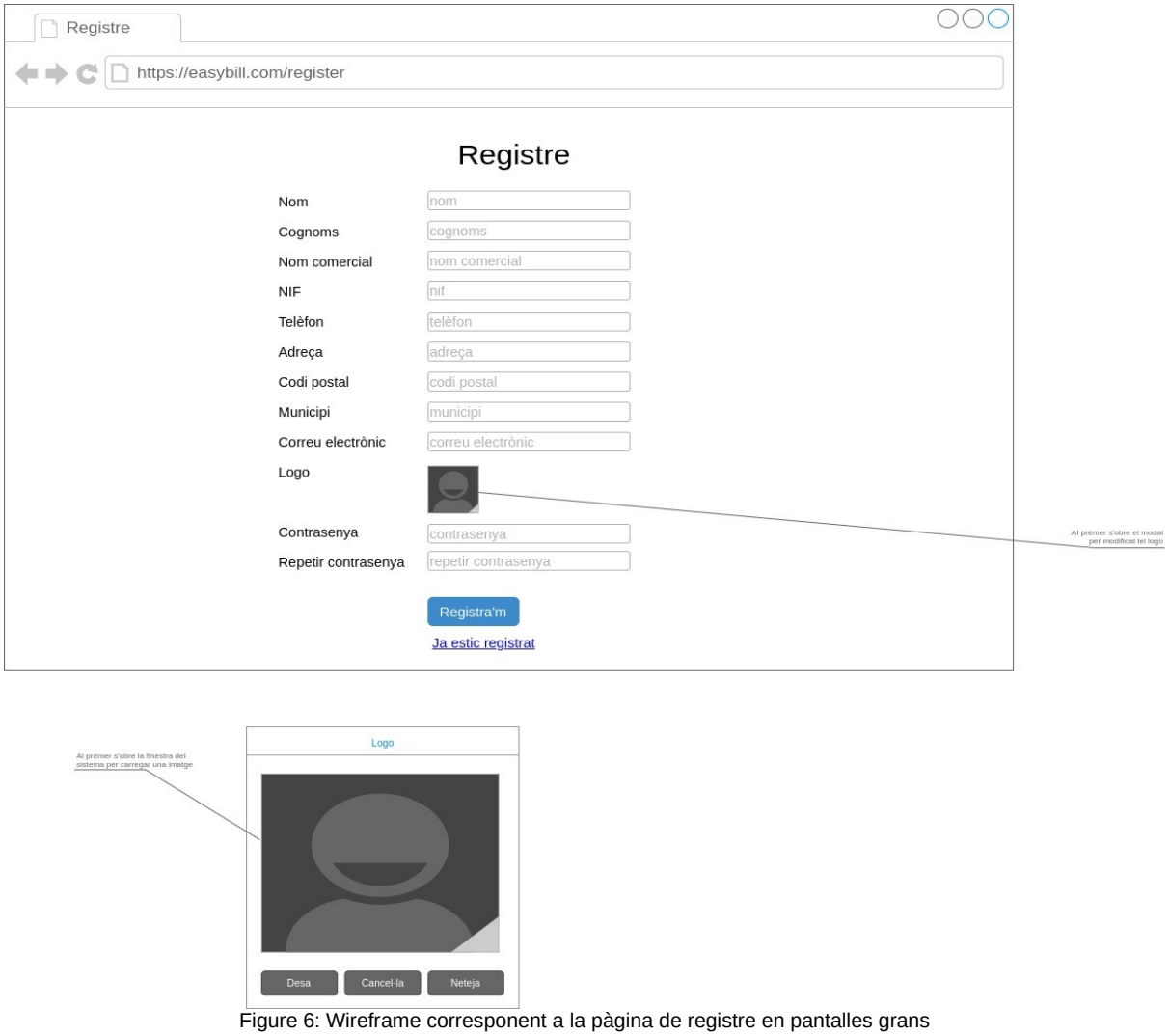


Figure 6: Wireframe corresponent a la pàgina de registre en pantalles grans

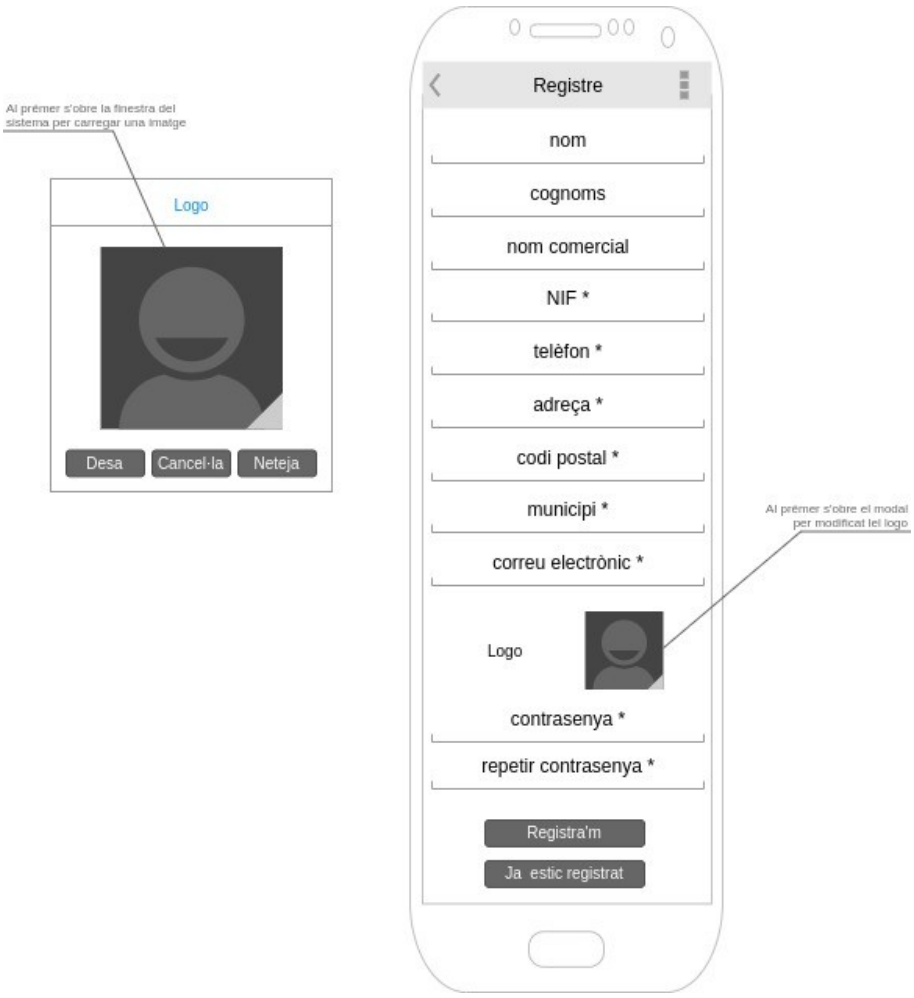


Figure 7: Wireframe corresponent a la pàgina de registre per pantalles petites

Prototips

S'ha utilitzat la llibreria **material-ui** per realitzar la interfície d'usuari. Aquesta decisió ha determinat la visualització final de l'aplicació. Per aquest motiu, no s'han fet prototips i s'ha dissenyat l'aplicació directament sobre la llibreria.

Algunes mostres de les pàgines de l'aplicació

Figure 8: Pàgina d'inici per pantalles petites

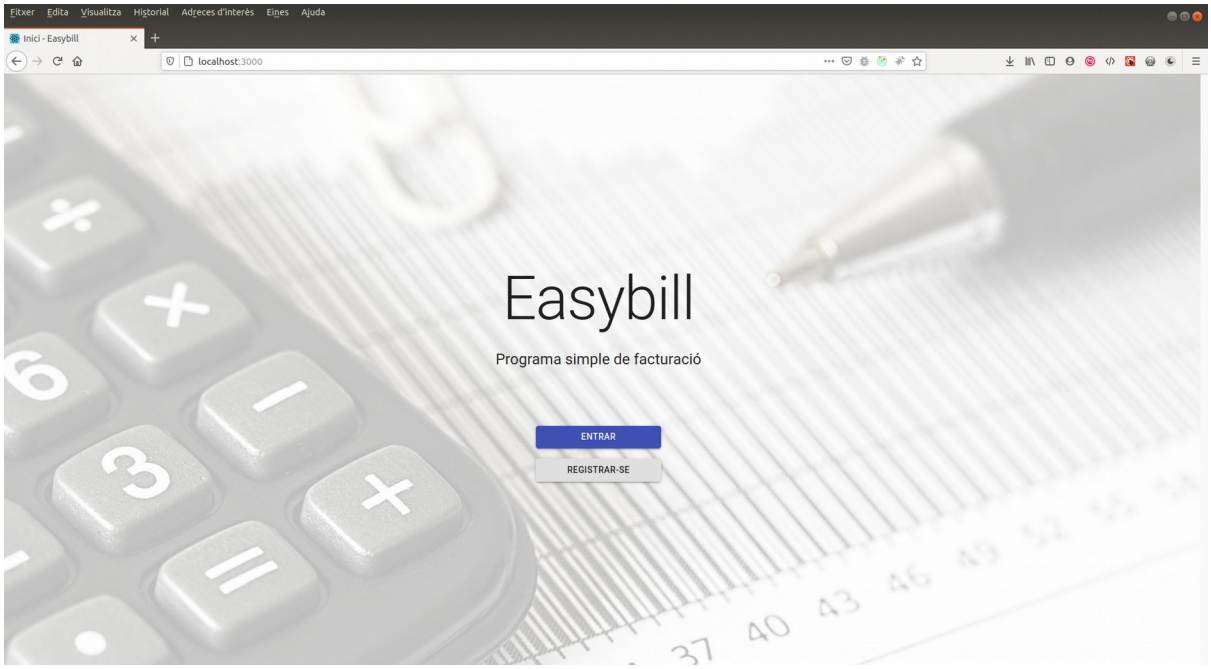


Figura 9: Pàgina d'inici per pantalles grans

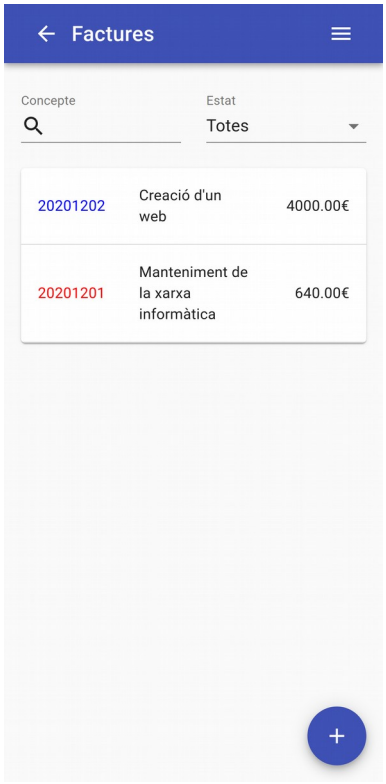


Figura 10: Pàgina de factures per pantalles petites

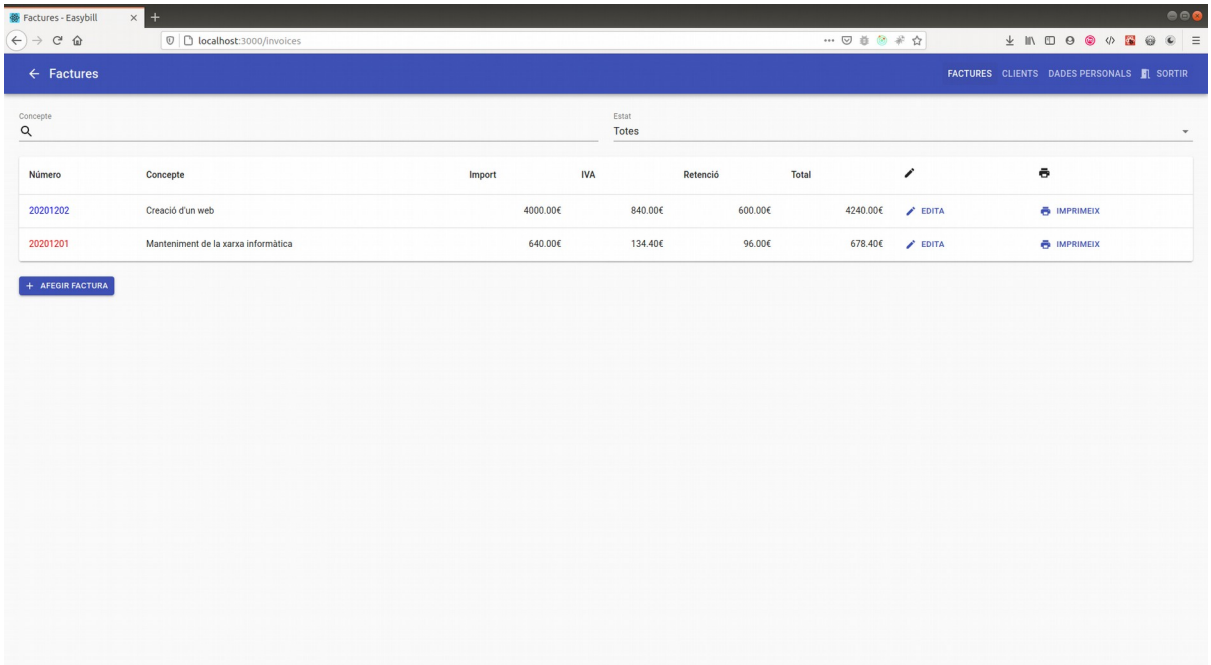
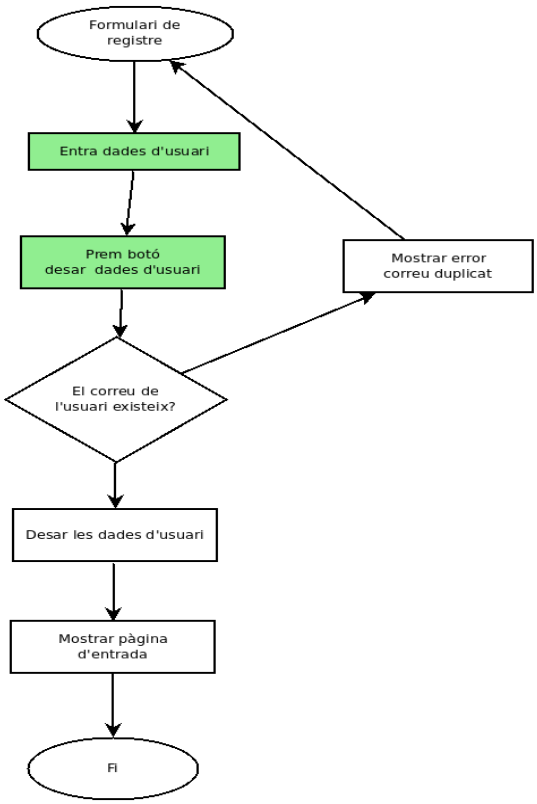


Figura 11: Pàgina de factures per pantalles grans

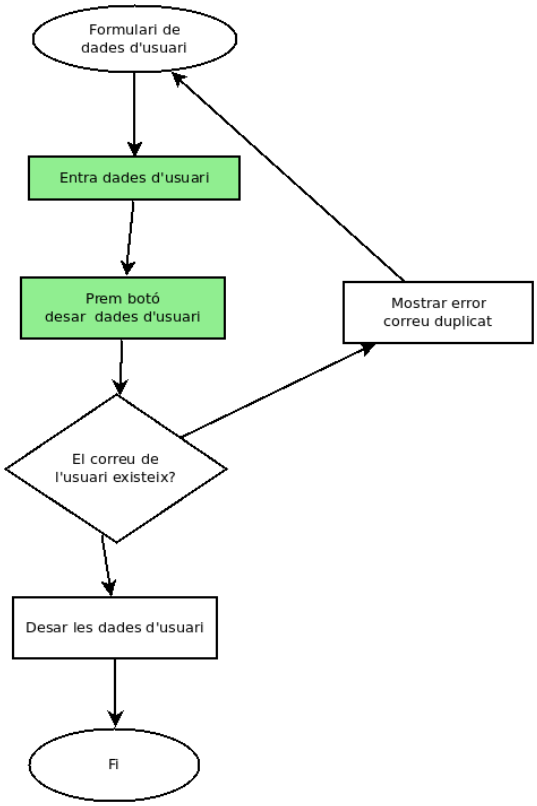
11. Diagrames de flux

11.1 Registre



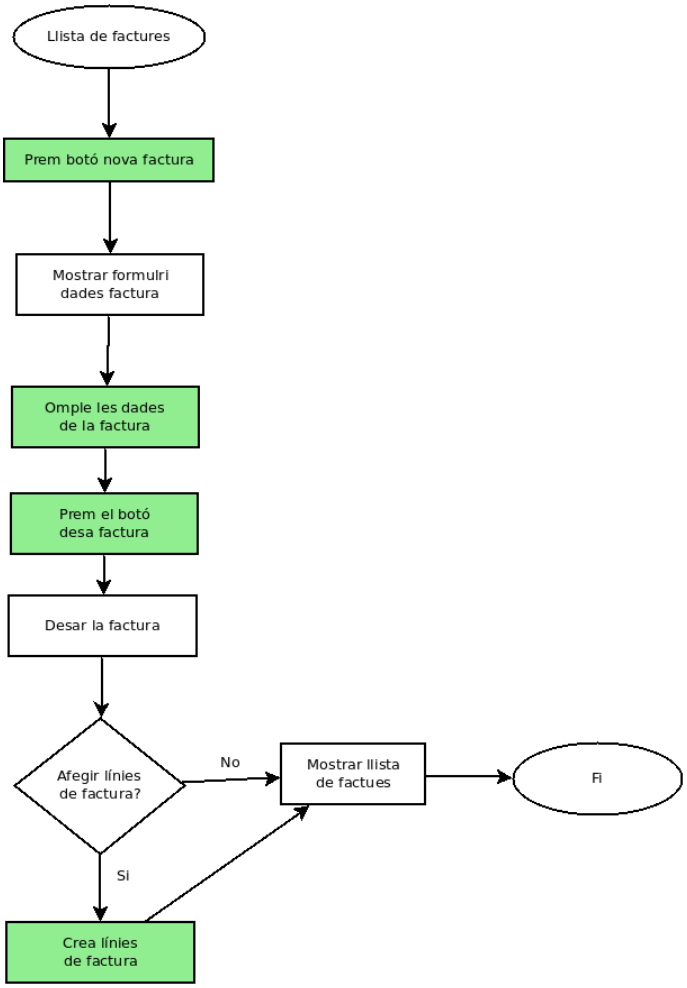
A la pàgina de registre s'hi accedeix des de la pàgina inicial, sense necessitat d'estar registrat. L'usuari omple les seves dades i prem el botó desar. En cas que no existeixi cap altre usuari amb el mateix correu, el sistema desa les dades i mostra la pàgina d'entrada (login).

11.2 Modificar perfil



A la pàgina de modificació del perfil s'hi entra des del menú.
L'usuari entra les seves dades. En el moment de desar-les, el sistema comprova si el correu electrònic ja existeix. Si no és el cas, desa les dades al sistema.

11.3 Crear factura

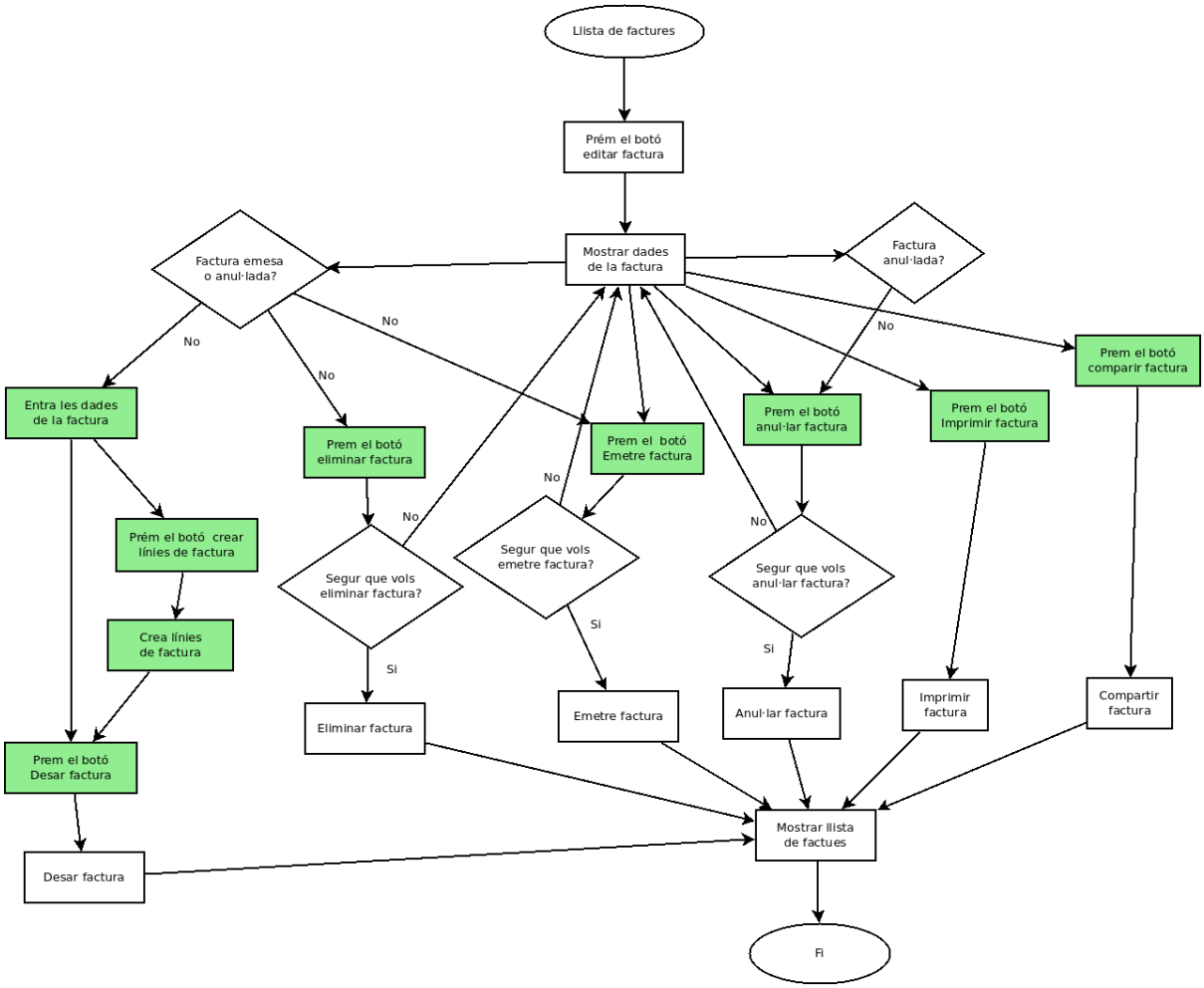


A partir de la llista de factures, l'usuari escull crea una factura nova.

Després de desar la factura, el sistema demana a l'usuari si vol afegir línies a la factura. Si és així, se li mostra el formulari per poder-les afegir.

Finalment es redirigeix l'usuari al llistat de factures, on es mostra la factura creada.

11.4 Editar factura



A partir de la llista de factures, l'usuari escull editar una factura de la llista.

Les dades de la factura només es poden editar si la factura no ha estat emesa ni anul·lada. L'usuari omple les dades de la factura i, si s'escau, afegeix o edita les línies de la factura. Finalment l'usuari escull desar la factura i el sistema desa els canvis.

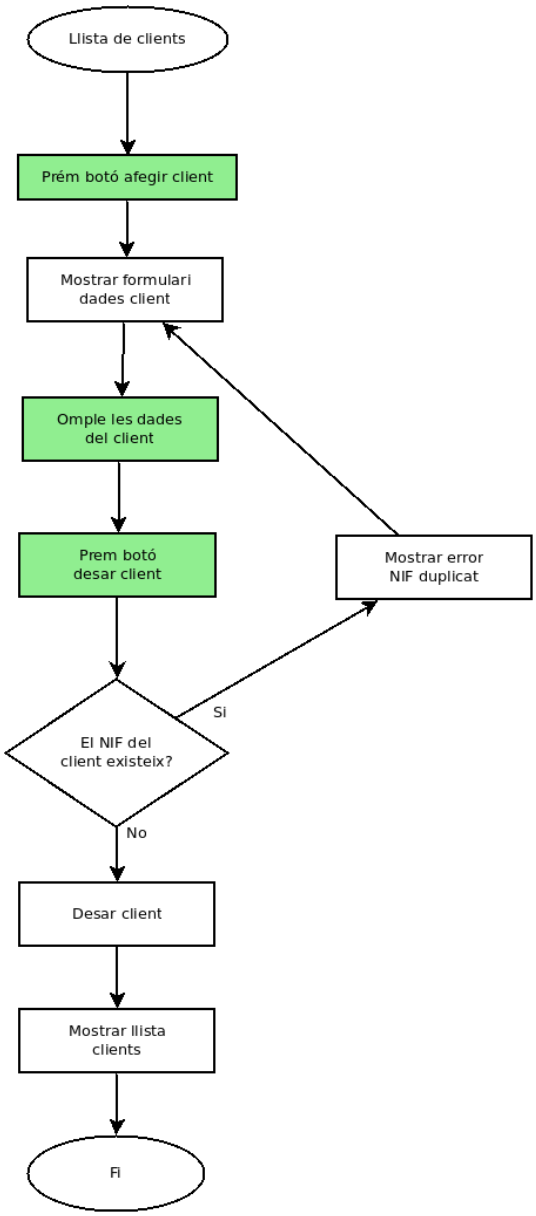
Si la factura no ha estat emesa ni anul·lada, el sistema ofereix la possibilitat d'eliminar-la. En aquest cas, s'adverteix l'usuari que aquesta acció és irreversible, i es procedeix a l'eliminació si hi està conforme.

Si la factura no ha estat emesa ni anul·lada, el sistema ofereix la possibilitat d'emetre-la. En aquest cas, s'adverteix l'usuari que aquesta acció és irreversible, i es procedeix a l'emissió si hi està conforme.

Si la factura no ha estat anul·lada, el sistema ofereix la possibilitat d'anul·lar-la. En aquest cas, s'adverteix l'usuari que aquesta acció és irreversible, i es procedeix a l'anul·lació si hi està conforme.

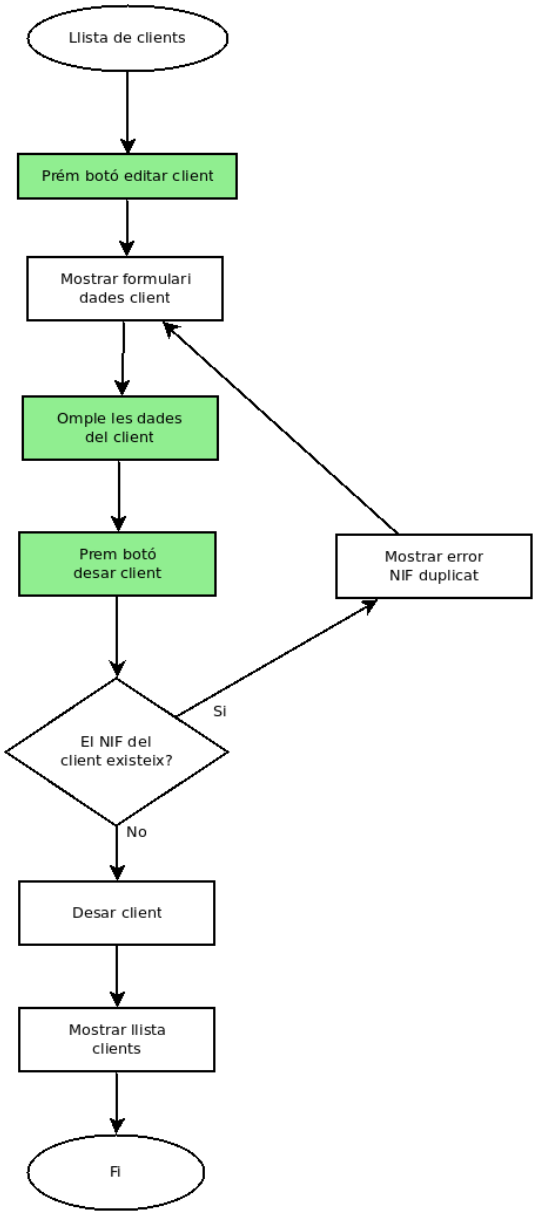
Altres accions que es poden realitzar per qualsevol estat de la factura són la seva impressió i compartició amb altres aplicacions del sistema (correu electrònic, missatgeria instantània...).

11.5 Crear client



A partir de la llista de clients, l'usuari pot crear un nou client.
El sistema pot afegir el nou client sempre que el NIF assignat al nou client no correspongui a cap altre client. En cas contrari, el sistema notificarà aquest error a l'usuari.

11.6 Editar client



El procés per editar un client és el mateix que per crear-lo.

12. Casos d’ús

Cu01: Login

Actor principal: usuari

Àmbit: sistema

Nivell d’objectiu: usuari

Precondicions: cap

Garanties mínimes: cap

Garanties en cas d’èxit: l’usuari podrà accedir al sistema i a les seves dades

Escenari principal d’èxit:

1. L’usuari entra el seu nom d’usuari i la contrasenya.
2. El sistema permet l’accés a l’usuari.

Extensions:

2a. El sistema no reconeix l’usuari

2a1. El sistema informa l’usuari de l’error

2a2. Tornem al pas 1

Cu02: Registre

Actor principal: usuari

Àmbit: sistema

Nivell d’objectiu: usuari

Precondicions: cap

Garanties mínimes: cap

Garanties en cas d’èxit: el sistema té un usuari nou amb les dades de registre

Escenari principal d’èxit:

1. L’usuari entra les dades amb les que es vol registrar
2. El sistema registra l’usuari amb les dades entrades.

Extensions:

2a. El correu electrònic de l’usuari ja està registrat a un altre usuari

2a1. El sistema informa l’usuari de l’error

2a2. Tornem al pas 1

2b. La contrasenya i la verificació de contrasenya no coincideixen

2b1. El sistema informa l’usuari de l’error

2b2. Tornem al pas 1

Cu03: Restaurar contrasenya

Actor principal: usuari

Àmbit: sistema

Nivell d’objectiu: usuari

Precondicions: cap

Garanties mínimes: cap

Garanties en cas d'èxit: el sistema té un usuari registrat amb la contrasenya nova

Escenari principal d'èxit:

1. L'usuari entra el correu electrònic amb el que s'ha registrat i prem el botó *No recordo la contrasenya*
2. El sistema envia un correu a l'usuari amb un token i un enllaç a la pàgina de restauració, i avisa l'usuari que comprovi el correu.
3. L'usuari prem l'enllaç i entra automàticament a la pàgina de restauració.
4. L'usuari escriu la nova contrasenya i prem el botó d'enviar.
5. El sistema emmagatzema la nova contrasenya.

Extensions:

2a. El sistema no reconeix el correu electrònic

2a1. El sistema informa l'usuari de l'error

2a2. Tornem al pas 1

3a. El token ha caducat

3a1. El sistema informa l'usuari de l'error

3a2. Tornem al pas 1

5a. Les contrasenyes no coincideixen

5a1. El sistema informa l'usuari de l'error

5b2. Tornem al pas 4

Cu04: Logout

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat

Garanties mínimes: cap

Garanties en cas d'èxit: el sistema no té cap usuari registrat.

Escenari principal d'èxit:

1. L'usuari surt del sistema.

Cu05: Crear una nova factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat

Garanties mínimes: cap

Garanties en cas d'èxit: el sistema té una nova factura associada a l'usuari registrat

Escenari principal d'èxit:

1. L'usuari entra les dades de la factura.
2. El sistema enregistra la factura amb les dades entrades.
3. L'usuari entra les línies de la factura.
4. El sistema enregistra les línies corresponents a la nova factura.

Cu06: Editar una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat, la factura existeix i no està emesa ni anul·lada

Garanties mínimes: cap

Garanties en cas d'èxit: existeix al sistema la mateixa factura amb les dades entrades.

Escenari principal d'èxit:

1. L'usuari modifica les dades de la factura.
2. El sistema enregistra la factura amb les dades modificades.

Cu07: Eliminar una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat, la factura existeix i no està emesa ni anul·lada

Garanties mínimes: cap

Garanties en cas d'èxit: no existeix al sistema la factura eliminada.

Escenari principal d'èxit:

1. L'usuari escull eliminar la factura.
2. El sistema elimina la factura.

Cu08: Emetre una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat, la factura existeix i no està emesa ni anul·lada

Garanties mínimes: cap

Garanties en cas d'èxit: existeix al sistema la factura amb les dades del client, i consta com a emesa.

Escenari principal d'èxit:

1. L'usuari escull emetre la factura.
2. El sistema emet la factura.

Cu09: Emetre una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat, la factura existeix i no està anul·lada

Garanties mínimes: cap

Garanties en cas d'èxit: existeix al sistema la factura i consta com a anul·lada.

Escenari principal d'èxit:

1. L'usuari escull anul·lar la factura.
2. El sistema anul·la la factura.

Cu10: Imprimir una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i la factura existeix.

Garanties mínimes: cap

Garanties en cas d'èxit: cap.

Escenari principal d'èxit:

1. L'usuari escull imprimir la factura.
2. El sistema mostra a l'usuari una vista prèvia d'impressió.

Cu11: Compartir una factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat, existeixen aplicacions al sistema que permetin compartir dades i la factura existeix.

Garanties mínimes: cap

Garanties en cas d'èxit: cap.

Escenari principal d'èxit:

1. L'usuari escull compartir la factura.
2. El sistema mostra a l'usuari les aplicacions amb les que es pot compartir.
3. El sistema envia a l'aplicació escollida un arxiu amb la factura en format pdf.

Cu12: Crear un client

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat.

Garanties mínimes:cap.

Garanties en cas d'èxit: existeix un nou client amb les dades entrades associat a l'usuari.

Escenari principal d'èxit:

1. L'usuari entra les dades del client.
2. El sistema enregistra el client amb les dades entrades.

Extensions:

2a. El NIF del client ja està registrat a un altre client del mateix usuari.

2a1. El sistema informa l'usuari de l'error

2a2. Tornem al pas 1

Cu13: Editar un client

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i té associat el client.

Garanties mínimes:cap.

Garanties en cas d'èxit: existeix el mateix client amb les dades modificades.

Escenari principal d'èxit:

1. L'usuari entra les dades del client.
2. El sistema enregistra el client amb les dades entrades.

Extensions:

2a. El NIF del client ja està registrat a un altre client del mateix usuari.

2a1. El sistema informa l'usuari de l'error

2a2. Tornem al pas 1

Cu14: Eliminar un client

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i té associat el client.

Garanties mínimes:cap.

Garanties en cas d'èxit: no existeix el client associat a l'usuari.

Escenari principal d'èxit:

1. L'usuari escull eliminar el client.
2. El sistema elimina el client.

Extensions:

2a. HI ha factures no emeses associades a aquest client.

2a1. El sistema informa l'usuari de l'error

2a2. Tornem al pas 1

Cu15: Crear una línia de factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i té una factura associada.

Garanties mínimes:cap.

Garanties en cas d'èxit: la factura té associada la línia de factura i les dades derivades actualitzades.

Escenari principal d'èxit:

1. L'usuari crea una línia de factura.
2. El sistema enregistra la línia de factura, l'associa a la factura i actualitza les dades derivades de la factura.

Cu16: Editar una línia de factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i té una factura associada amb línies de factura.

Garanties mínimes:cap.

Garanties en cas d'èxit: la factura té associada la línia de factura amb les dades actualitzades i les dades derivades de la factura estan actualitzades.

Escenari principal d'èxit:

1. L'usuari edita una línia de factura.
2. El sistema enregistra la línia de factura i actualitza les dades derivades de la factura.

Cu17: Eliminar una línia de factura

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: l'usuari està registrat i té una factura associada amb línies de factura.

Garanties mínimes:cap.

Garanties en cas d'èxit: la línia de factura no existeix al sistema i les dades derivades de la factura associada estan actualitzades.

Escenari principal d'èxit:

1. L'usuari escull eliminar una línia de factura.
2. El sistema elimina la línia de factura i actualitza les dades derivades de la factura.

Cu18: Editar perfil d'usuari

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: cap

Garanties mínimes: cap

Garanties en cas d'èxit: existeix al sistema el mateix usuari amb les dades modificades.

Escenari principal d'èxit:

1. L'usuari entra les dades que vol modificar.
2. El sistema registra l'usuari amb les dades entrades.

Extensions:

2a. El correu electrònic de l'usuari ja està registrat a un altre usuari

2a1. El sistema informa l'usuari de l'error

2a2. Tornem al pas 1

Cu19: Eliminar usuari

Actor principal: usuari

Àmbit: sistema

Nivell d'objectiu: usuari

Precondicions: cap

Garanties mínimes: cap

Garanties en cas d'èxit: ni l'usuari ni les seves dades associades existeixen al sistema.

Escenari principal d'èxit:

1. L'usuari escull eliminar-se.
2. El sistema elimina l'usuari amb totes les seves dades associades.

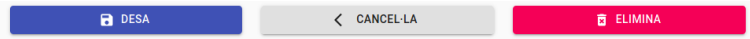
Consideracions sobre els casos d'ús

El cas d'ús **Cu11** (compartir una factura) no està previst que es pugui realitzar en el present treball.

13. Usabilitat/UX

Per obtenir una experiència d'usuari correcta, s'han aplicat els següents principis:

- **Consistència**: està reflectida en diversos aspectes de l'aplicació:
  - Els botons estan etiquetats de manera consistent en les diverses pantalles. Les icones que els acompanyen es mantenen per la mateixa acció en cada pantalla. Els colors també reflecteixen el tipus d'acció a realitzar, segons si és l'acció per defecte (per exemple, desar els canvis), una acció que no provoca canvis en les dades (per exemple, cancel·lar una modificació) o una acció que pot modificar les dades de forma no desitjada i irreversible (per exemple, eliminar un element o emetre una factura).



  - Els elements gràfics estan dissenyats de manera consistent en tota l'aplicació. Això afecta el posicionament de cadascun dels elements (menús, botons, representació de dades, avisos...), el color, el tipus i mida de la lletra...
- **Prevenció d'errors**: l'aplicació només permet introduir el tipus de dada que s'admet en cada cas. Per exemple, si introduïm un text en un camp numèric o bé un format de correu electrònic incorrecte, l'aplicació ens avisarà de l'error i de com cal corregir-lo. Com a altre exemple, també hi ha la funcionalitat de seleccionar el municipi i el codi postal d'una llista en comptes d'entrar-ho manualment. A més a més, les accions que no es poden realitzar perquè el context no ho permet, estan desactivades de forma visual.
- **Missatges d'error**: en cas d'error (normalment per inconsistència en l'entrada de dades), l'aplicació avisa l'usuari amb missatges clars i entenedors, i aplicats en l'element que ha produït l'error (per exemple, si la llargada de la contrasenya és massa curta es mostra l'error sota el camp d'entrada de contrasenya).

- **Indicacions de temps d'espera**: tot i que normalment els temps d'espera són molt curts, ja que el volum de dades que es transmet entre *forntend* i *backend* és de poc pes, es notifica aquest temps d'espera a l'usuari amb un *loader* de tipus modal.
- **Impressió correcta de les pàgines**: en el cas de les factures, la visualització prèvia de la factura impresa és fidel al resultat que se n'obté en el moment de la impressió en paper.
- **Navegació**: s'han seguit les següents pautes:
  - Només s'han utilitzat menús desplegables en casos molt puntuals. Per exemple, en pantalles petites (dispositius mòbils), el menú principal es desplega a partir d'un botó tipus *hamburguer*  , ja que s'ha considerat més important aprofitar la màxima àrea de pantalla per representar les dades (que en aquest tipus d'aplicació és important) que no pas mostrar sempre els elements del menú.

  - El menú és estàtic, presenta les mateixes opcions sigui quina sigui la pantalla on ens trobem.
  - L'usuari sap en tot moment en quin punt de l'aplicació es troba, ja que està indicat en la barra superior de l'aplicació.
  - La ruta de navegació representa el lloc on està actualment l'usuari de la manera més entenedora possible. Per exemple, si està modificant les dades d'un client amb NIF 12345678A, la ruta és https://easybill/clients/12345678A. O bé si està editant la factura amb número 20201206, la ruta és https://easybill/invoices/20201206.
  - El títol de l'aplicació explica clarament l'acció que s'està creant una factura, el títol mostrarà «Nova factura».
  - Totes les pàgines proporcionen opcions de navegació, normalment a partir del menú principal.
- **Enllaços**: Les etiquetes dels enllaços són descriptives, amb una longitud de text el màxim de breu perquè siguin comprensibles. Els noms corresponent clarament amb la pàgina de destinació. Tots els enllaços són textuals, i en alguns casos incorporen icones que ajuden a identificar millor l'acció que se'n deriva.
- **Disseny de la pàgina**: s'han tingut en compte diversos factors a l'hora de dissenyar la pàgina:
  - Evitar la sobreinformació, de forma que l'usuari no es distregui amb informació poc rellevant o innecessària.
  - Col·locació dels elements més importants a la part superior de la pàgina, com poden ser les llistes de factures, o les dades d'una factura en la pantalla d'edició.
  - Dintre de les pàgines, els elements estant col·locats amb una densitat correcta i alineats entre ells i els marges de la pàgina de forma agradable i consistent.

14. Seguretat

Les dades de facturació d'una empresa són un tema força sensible, fins i tot si estem dins el context d'una empresa petita o fins i tot d'un autònom. És per aquest motiu que cal ser curosos amb la seguretat de les dades que s'hi generen.

A nivell de base de dades, la seguretat dependrà de l'instal·lador. L'aplicació treballa amb un gestor de bases de dades MongoDB de forma remota. Per tant, serà important que es defineixin unes claus d'accés prou fortes.

A nivell d'aplicació, hi ha varis factors a tenir en compte:

- L'aplicació està dissenyada perquè es necessiti un JWT per accedir a totes les dades personals. En aquest sentit, l'aplicació es pot considerar segura.
- Les crides al backend es fan a través d'API REST des del frontend. És molt convenient instal·lar-lo en un servidor amb protocol https, de tal forma que el pas de dades entre client i servidor es faci de forma encriptada.
- L'aplicació no valida les dades d'accés d'usuari en funció de la seguretat, simplement demana que el nom d'usuari sigui un correu electrònic sintàcticament vàlid i que la contrasenya tingui almenys 6 caràcters. Serà el propi usuari qui determini la seguretat d'accés al seu compte (i per tant, a les seves dades) a través d'una contrasenya més o menys forta.

15. Tests

S’han programat tests unitaris automàtics per algunes de les llibreries programades per l’aplicació:

- filters.test.ts: relacionada amb el filtratge d’elements a partir d’una cadena de text
- invoices.test.ts: relacionada amb el tractament de les factures

16. Requisits d’instal·lació

L’aplicació es pot instal·lar en qualsevol sistema que requereixi les següents característiques:

- Sistema operatiu Linux, Windows o macOS que permeti instal·lar NodeJS versió 13 (mínim).
- NodeJS versió 13 (mínim).
- Gestor de paquets (npm, yarn, brew...).
- Connexió a internet (la base de dades és remota).

A part, cal tenir un compte de MongoDB.

17. Instruccions d’instal·lació

Per instal·lar l’aplicació, cal:

- Descomprimir en una carpeta els fitxers **client.zip, server.zip** i **postalcodes.zip**.
- Dins de la carpeta client, instal·lar els paquets (per exemple, `npm i`).
- Dins de la carpeta server, instal·lar els paquets (per exemple, `npm i`).
- Dins de la carpeta postalcodes, instal·lar els paquets (per exemple, `npm i`).
- A la carpeta server, crear les variables d’entorn **mongoURI** (amb les dades de connexió proporcionades per MongoDB), **PORT** (amb el número de port), i les referents a l’enviament de correus electrònics (**MAILER_HOST**, **MAILER_PORT**, **MAILER_USER** i **MAILER_PASSWD**). Per exemple:
  - mongoURI="mongodb+srv://pepitu:oK4UfkP48U10PlM4@cluster0.ewwwx.mongodb.net/easybill?retryWrites=true&w=majority"
  - PORT=5000
  - MAILER_HOST=smtp.mailtrap.io
  - MAILER_PORT=2525
  - MAILER_USER=[user]
  - MAILER_PASSWD=[passwd]
- A la carpeta postalcodes, crear les variables d’entorn **mongoURI** (amb les dades de connexió proporcionades per MongoDB), i **PORT** (amb el número de port). Per exemple:
  - mongoURI="mongodb+srv://pepitu:oK4UfkP48U10PlM4@cluster0.ewwwx.mongodb.net/postalcodes?retryWrites=true&w=majority"
  - PORT=5100
- A la carpeta client, crear les variables d’entorn **REACT_APP_API_URL** (amb l’adreça del servei API del backend), **REACT_APP_API_CITIES_URL** (amb l’adreça del servei API dels municipis), **REACT_APP_HOMEPAGE** (amb el subdirectori d’instal·lació) i **REACT_APP_LANGUAGE** (amb el codi d’idioma, només disponibles ca i en). Per exemple:
  - REACT_APP_API_URL="http://localhost:5000/api/"
  - REACT_APP_API_CITIES_URL="http://localhost:5100/api/"
  - REACT_APP_HOMEPAGE="/"
  - REACT_APP_LANGUAGE=ca
- Des de la carpeta server, engegar el servidor amb la comanda `npm run server`. Si tot va bé, es mostraran el missatges Server running on port 5000 i MongoDB connected
- Des de la carpeta postalcodes, engegar el servidor amb la comanda `npm run server`. Si tot va bé, es mostraran el missatges Server running on port 5100 i MongoDB connected
- Des de la carpeta client, engegar el frontend amb la comanda `npm start`, S’obrirà automàticament una finestra al navegador amb l’aplicació.

18. Instruccions d’ús

L’aplicació està instal·lada per qüestions de testing a http://carlescanellas.cat/easybill. Tots els serveis de backend estan engegats, per tant està a punt per ser utilitzada.

En cas que haguem instal·lat l’aplicació nosaltres mateixos, primer de tot cal engegar els diferents serveis a l’ordinador client perquè pugui funcionar tot el sistema:

- Des de la carpeta *postalcodes*, teclejar `npm run server`
- Des de la carpeta *server*, teclejar `npm run server`
- Des de la carpeta *client* teclejar `npm start`. Si no s’obrís automàticament una pestanya del navegador amb la pàgina inicial, obrir-lo manualment a l’adreça http://localhost:3000. Alternativament podem copiar el contingut de la carpeta *build* en un servidor i accedir directament a l’aplicació directament, sense engegar el servei manualment.

Per començar a utilitzar l’aplicació només cal registrar-se com a nou usuari, i posteriorment entrar al sistema amb les dades de registre (correu electrònic i contrasenya). Els enllaços de registre i d’entrada es troben a la pàgina d’inici.

En cas d’estar registrat i haver oblidat la contrasenya, des de la pàgina d’entrada cal prémer l’enllaç *No recordo la contrasenya*. El sistema envia un correu a l’adreça introduïda amb un token i un enllaç per poder restaurar la contrasenya d’usuari.

Un cop dins de l’aplicació, cal que creem almenys un client, ja que sinó no podrem crear cap factura. Entrem les dades del client (les necessàries per fer una factura són requerides) i el desem. Posteriorment podem crear una factura des de la llista de factures.

Un cop fetes les factures, les podem editar, emetre, anul·lar, eliminar i imprimir. A les factures emeses no se’ls poden canviar les dades. En canvi, sí que es pot modificar si la factura ha estat cobrada o si s’han pagat els seus impostos, i també es poden anul·lar. Les factures anul·lades només es poden visualitzar, però no permeten cap tipus d’edició.

Les dades personals es poden modificar. En canvi, no es permet l’eliminació de l’usuari.

19. Projecció a futur

Easybill és, en el moment de lliurar el projecte, una aplicació pensada per posar en pràctica els coneixements apresos durant el màster de llocs i aplicacions web. Les funcionalitats que ofereix es poden trobar en altres aplicacions que existeixen al mercat, ja siguin lliures o de pagament.

Tenint això en compte, és cert que l'aplicació té una bona base per poder ésser ampliada, per tal que pugui competir amb altres aplicacions enfocades a facilitat la facturació de petits empresaris. Alguns punts que es podrien implementar són:

- Creació de factures electròniques: actualment hi ha moltes administracions públiques (sobretot ajuntaments) que demanen les factures en aquest format.
- Importació i exportació de les factures en altres formats (com a mínim, CSV).
- Eines de gestió de taxes, sobretot pagaments trimestrals d'IVA.
- Relacionat amb el punt anterior, gestió de despeses.
- Eines estadístiques: gràfics d'evolució de facturació, de número de factures...

Són només algunes de les funcionalitats que es podrien implementar per tal de fer una aplicació competitiva a nivell comercial.

20. Decisions de programació

Destaco aquí algunes de les decisions rellevants en el moment de la programació.

20.1 Frontend

El frontend és l'aplicació més complexa, tant a nivell de lògica com, sobretot, d'interfície d'usuari.

Estructura de les carpetes

Per entendre el codi és important saber com s'ha estructurat a nivell de carpetes. El conveni que s'ha seguit és:

- El codi generat és dins la carpeta *src*.
- La carpeta *pages* conté els components contenidors de cada pàgina.
- La carpeta *components* conté els components interns de cada pàgina, agrupats per carpetes segons la pàgina on són utilitzats. També hi ha una carpeta *common* pels components utilitzats a diverses pàgines (per exemple, la capçalera que surt a totes les pàgines).
- La carpeta *languages* conté traduccions de l'aplicació. Per aquesta versió només s'han traduït els missatges d'error. L'idioma no es pot escollir des del menú d'usuari sinó que cal configurar-lo com a variable d'entorn a l'hora de compilar-lo.
- La carpeta *lib* conté llibreries de funcions agrupades per tipus de funcionalitat.
- A la carpeta *classes* hi ha les definicions d'objectes creades per aquesta aplicació. Només hi ha classes de tipus d'error.
- La carpeta *api* conté totes les crides que es fan all backend.
- A la carpeta *constants* hi ha els arxius amb les constants utilitzades al programa.
- A la carpeta *redux* hi ha les *actions*, *stores* i *reducers* de Redux.

Estructura dels components

Depenent del tipus de pàgina on ens trobem, els components estan estructurats segons un patró:

- Els components de pàgina contenen la funcionalitat i els subcomponents que la formen.
- En les pàgines de llista (clients, factures), hi ha un component de filtratge i un amb la llista pròpiament dita.
- En les pàgines amb formulari (CRUD client, CRUD factura, perfil d'usuari...), el formulari és un component a part. En el cas de les factures, el formulari inclou altres components, com són els botons, llista de línies de factura, resum de la factura...

20.2 Backend

Estructura de les carpetes

- *config*: arxius de configuració de l'accés a base de dades i seguretat.
- *constants:* constants referents als missatges d'error.

- *db*: arxius de connexió a base de dades.
- *languages*: arxius d'idioma pels missatges d'error.
- *lib*: llibreries programades per aquesta aplicació.
- *models*: models de dades que segueixen l'esquema determinat per Mongoose per fer un mapatge directe amb la base de dades.
- *routes*: lògica executada segons l'endpoint utilitzat.
- *validation*: arxius de validació de les dades rebudes a través de HTTP.

Esquema funcional:

L'arxiu d'entrada de l'aplicació (*index.js*) realitza les següents accions:

- Configura la seguretat
- Configura l'accés a la base de dades
- Registra els diversos punts d'entrada
- Engega el servidor

21. Conclusions

Easybill és un projecte que m'ha permès, sobretot, créixer a nivell professional i de coneixements.

Alguns dels objectius que m'havia proposat en el moment de plantejar el projecte els considero plenament assolits (si és que en el món de la informàtica es pot assolir plenament algun objectiu).

Alguns d'aquests objectius són:

- Introducció en els serveis API programats sobre la base de NodeJS.
- Aprofundiment en les bases de dades NoSQL. Particularment, introducció en MongoDB (anteriorment ja havia utilitzat Firebase i RethinkDB per altres projectes).
- Aprofundiment en la programació d'aplicacions frontend basades en React i les seves principals llibreries associades, com Redux.
- Utilització en React estrictament de components funcionals. Ús de hooks per controlar l'estat de l'aplicació.
- Aprofundiment en l'ús de material-ui per crear interfícies d'usuari que segueixin els paràmetres de disseny especificats per Material Design, tant en dispositius mòbils com d'escriptori o tauleta.
- Millora del procés d'anàlisi, disseny i desenvolupament d'aplicacions web.
- Introducció als tests unitaris en javascript, i concretament en React.
- Introducció en el desenvolupament d'aplicacions web progressives (PWA).

Evidentment, és molt el camp que em queda per recórrer en el desenvolupament d'aplicacions web. Segurament els punts on he d'incidir més són:

- Involucrar l'usuari final en el procés de creació, ja sigui en el moment de l'anàlisi, disseny, testing...
- Procés de màrqueting.
- Programació més basada en el paradigma TDD (crear primer els tests en base els casos d'ús). Era un objectiu que tenia inicialment però que no he acabat d'aconseguir. Crec que segueix essent interessant i cal que m'ho plantegi en propers projectes.

En definitiva, estic força satisfet del resultat del projecte i de l'evolució que ha suposat en els meus coneixements. També pels reptes de futur que m'ha generat com a desenvolupador d'aplicacions web.

Annex 1. Lliurables del projecte

Els continguts inclosos en el lliurament del projecte inclouen:

- PAC_FINAL_mem_Canellas_Crusellas_Carles: és el document actual, la memòria del treball.
- client.zip: arxiu amb el codi font del frontend.
- server.zip: arxiu amb el codi font del backend.
- postalcodes.zip: arxiu amb el codi font del servidor de municipis.
- PAC_FINAL_prs_Canellas_Crusellas_Carles: presentació del projecte en diapositives.

A part, es lliuren els següents vídeos mitjançant l'eina Present@:

- PAC_FINAL_vid_Canellas_Crusellas_Carles.mp4: vídeo de presentació i defensa de l'aplicació.

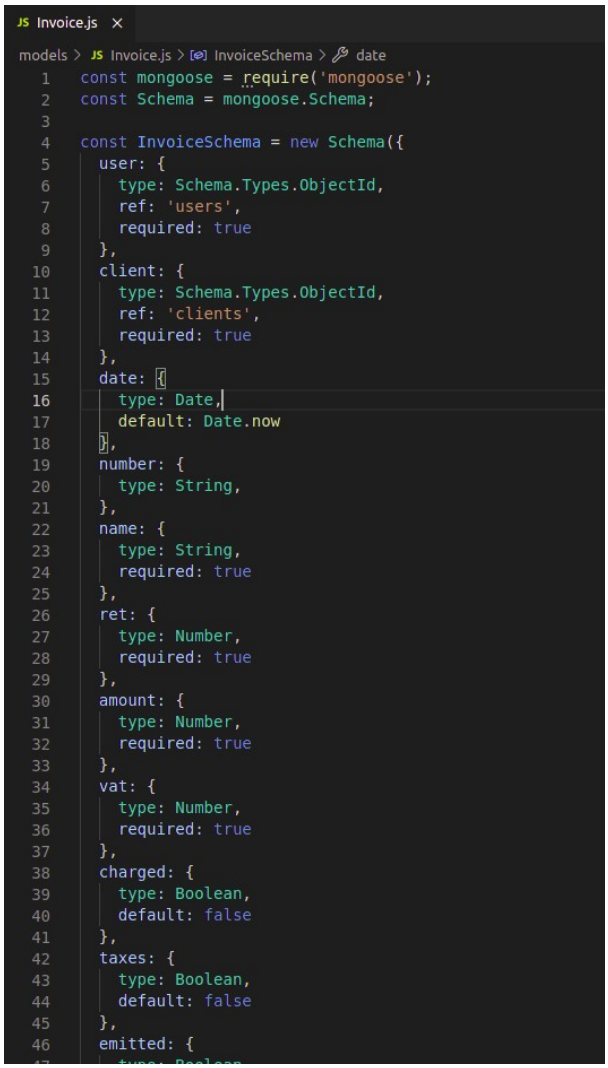
Annex 2. Codi font (extractes)

server

index.js

```
const express = require('express')
const bodyParser = require('body-parser')
const cors = require('cors')
const passport = require('passport');
const mongoose = require('mongoose');

const dotenv = require('dotenv');
dotenv.config();

const db = process.env.mongoURI;

const auth = require('./routes/api/auth');
const users = require('./routes/api/users');
const clients = require('./routes/api/clients');
const cities = require('./routes/api/cities');
const invoices = require('./routes/api/invoices');
const lines = require('./routes/api/lines');

const app = express()
const apiPort = process.env.PORT || 5000;

app.use(bodyParser.urlencoded({ extended: true }))
app.use(cors())
app.use(bodyParser.json())

// connect to MongoDB
// db.on('error', console.error.bind(console, 'MongoDB connection error:'))
mongoose.connect(db, { useNewUrlParser: true, useUnifiedTopology: true })
  .then(() => console.log('MongoDB connected'))
  .catch(err => console.log(err));

// passport middleware
app.use(passport.initialize());
require('./config/passport')(passport);

// Routes
app.use('/api/auth', auth);
app.use('/api/users', users);
app.use('/api/clients', clients);
app.use('/api/cities', cities);
app.use('/api/invoices', invoices);
app.use('/api/lines', lines);

app.listen(apiPort, () => console.log(`Server running on port ${apiPort}`))
```

Inici del servidor API, amb l'ús destacat de les llibreries Express (framework per crear l'API), Passport (seguretat a través de JWT) i Mongoose (connexió amb MongoDB).



Part del model de factura, on es pot veure la dependència amb els models User i Client.

```
routes > api > auth.js > router.post('/login') callback
56  });
57
58  // @route   POST api/auth/login
59  // @desc    Login user. Return JWT
60  // @access  Public
61  router.post('/login', async (req, res) => {
62    const { errors, isValid } = validateLoginInput(req.body);
63
64    if (!isValid) {
65      return res.status(400).json(errors);
66    }
67
68    const { email, password } = req.body;
69
70    try {
71      const user = await User.findOne({ email })
72
73      if (!user) {
74        errors.push(setError(INVALID_CREDENTIALS));
75        return res.status(400).json(errors);
76      }
77
78      const isMatch = await bcrypt.compare(password, user.password);
79
80      if (!isMatch) {
81        errors.push(setError(INVALID_CREDENTIALS));
82        return res.status(400).json(errors);
83      }
84
85      const payload = { id: user.id, name: user.name, surnames: user.surnames };
86
87      jwt.sign(payload, keys.secretOrKey, { expiresIn: 86400 }, (err, token) => {
88        if (err) throw err;
89        res.send(token);
90      });
91    } catch (error) {
92      console.log(error);
93      return res.status(500).json(setError(SERVER_ERROR));
94    }
95  });
96
97  // @route   POST api/auth/sendjwt
98  // @desc    Send email to restore password
99  // @access  Public
100 router.post('/sendjwt', async (req, res) => {
101   try {
```

Extracte de l’endpoint d’autorització. Es pot veure la comprovació de la contrasenya encriptada i la creació i retorn del JWT.

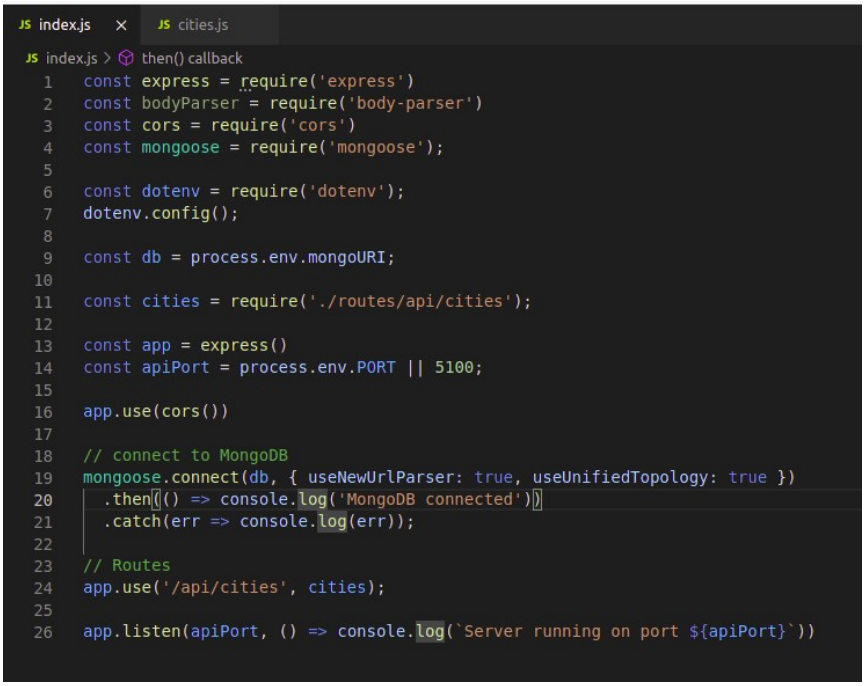
client.js

validation > client.js > <unknown> > validateClientInput

```
8
9   data.name = !isEmpty(data.name) ? data.name : '';
10  data.surnames = !isEmpty(data.surnames) ? data.surnames : '';
11  data.email = !isEmpty(data.email) ? data.email : '';
12  data.commercialName = !isEmpty(data.commercialName) ? data.commercialName : '';
13  data.phone = !isEmpty(data.phone) ? data.phone : '';
14  data.nif = !isEmpty(data.nif) ? data.nif : '';
15  data.address = !isEmpty(data.address) ? data.address : '';
16  data.city = !isEmpty(data.city) ? data.city : '';
17  data.postalcode = !isEmpty(data.postalcode) ? data.postalcode : '';
18  data.acoount = !isEmpty(data.acoount) ? data.acoount : '';
19
20  if (Validator.isEmpty(data.name) && Validator.isEmpty(data.commercialName)) {
21    errors.push(setError(COMMERCIAL_NAME_REQUIRED));
22  }
23
24  if (!Validator.isEmpty(data.email) && !Validator.isEmail(data.email)) {
25    errors.push(setError(INVALID_EMAIL));
26  }
27
28  if (Validator.isEmpty(data.nif)) {
29    errors.push(setError(NIF_REQUIRED));
30  }
31
32  if (Validator.isEmpty(data.address)) {
33    errors.push(setError(ADDRESS_REQUIRED));
34  }
35
36  if (Validator.isEmpty(data.city)) {
37    errors.push(setError(CITY_REQUIRED));
38  }
39
40  if (!Validator.isLength(data.postalcode, { min: 5, max: 5 })) {
41    errors.push(setError(POSTAL_CODE_LENGTH));
42  }
43
44  if (Validator.isEmpty(data.postalcode)) {
45    errors.push(setError(POSTAL_CODE_REQUIRED));
46  }
47
48  return {
49    errors,
50    isValid: isEmpty(errors)
51  }
52 }
```

Extracte del validador de dades d'entrada d'un client. Es poden veure, entre d'altres, les comprovacions de format de correu electrònic, la llargada del codi postal o els diferents camps requerits.

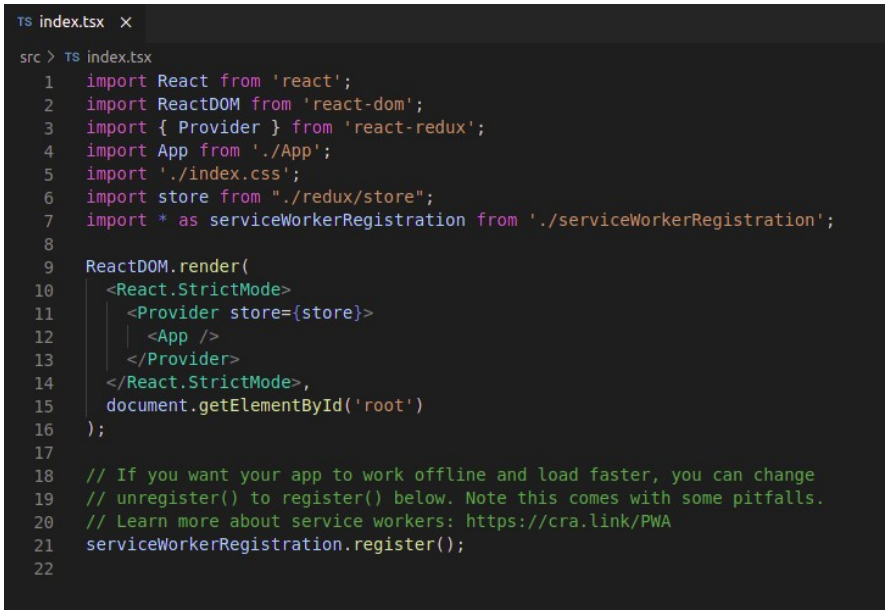
postalcodes



```
index.js  cities.js
index.js > then() callback
1   const express = require('express')
2   const bodyParser = require('body-parser')
3   const cors = require('cors')
4   const mongoose = require('mongoose');
5
6   const dotenv = require('dotenv');
7   dotenv.config();
8
9   const db = process.env.mongoURI;
10
11  const cities = require('./routes/api/cities');
12
13  const app = express()
14  const apiPort = process.env.PORT || 5100;
15
16  app.use(cors())
17
18  // connect to MongoDB
19  mongoose.connect(db, { useNewUrlParser: true, useUnifiedTopology: true })
20    .then(() => console.log('MongoDB connected'))
21    .catch(err => console.log(err));
22
23  // Routes
24  app.use('/api/cities', cities);
25
26  app.listen(apiPort, () => console.log(`Server running on port ${apiPort}`))
```

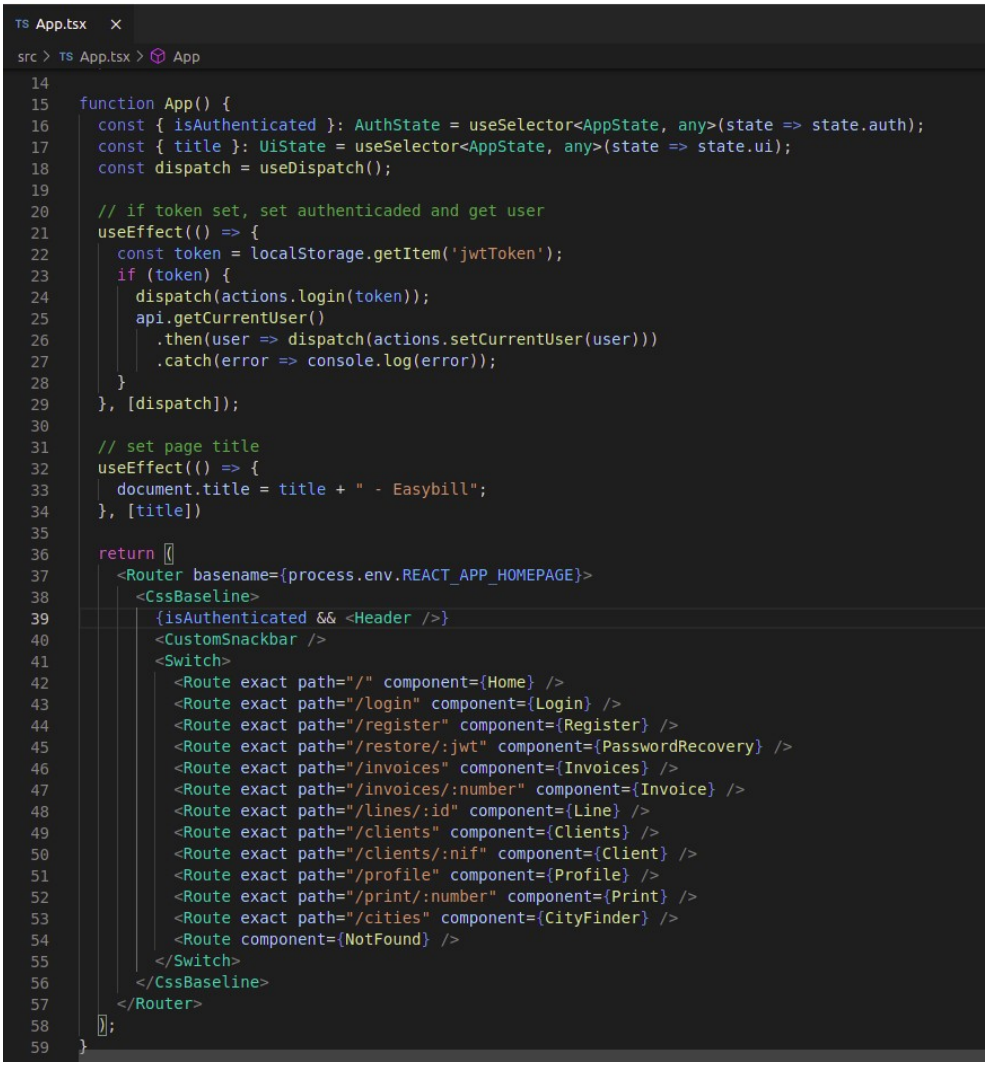
Arxiu inicial de l'API de codis postals i municipis. Tornem a veure l'ús de les llibreries Express i Mongoose.

client



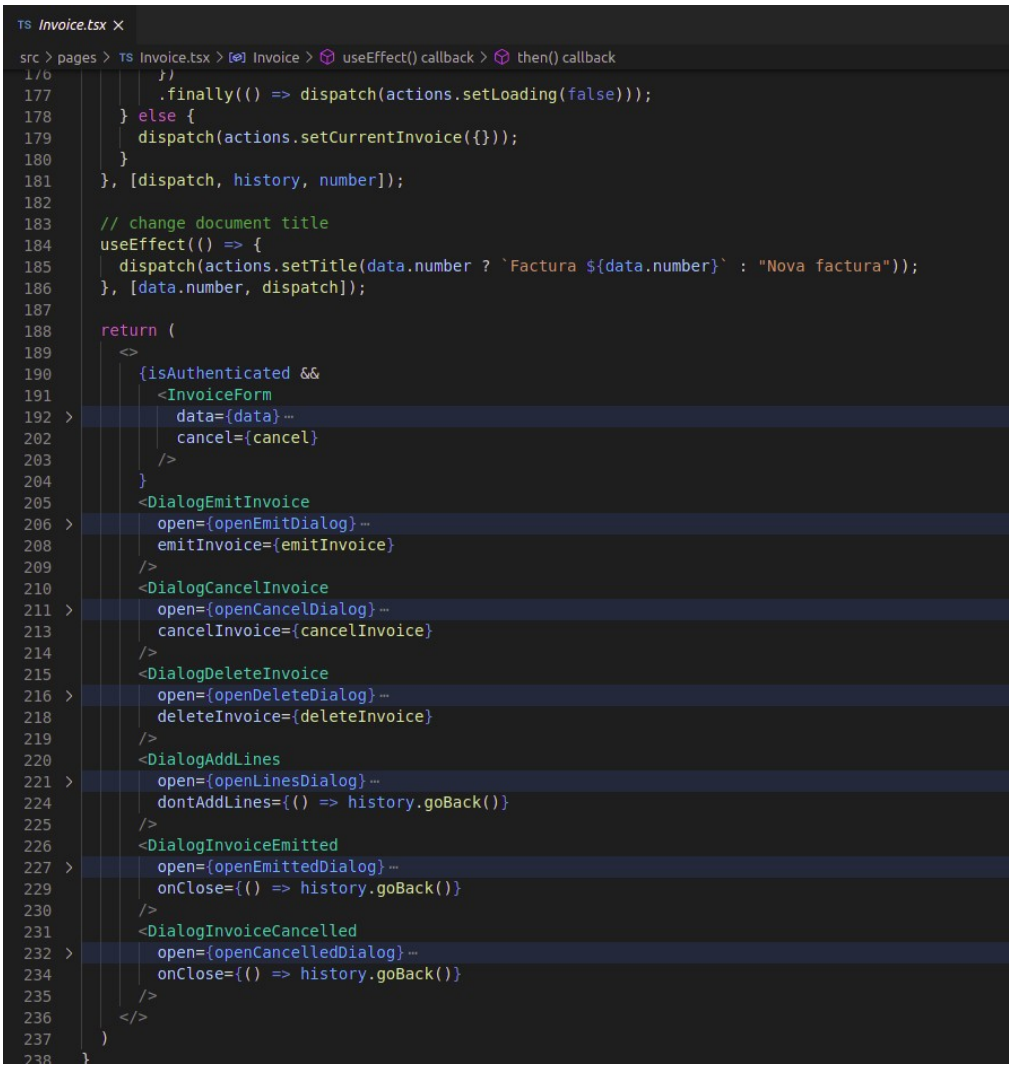
```
index.tsx
src > index.tsx
1   import React from 'react';
2   import ReactDOM from 'react-dom';
3   import { Provider } from 'react-redux';
4   import App from './App';
5   import './index.css';
6   import store from "./redux/store";
7   import * as serviceWorkerRegistration from './serviceWorkerRegistration';
8
9   ReactDOM.render(
10    <React.StrictMode>
11      <Provider store={store}>
12        <App />
13      </Provider>
14    </React.StrictMode>,
15    document.getElementById('root')
16  );
17
18  // If you want your app to work offline and load faster, you can change
19  // unregister() to register() below. Note this comes with some pitfalls.
20  // Learn more about service workers: https://cra.link/PWA
21  serviceWorkerRegistration.register();
22
```

Inicialització de l'aplicació, on es pot veure la integració de Redux i el registre del service worker.



```tsx
// src/App.tsx
function App() {
  const { isAuthenticated }: AuthState = useSelector<AppState, any>(state => state.auth);
  const { title }: UiState = useSelector<AppState, any>(state => state.ui);
  const dispatch = useDispatch();

  // if token set, set authenticaded and get user
  useEffect(() => {
    const token = localStorage.getItem('jwtToken');
    if (token) {
      dispatch(actions.login(token));
      api.getCurrentUser()
        .then(user => dispatch(actions.setCurrentUser(user)))
        .catch(error => console.log(error));
    }
  }, [dispatch]);

  // set page title
  useEffect(() => {
    document.title = title + " - Easybill";
  }, [title])

  return (
    <Router basename={process.env.REACT_APP_HOMEPAGE}>
      <CssBaseline>
        {isAuthenticated && <Header />}
        <CustomSnackbar />
        <Switch>
          <Route exact path="/" component={Home} />
          <Route exact path="/login" component={Login} />
          <Route exact path="/register" component={Register} />
          <Route exact path="/restore/:jwt" component={PasswordRecovery} />
          <Route exact path="/invoices" component={Invoices} />
          <Route exact path="/invoices/:number" component={Invoice} />
          <Route exact path="/lines/:id" component={Line} />
          <Route exact path="/clients" component={Clients} />
          <Route exact path="/clients/:nif" component={Client} />
          <Route exact path="/profile" component={Profile} />
          <Route exact path="/print/:number" component={Print} />
          <Route exact path="/cities" component={CityFinder} />
          <Route component={NotFound} />
        </Switch>
      </CssBaseline>
    </Router>
  );
}
```

Punt d’entrada de l’aplicació. Configuració de les diferents rutes. Entrada al sistema sense login en cas d’existir un JWT a local storage.



```
TS Invoice.tsx ×
src > pages > TS Invoice.tsx > Invoice > useEffect() callback > then() callback
176           })
177           .finally(() => dispatch(actions.setLoading(false)));
178       } else {
179         dispatch(actions.setCurrentInvoice({}));
180       }
181     }, [dispatch, history, number]);
182
183     // change document title
184     useEffect(() => {
185       dispatch(actions.setTitle(data.number ? `Factura ${data.number}` : "Nova factura"));
186     }, [data.number, dispatch]);
187
188     return (
189       <>
190         {isAuthenticated &&
191           <InvoiceForm
192 >           data={data} …
202             cancel={cancel}
203           />
204         }
205         <DialogEmitInvoice
206 >         open={openEmitDialog} …
208           emitInvoice={emitInvoice}
209         />
210         <DialogCancelInvoice
211 >         open={openCancelDialog} …
213           cancelInvoice={cancelInvoice}
214         />
215         <DialogDeleteInvoice
216 >         open={openDeleteDialog} …
218           deleteInvoice={deleteInvoice}
219         />
220         <DialogAddLines
221 >         open={openLinesDialog} …
224           dontAddLines={() => history.goBack()}
225         />
226         <DialogInvoiceEmitted
227 >         open={openEmittedDialog} …
229           onClose={() => history.goBack()}
230         />
231         <DialogInvoiceCancelled
232 >         open={openCancelledDialog} …
234           onClose={() => history.goBack()}
235         />
236       </>
237     )
238   }
```

Pàgina de creació i edició de factura, amb la llista de components: la part de formulari i els diferents diàlegs d'interacció amb l'usuari.

Pàgina de creació i edició de factura. Funció submit que s'executa per modificar les dades de la factura o crear-ne una de nova. Es pot veure tot el procés: netejar els missatges d'error, notificar que s'està duent una operació remota, escollir l'endpoint segons si es crea o s'edita una factura, desar les dades a Redux, obrir el diàleg per afegir noves línies de comanda en cas de nova factura, tornar a la llista de factures en cas d'editar una factura. En cas d'error, si és de xarxa notificar a l'usuari que no està connectat; altrament, activar els missatges d'error. En qualsevol cas, notificar que l'operació remota ha finalitzat.

```tsx
// Invoices.tsx
// src > pages > Invoices.tsx > ...

  return (
    <>
      {isAuthenticated &&
        <Container maxWidth={false}>
          <Grid container justify="center" spacing={3}>

            <Grid item xs={12}>
              <InvoicesListFilter
                list={invoices}
                setFiltered={setFiltered}
              />
            </Grid>

            <Grid item xs={12}>
              <InvoicesList
                invoices={filtered}
              />
            </Grid>

            <AddInvoiceButton addInvoice={addInvoice} />

          </Grid>

        </Container>
      }
    </>
  )
}

export default Invoices
```

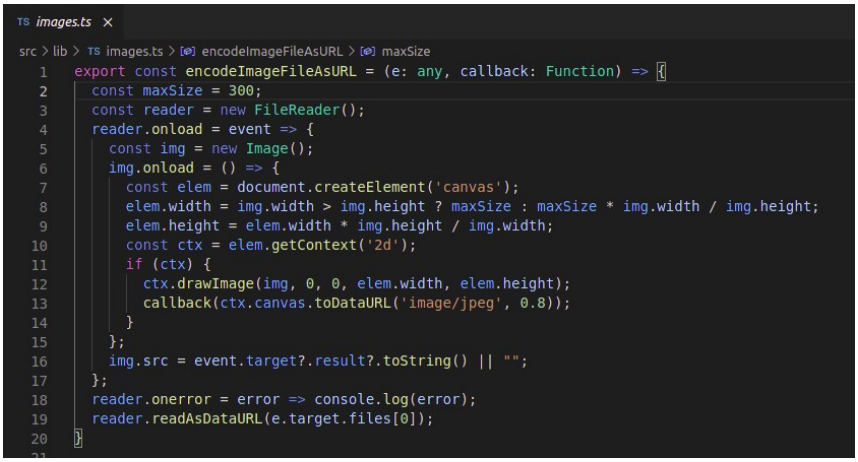
Pàgina de llista de factures. Components que la conformen: filtre de factures llistades, llista de factures i botó d'afegir factura, que es visualitza diferent segons la mida de la pantalla.

```ts
// invoices.ts
// src > api > invoices.ts > ...
import { getItemList, getItem, addItem, updateItem, deleteItem } from "./common";

export const getInvoices = async (): Promise<any> => {
  return getItemList('invoices');
}

export const getInvoice = async (invoiceNumber: string): Promise<any> => {
  return getItem('invoices', invoiceNumber);
}

export const addInvoice = async (invoiceData: any): Promise<any> => {
  return addItem('invoices', invoiceData);
}

export const updateInvoice = async (invoiceData: any): Promise<any> => {
  return updateItem('invoices', invoiceData);
}

export const deleteInvoice = async (invoiceId: string): Promise<any> => {
  return deleteItem('invoices', invoiceId);
}

export const emitInvoice = async (invoiceId: string): Promise<any> => {
  return updateItem('invoices/emit', { _id: invoiceId });
}

export const cancelInvoice = async (invoiceId: string): Promise<any> => {
  return updateItem('invoices/cancel', { _id: invoiceId });
}

export const getEmittedInvoice = async (invoiceNumber: string): Promise<any> => {
  return getItemList('invoices/emitted/' + invoiceNumber);
}
```

Comunicacions API de les factures. Per cada cas es criden funcions més genèriques definides en una altra llibreria.

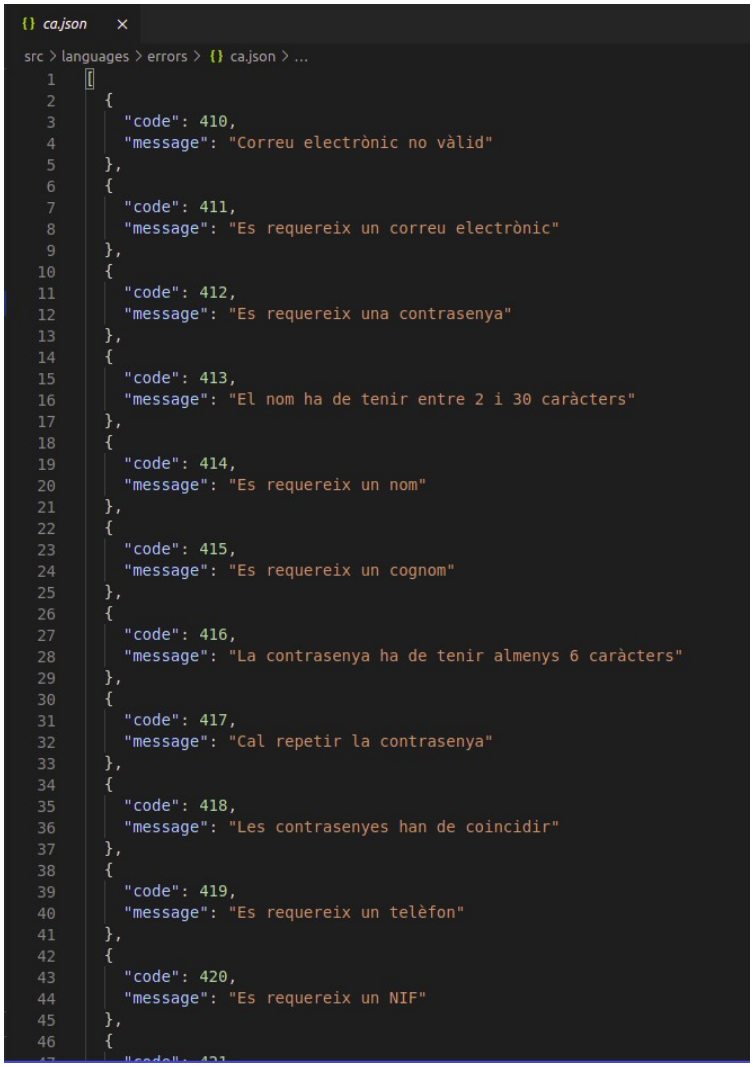
```ts
// src > api > common.ts > checkResponse
export const getItemList = async (endpoint: string): Promise<any> => {
  checkConnection();
  const token = getToken();
  const response = await fetch(url + endpoint, {
    headers: {
      'Authorization': `${token_prefix} ${token}`,
      'Content-Type': 'application/json',
    }
  });
  return checkResponse(response, 200);
}

export const getItem = async (endpoint: string, id: string): Promise<any> => {
  checkConnection();
  const token = getToken();
  const response = await fetch(`${url}${endpoint}/${id}`, {
    headers: {
      'Authorization': `${token_prefix} ${token}`,
      'Content-Type': 'application/json',
    }
  });
  return checkResponse(response, 200);
}

export const addItem = async (endpoint: string, data: any): Promise<any> => {
  checkConnection();
  const token = getToken();
  const response = await fetch(url + endpoint, {
    method: "POST",
    headers: {
      'Authorization': `${token_prefix} ${token}`,
      'Content-Type': 'application/json',
    },
    body: JSON.stringify(data)
  });
  return checkResponse(response, 201);
}

export const updateItem = async (endpoint: string, data: any): Promise<any> => {
  checkConnection();
  const token = getToken();
  const response = await fetch(url + endpoint + '/' + data._id, {
    method: "PUT",
    headers: {
      'Authorization': `${token_prefix} ${token}`,
      'Content-Type': 'application/json',
```

Funcions genèriques de comunicació amb l'API del servidor. Comprovació d'errors de connexió, enviament de dades i comprovació d'errors de resposta.



```ts
// src > lib > images.ts > encodeImageFileAsURL > maxSize
export const encodeImageFileAsURL = (e: any, callback: Function) => {
  const maxSize = 300;
  const reader = new FileReader();
  reader.onload = event => {
    const img = new Image();
    img.onload = () => {
      const elem = document.createElement('canvas');
      elem.width = img.width > img.height ? maxSize : maxSize * img.width / img.height;
      elem.height = elem.width * img.height / img.width;
      const ctx = elem.getContext('2d');
      if (ctx) {
        ctx.drawImage(img, 0, 0, elem.width, elem.height);
        callback(ctx.canvas.toDataURL('image/jpeg', 0.8));
      }
    };
    img.src = event.target?.result?.toString() || "";
  };
  reader.onerror = error => console.log(error);
  reader.readAsDataURL(e.target.files[0]);
}
```

Llibreria que converteix un arxiu d'imatge en un tipus base64 d'unes mides màximes.



{} ca.json

src > languages > errors > {} ca.json > ...

```
[
  {
    "code": 410,
    "message": "Correu electrònic no vàlid"
  },
  {
    "code": 411,
    "message": "Es requereix un correu electrònic"
  },
  {
    "code": 412,
    "message": "Es requereix una contrasenya"
  },
  {
    "code": 413,
    "message": "El nom ha de tenir entre 2 i 30 caràcters"
  },
  {
    "code": 414,
    "message": "Es requereix un nom"
  },
  {
    "code": 415,
    "message": "Es requereix un cognom"
  },
  {
    "code": 416,
    "message": "La contrasenya ha de tenir almenys 6 caràcters"
  },
  {
    "code": 417,
    "message": "Cal repetir la contrasenya"
  },
  {
    "code": 418,
    "message": "Les contrasenyes han de coincidir"
  },
  {
    "code": 419,
    "message": "Es requereix un telèfon"
  },
  {
    "code": 420,
    "message": "Es requereix un NIF"
  },
  {
```

Arxiu tipus JSON amb un array d'objectes que associen un número de codi d'error amb el seu corresponent missatge.

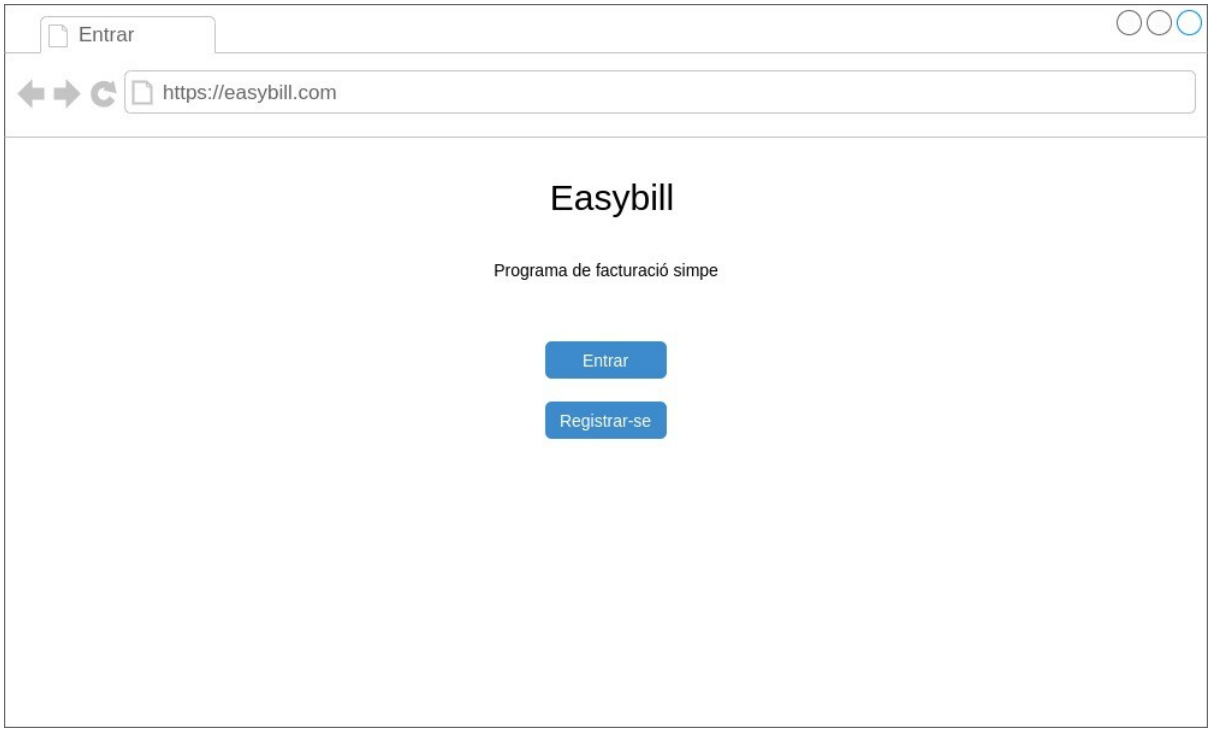
Annex 3. Llibreries externes utilitzades

- **Express**: framework per Node que proveeix eines HTTP per Node, fent-lo molt adequat per crear API de forma senzilla i robusta https://expressjs.com/
- **Mongoose**: eines de connexió i modelatge d'objectes per MongoDB. https://mongoosejs.com/
- **Nodemailer**: mòdul per Node que facilita l'enviament de correus electrònics. https://nodemailer.com/
- **Passport**: middleware per Node per gestionar l'autenticació. http://www.passportjs.org/
- **Bcrypt**: llibreria que facilita el hashing de contrasenyes. https://github.com/kelektiv/node.bcrypt.js
- **Dotenv**: mòdul per Node que carrega variables d'entorn des d'un arxiu .env. https://github.com/motdotla/dotenv
- **Jsonwebtoken**: llibreria per gestionar Json Web Tokens. https://github.com/auth0/node-jsonwebtoken
- **Body-parser**: middleware per Node que analitza el body dels requests HTTP. https://github.com/expressjs/body-parser
- **Cors**: middleware que habilita CORS amb diferents opcions. https://github.com/expressjs/cors
- **Validator**: valida cadenes de text de diferents tipus. https://github.com/validatorjs/validator.js
- **Material-ui**: components per React que segueixen les pautes d'experiència d'usuari de material design. https://material-ui.com/
- **Redux**: gestió de l'estat de l'aplicació. https://redux.js.org/
- **Redux-devtools-extension**: extensió per poder fer el seguiment de l'estat des d'un navegador on s'executa l'aplicació. https://github.com/zalmoxisus/redux-devtools-extension

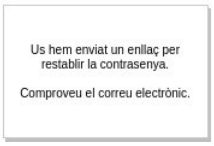
Annex 4. Captures de pantalla

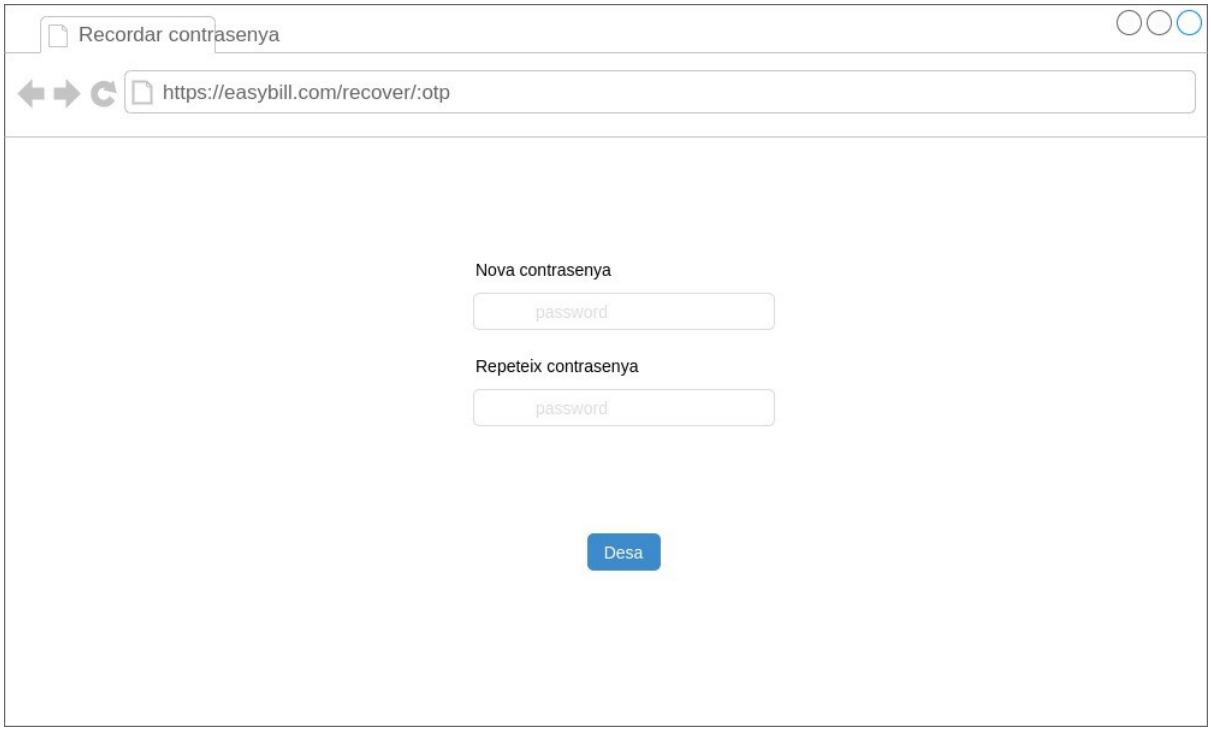
Wireframes

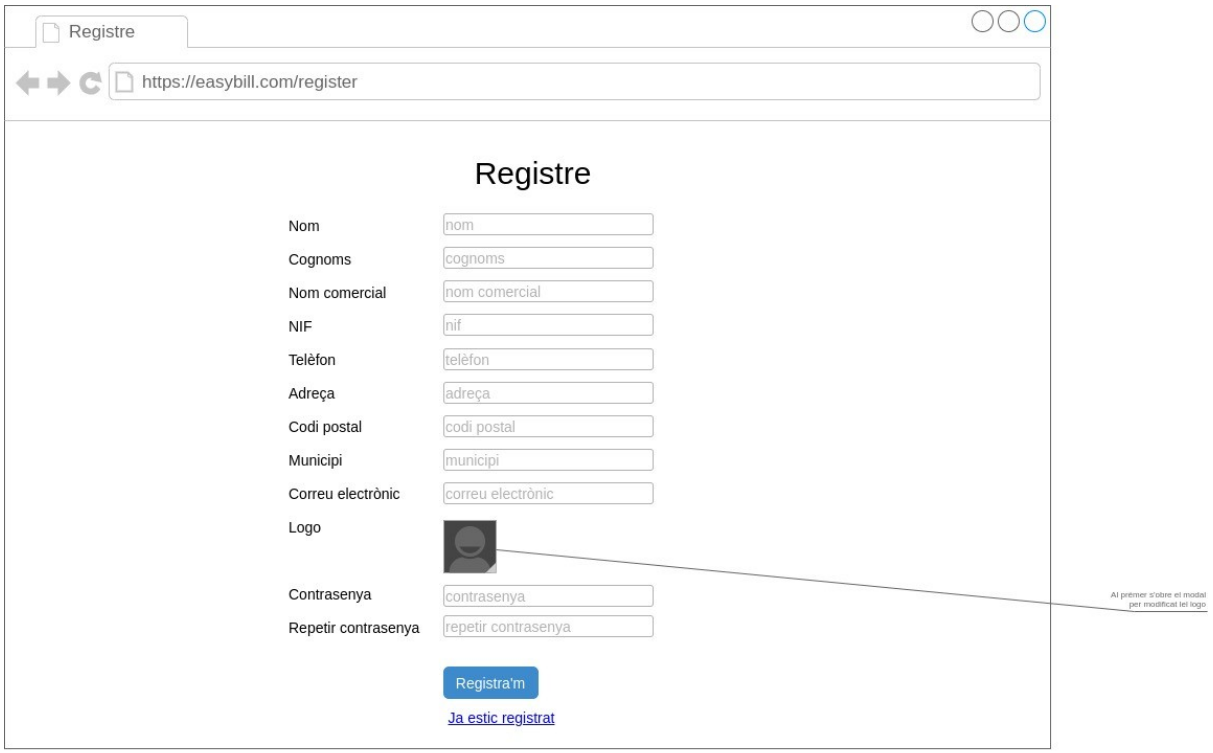
Desktop

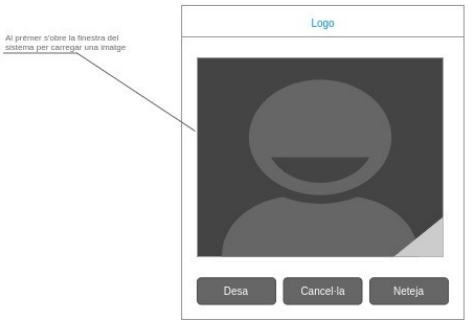


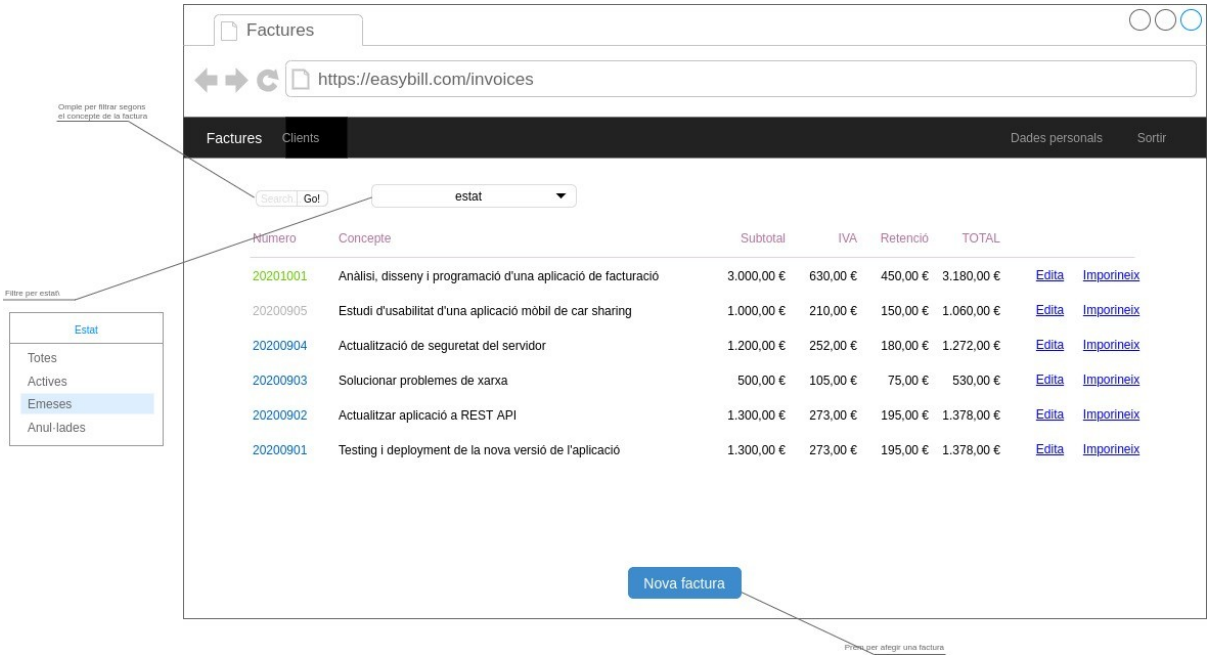

Entrar
https://easybill.com/login
Entrar
Correu electrònic
@
Username
Contrasenya
Password
Recordar contrasenya
Entra
Envia la contrasenya
i mostra el modal
No estic registrat

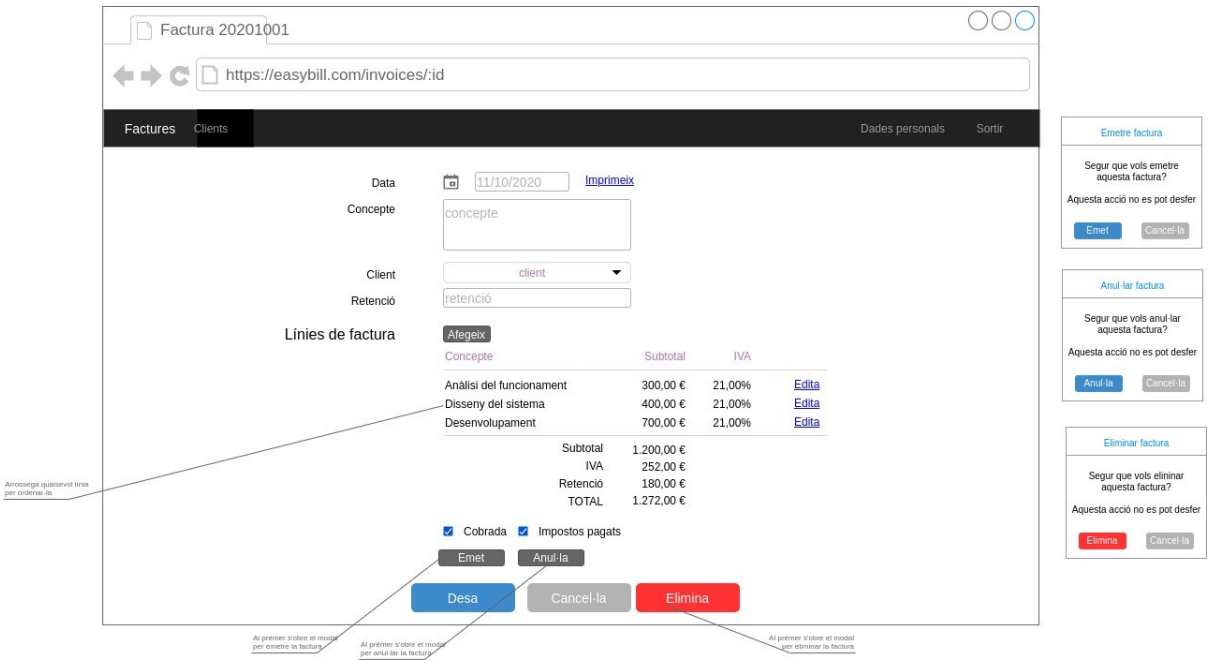

Us hem enviat un enllaç per
restablir la contrasenya.
Comproveu el correu electrònic.

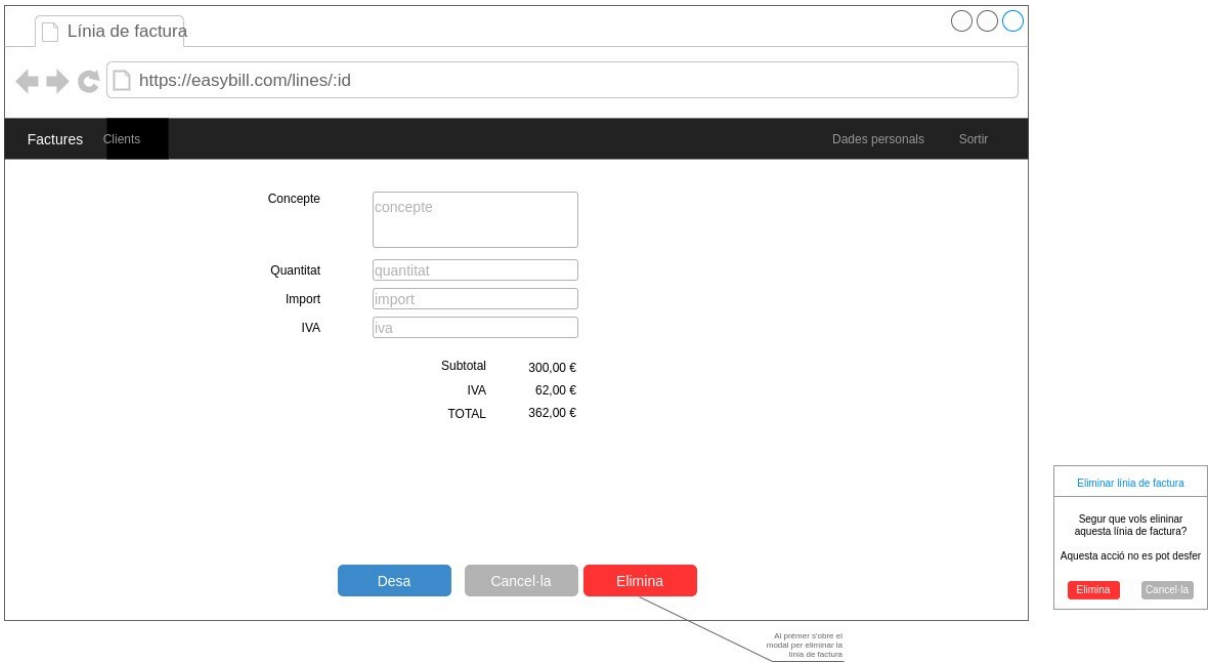

Recordar contrasenya
https://easybill.com/recover/:otp
Nova contrasenya
password
Repeteix contrasenya
password
Desa

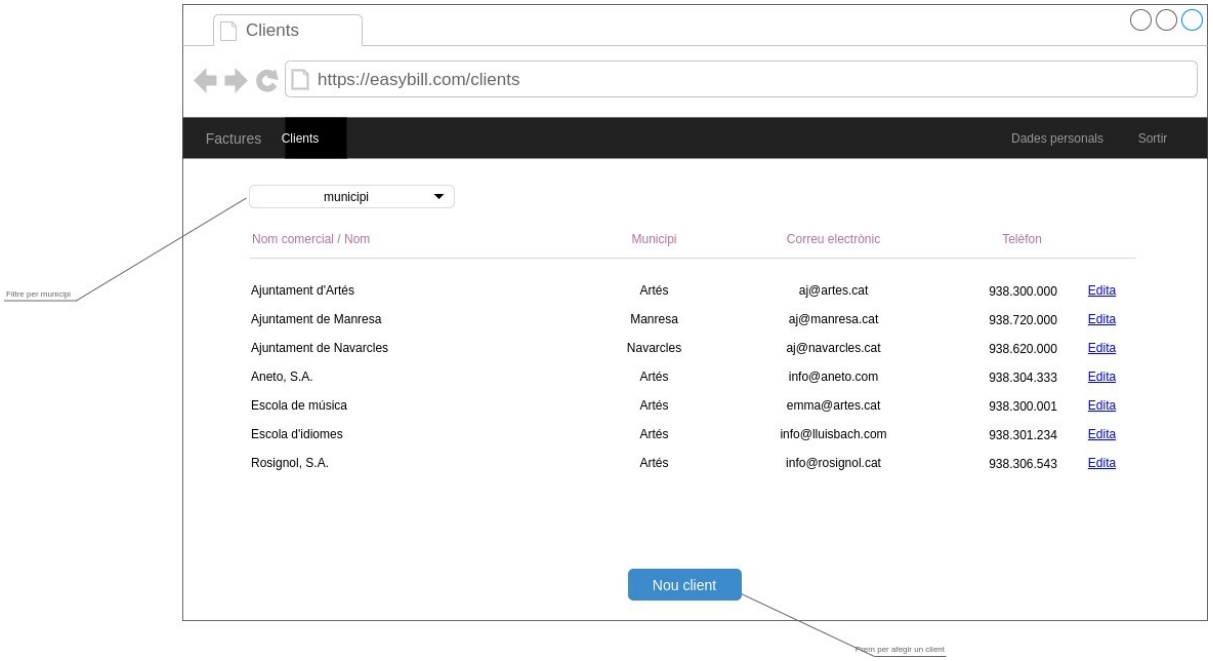

Registre
https://easybill.com/register
Registre
Nom
nom
Cognoms
cognoms
Nom comercial
nom comercial
NIF
nif
Telèfon
telèfon
Adreça
adreça
Codi postal
codi postal
Municipi
municipi
Correu electrònic
correu electrònic
Logo
Contrasenya
contrasenya
Repetir contrasenya
repetir contrasenya
Registra'm
Ja estic registrat
Al prémer s'obre el modal per modificat lel logo

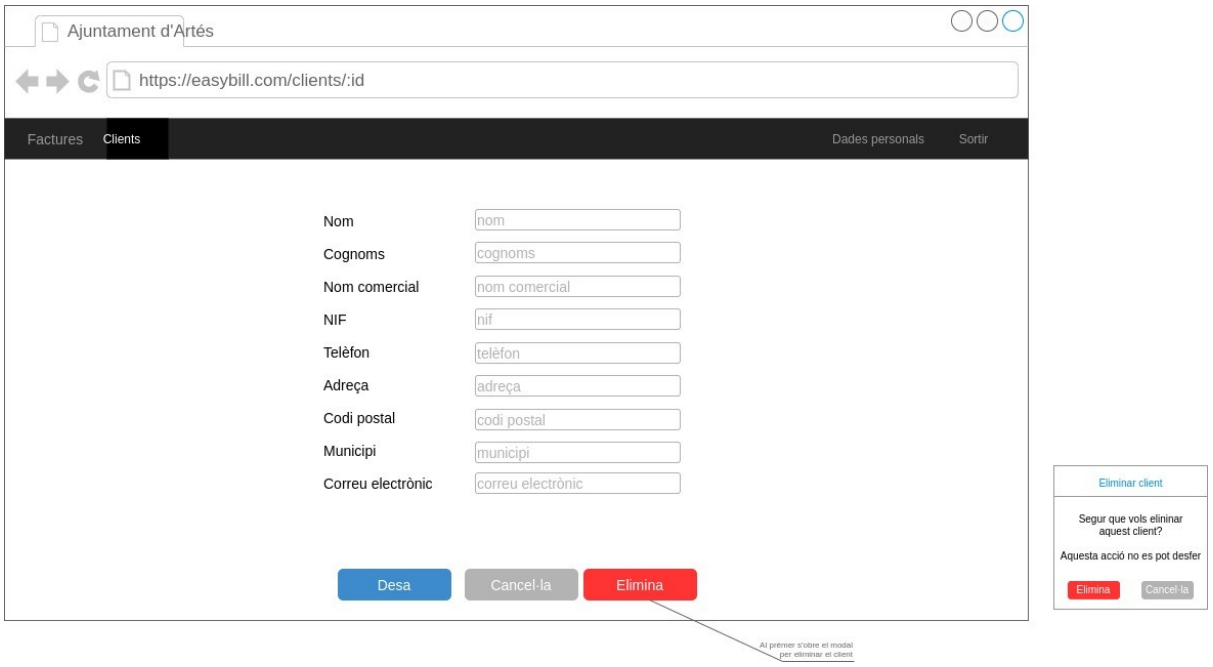

Al prémer s'obre la finestra del sistema per carregar una imatge
Logo
Desa
Cancel·la
Neteja

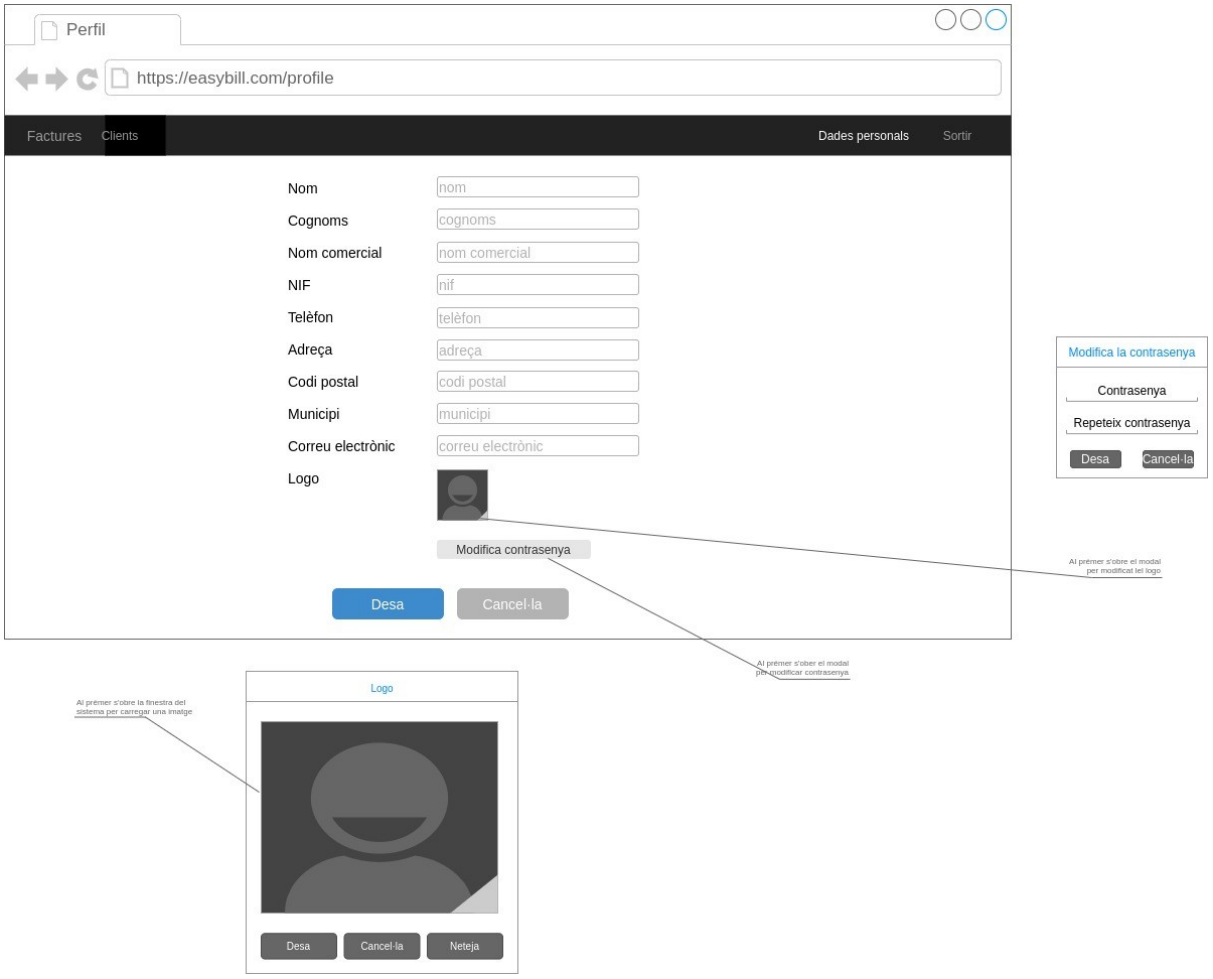

Factures
https://easybill.com/invoices
Omple per filtrar segons el concepte de la factura
Factures
Clients
Dades personals
Sortir
Search
Go!
estat
Filtre per estat
Estat
Totes
Actives
Emeses
Anul·lades
Numero
Concepte
Subtotal
IVA
Retenció
TOTAL
20201001 Anàlisi, disseny i programació d'una aplicació de facturació 3.000,00 € 630,00 € 450,00 € 3.180,00 € Edita Imprimeix
20200905 Estudi d'usabilitat d'una aplicació mòbil de car sharing 1.000,00 € 210,00 € 150,00 € 1.060,00 € Edita Imprimeix
20200904 Actualització de seguretat del servidor 1.200,00 € 252,00 € 180,00 € 1.272,00 € Edita Imprimeix
20200903 Solucionar problemes de xarxa 500,00 € 105,00 € 75,00 € 530,00 € Edita Imprimeix
20200902 Actualitzar aplicació a REST API 1.300,00 € 273,00 € 195,00 € 1.378,00 € Edita Imprimeix
20200901 Testing i deployment de la nova versió de l'aplicació 1.300,00 € 273,00 € 195,00 € 1.378,00 € Edita Imprimeix
Nova factura
Prem per afegir una factura


Factura 20201001
https://easybill.com/invoices/:id
Factures
Clients
Dades personals
Sortir
Data
11/10/2020
Imprimeix
Concepte
concepte
Client
client
Retenció
retenció
Línies de factura
Afegeix
Concepte
Subtotal
IVA
Anàlisi del funcionament 300,00 € 21,00% Edita
Disseny del sistema 400,00 € 21,00% Edita
Desenvolupament 700,00 € 21,00% Edita
Subtotal 1.200,00 €
IVA 252,00 €
Retenció 180,00 €
TOTAL 1.272,00 €
Cobrada
Impostos pagats
Emet
Anul·la
Desa
Cancel·la
Elimina
Arrossega qualsevol línia per ordenar-la
Al prémer s'obre el modal per emetre la factura
Al prémer s'obre el modal per anul·lar la factura
Al prémer s'obre el modal per eliminar la factura
Emetre factura
Segur que vols emetre aquesta factura?
Aquesta acció no es pot desfer
Emet
Cancel·la
Anul·lar factura
Segur que vols anul·lar aquesta factura?
Aquesta acció no es pot desfer
Anul·la
Cancel·la
Eliminar factura
Segur que vols eliminar aquesta factura?
Aquesta acció no es pot desfer
Elimina
Cancel·la


Línia de factura
https://easybill.com/lines/:id
Factures
Clients
Dades personals
Sortir
Concepte
concepte
Quantitat
quantitat
Import
import
IVA
iva
Subtotal 300,00 €
IVA 62,00 €
TOTAL 362,00 €
Desa
Cancel·la
Elimina
Al prémer s'obre el modal per eliminar la línia de factura
Eliminar línia de factura
Segur que vols eliminar aquesta línia de factura?
Aquesta acció no es pot desfer
Elimina
Cancel·la


Clients
https://easybill.com/clients
Factures
Clients
Dades personals
Sortir
municipi
Filtre per municipi
Nom comercial / Nom
Municipi
Correu electrònic
Telèfon
Ajuntament d'Artés Artés aj@artes.cat 938.300.000 Edita
Ajuntament de Manresa Manresa aj@manresa.cat 938.720.000 Edita
Ajuntament de Navarcles Navarcles aj@navarcles.cat 938.620.000 Edita
Aneto, S.A. Artés info@aneto.com 938.304.333 Edita
Escola de música Artés emma@artes.cat 938.300.001 Edita
Escola d'idiomes Artés info@lluisbach.com 938.301.234 Edita
Rosignol, S.A. Artés info@rosignol.cat 938.306.543 Edita
Nou client
Premi per afegir un client

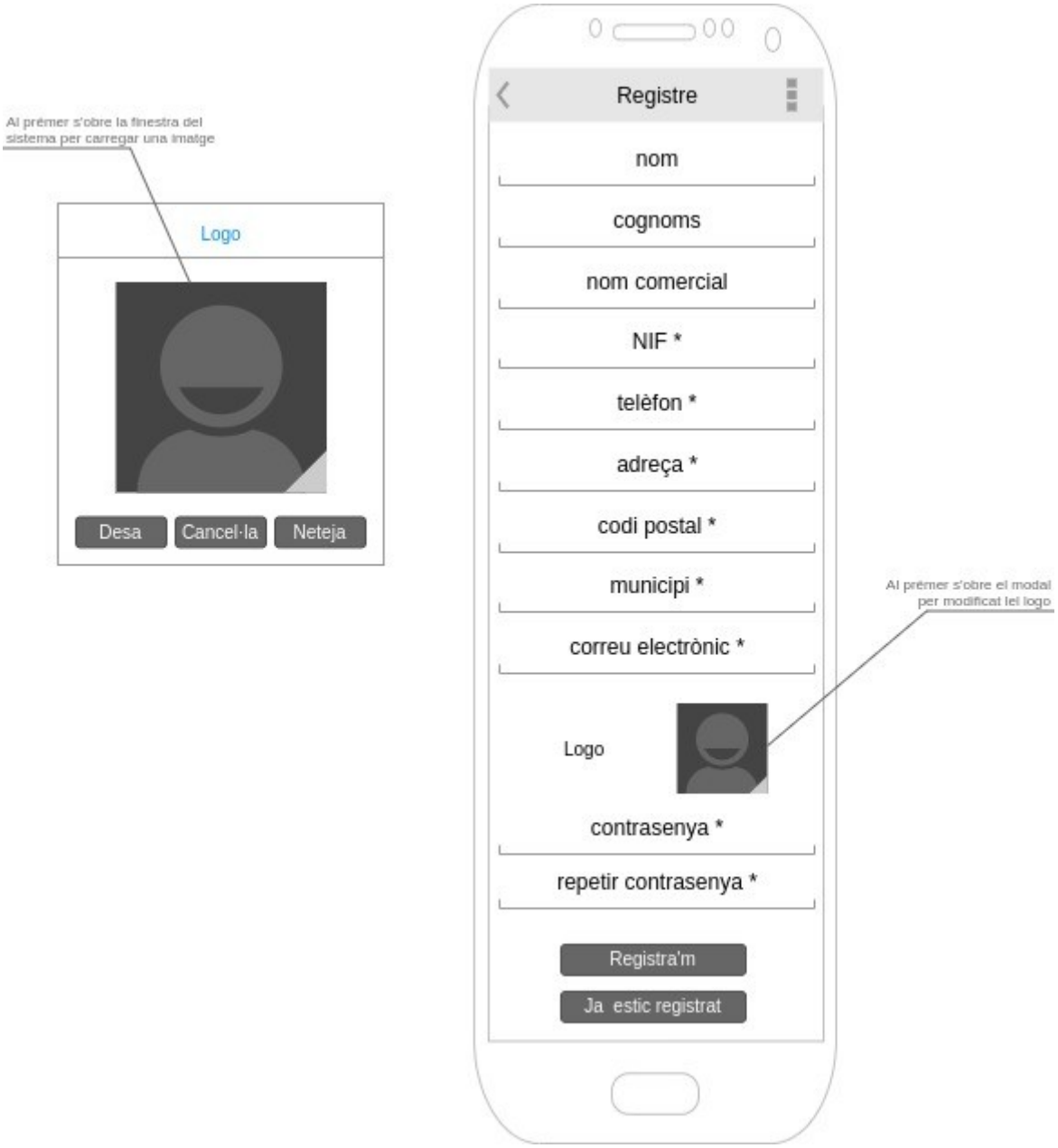

Ajuntament d'Artés
https://easybill.com/clients/:id
Factures
Clients
Dades personals
Sortir
Nom
nom
Cognoms
cognoms
Nom comercial
nom comercial
NIF
nif
Telèfon
telèfon
Adreça
adreça
Codi postal
codi postal
Municipi
municipi
Correu electrònic
correu electrònic
Desa
Cancel·la
Elimina
Al prémer s'obre el modal per eliminar el client
Eliminar client
Segur que vols elininar aquest client?
Aquesta acció no es pot desfer
Elimina
Cancel·la

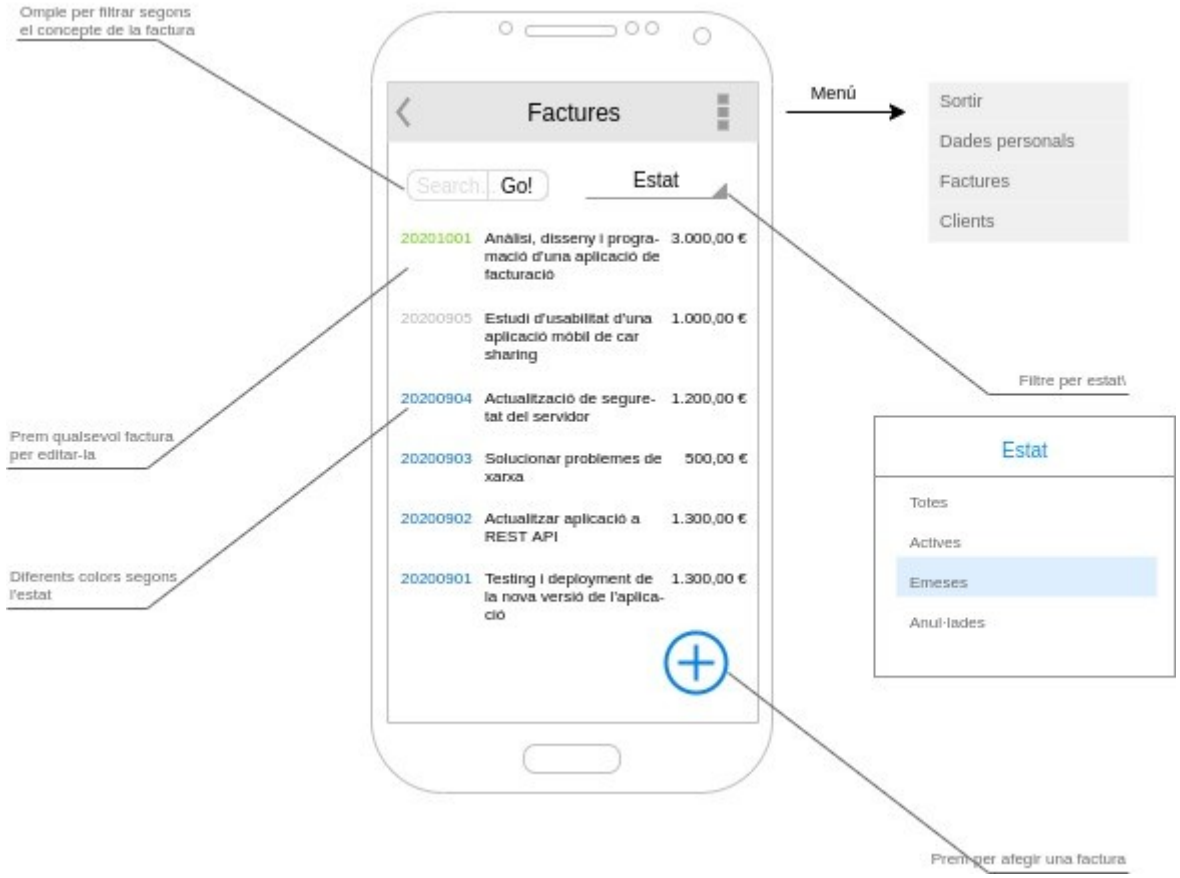

Perfil
https://easybill.com/profile
Factures
Clients
Dades personals
Sortir
Nom
nom
Cognoms
cognoms
Nom comercial
nom comercial
NIF
nif
Telèfon
telèfon
Adreça
adreça
Codi postal
codi postal
Municipi
municipi
Correu electrònic
correu electrònic
Logo
Modifica contrasenya
Desa
Cancel·la
Al prémer s'obre el modal per modificar el logo
Al prémer s'ober el modal per modificar contrasenya
Modifica la contrasenya
Contrasenya
Repeteix contrasenya
Desa
Cancel·la
Al prémer s'obre la finestra del sistema per carregar una imatge
Logo
Desa
Cancel·la
Neteja

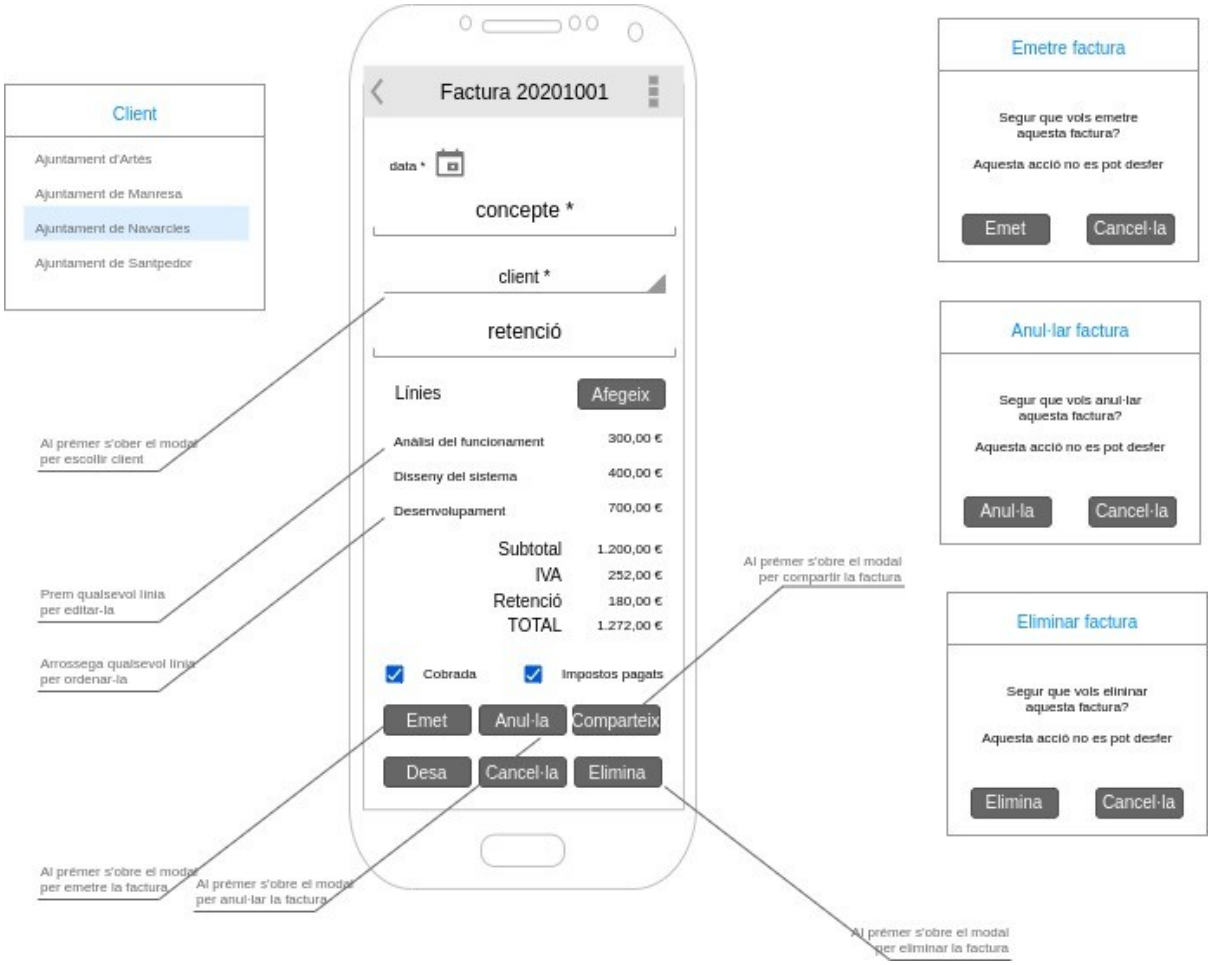
Mobile


Easybill
Programa de facturació simpe
Entrar
Registrar-me
Entrar
correu electrònic
contrasenya
Recorda contrasenya
Entra
No estic registrat
Envia la contrasenya
i mostra el modal
Us hem enviat un enllaç per
restablir la contrasenya.
Comproveu el correu electrònic.
D'acord


Recordar contrasenya
nova contrasenya *
repeteix contrasenya *
Desa


Al prémer s'obre la finestra del sistema per carregar una imatge
Logo
Desa
Cancel·la
Neteja
Registre
nom
cognoms
nom comercial
NIF *
telèfon *
adreça *
codi postal *
municipi *
correu electrònic *
Logo
Al prémer s'obre el modal per modificat lel logo
contrasenya *
repetir contrasenya *
Registra'm
Ja estic registrat

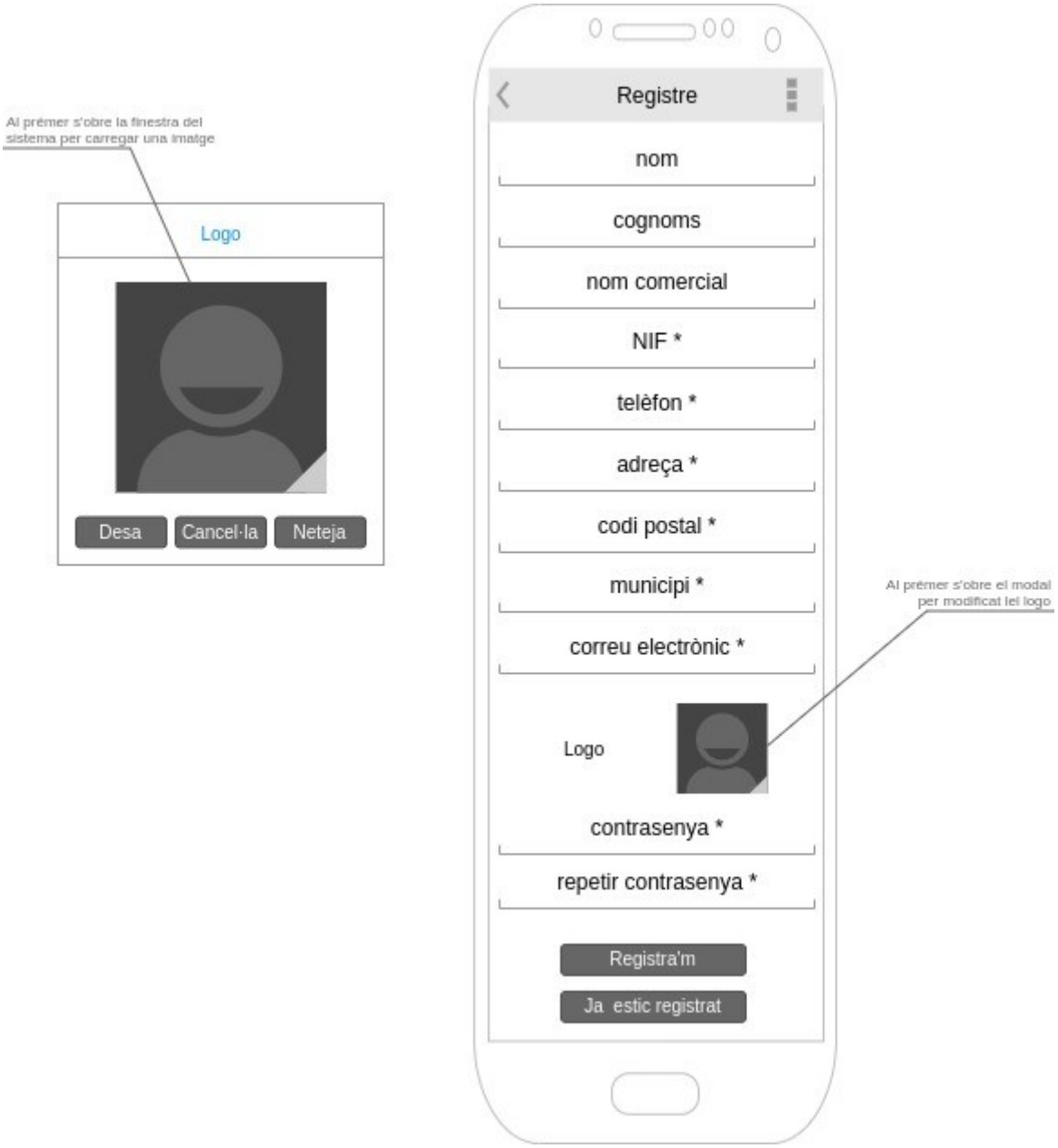

Omple per filtrar segons el concepte de la factura
Factures
Search...
Go!
Estat
Menú
Sortir
Dades personals
Factures
Clients
20201001 Anàlisi, disseny i programació d'una aplicació de facturació 3.000,00 €
20200905 Estudi d'usabilitat d'una aplicació mòbil de car sharing 1.000,00 €
20200904 Actualització de seguretat del servidor 1.200,00 €
20200903 Solucionar problemes de xarxa 500,00 €
20200902 Actualitzar aplicació a REST API 1.300,00 €
20200901 Testing i deployment de la nova versió de l'aplicació 1.300,00 €
Prem qualsevol factura per editar-la
Diferents colors segons l'estat
Filtre per estat\
Estat
Totes
Actives
Emeses
Anul·lades
Prem per afegir una factura


Client
Ajuntament d'Artés
Ajuntament de Manresa
Ajuntament de Navarcles
Ajuntament de Santpedor
Factura 20201001
data *
concepte *
client *
retenció
Línies
Afegeix
Anàlisi del funcionament 300,00 €
Disseny del sistema 400,00 €
Desenvolupament 700,00 €
Subtotal 1.200,00 €
IVA 252,00 €
Retenció 180,00 €
TOTAL 1.272,00 €
Cobrada
Impostos pagats
Emet
Anul·la
Comparteix
Desa
Cancel·la
Elimina
Emetre factura
Segur que vols emetre aquesta factura?
Aquesta acció no es pot desfer
Emet
Cancel·la
Anul·lar factura
Segur que vols anul·lar aquesta factura?
Aquesta acció no es pot desfer
Anul·la
Cancel·la
Eliminar factura
Segur que vols eliminar aquesta factura?
Aquesta acció no es pot desfer
Elimina
Cancel·la
Al prémer s'obre el modal per escollir client
Prem qualsevol línia per editar-la
Arrossega qualsevol línia per ordenar-la
Al prémer s'obre el modal per emetre la factura
Al prémer s'obre el modal per anul·lar la factura
Al prémer s'obre el modal per compartir la factura
Al prémer s'obre el modal per eliminar la factura


Línia de factura
concepte *
quantitat *
import *
iva *
Subtotal 300,00 €
IVA 62,00 €
TOTAL 362,00 €
Desa
Cancel·la
Elimina
Eliminar línia de factura
Segur que vols eliminar aquesta línia de factura?
Aquesta acció no es pot desfer
Elimina
Cancel·la
Al prémer s'obre el modal per eliminar la línia de factura


Filtra per municipi
Clients
Municipi
Ajuntament d'Artés
Artés
Ajuntament de Manresa
Manresa
Ajuntament de Navarcles
Navarcles
Aneto, S.A.
Artés
Escola de música
Artés
Escola d'idiomes
Artés
Rosignol, S.A.
Artés
Prem qualsevol client per editar-lo
Prem per afegir un client


Ajuntament d'Artés
nom
cognoms
nom comercial
NIF *
telèfon *
adreça *
codi postal *
municipi *
correu electrònic *
Desa
Cancel·la
Elimina
Eliminar client
Segur que vols eliminar aquest client?
Aquesta acció no es pot desfer
Elimina
Cancel·la
Al prémer s'obre el modal per eliminar el client


Al prémer s'obre la finestra del sistema per carregar una imatge
Logo
Desa
Cancel·la
Neteja
Registre
nom
cognoms
nom comercial
NIF *
telèfon *
adreça *
codi postal *
municipi *
correu electrònic *
Logo
Al prémer s'obre el modal per modificat lel logo
contrasenya *
repetir contrasenya *
Registra'm
Ja estic registrat

Annex 5. Guia d'usuari

Aquesta guia mostra pas a pas les diferents accions que es poden realitzar amb l'aplicació.

Inici

L'aplicació es carrega a través del web http://carlescanellas.cat/easybill

Registre

Per utilitzar el programa és necessari estar registrats com a usuari. El registre és lliure, i la única condició és escollir un NIF i un correu electrònic que no estiguin registrats.
Dins la pàgina d'inici, premem el botó **Registrar-se**. Ens portarà al formulari de registre.
Els camps marcats amb un asterisc són obligatoris, ja que són necessaris o bé com a dades de factura, o bé per entrar al sistema. La resta són opcionals.
Per omplir les dades de codi postal i municipi, ens podem ajudar del desplegable que s'obre prement el botó d'ubicació que hi ha davant el camp *Codi postal*. Només cal començar a escriure el nom per veure una llista de municipis que contenen la cadena entrada.
Qualsevol de les dades es pot modificar més endavant dins de l'apartat **Perfil**.

Entrada al sistema

Per entrar al sistema premem el botó Entrar de la pàgina d'inici. També ens hi redireccionarà l'aplicació cada cop que sortim del nostre compte d'usuari, després del registre o després de restaurar la contrasenya.
Com a usuari cal entrar el nostre correu electrònic. La contrasenya és la que hem escollit en el moment del registre.
Si no estem registrats o no recordem la contrasenya, podem prémer l'enllaç corresponent a cadascuna d'aquestes accions.

Restauració de contrasenya

En cas de no recordar la contrasenya de registre, podem demanar a l'aplicació que la restauri.
A la pàgina d'entrada al sistema, escrivim el nostre correu electrònic i deixem la contrasenya en blanc. Premem el botó **No recordo la contrasenya** i l'aplicació ens enviarà un correu electrònic amb un enllaç. Aquest enllaç ens portarà a una pàgina que ens permetrà escollir una contrasenya nova.
És important recalcar que, per motius de seguretat, la validesa de l'enllaç és d'un dia.

Llistat de factures

Prement l'enllaç **Factures** del menú principal, l'aplicació ens mostra totes les factures creades. El número de factura és de color diferent segons si la factura és activa (blau), emesa (verd) o anul·lada (vermell).

Sobre la llista i ha una barra de filtre. Podem filtrar les factures pel concepte, per l'estat o bé pel client.
Des del llistat podem editar les factures i veure una vista prèvia d'impressió.
Per afegir una factura, premem el botó de sota la llista. És necessari tenir com a mínim un client per poder crear factures.

Edició i creació d'una factura

Prement el botó **Edita** del llistat de factures, visualitzem i podem editar les dades d'una factura.
També podem crear una factura nova prement el botó **Afegir factura** del llistat de factures.
Per afegir una factura nova, és necessari especificar un concepte, un client i el % de retenció.
La data per defecte és l'actual, però la podem canviar si ho desitgem. El sistema assignarà un número a la factura segons aquesta data.
Al desar la factura, se'ns demanarà si volem afegir una línia de factura. Podem fer-ho ara o més endavant.
Les dades d'una factura (data, concepte, client, retenció i línies de factura) es poden editar sempre que aquesta estigui activa. També podem portar el registre de si ha estat cobrada i de si n'hem pagat les taxes corresponents.
En cas que la factura hagi estat emesa, no podrem canviar les dades de la factura. Una factura emesa és una factura que ja estat presentada al client, per tant no en podem modificar cap de les dades. Sí que podem fer, però, el seguiment de cobrament i taxes.
Una factura es pot anul·lar. Suposem que hem presentat una factura, conté un error i cal repetir-la. A l'estar emesa, necessitem tenir un registre de la factura errònia. Les factures anul·lades no admeten cap canvi, i no permeten fer el seguiment de cobrament i taxes.
Si una factura és activa, també podrem eliminar-la.
Les accions d'emissió, anul·lació i eliminació d'una factura són irreversibles. El programa ens avisarà abans de completar-les per estar-ne segurs.

Llistat de clients

Accedim al llistat de clients des de l'enllaç **Clients** del menú principal.
A la barra superior podem filtrar els clients segons nom/cognoms/nom comercial i municipi.
Per afegir un client, premem el botó de sota la llista.
Per editar un client, premem el botó **Edita** corresponent al client.

Edició i creació d'un client

Prement el botó **Edita** del llistat de clients, visualitzem i podem editar les dades d'un client. També podem crear un client nou prement el botó **Afegir client** del llistat de clients.
És necessari que el NIF i el correu electrònic no estiguin registrats a cap altre client.
Els camps marcats amb un asterisc cal emplenar-los, ja que són necessaris per fer una factura.

Totes les dades d'un client es poden canviar en qualsevol moment. Cal tenir en compte que els canvis només es reflectiran en les factures que no hagin estat emeses en el moment de fer aquests canvis.
L'eliminació d'un client és irreversible, i només és possible si el client no té cap factura assignada.

Perfil d'usuari

Podem modificar el nostre perfil d'usuari en qualsevol moment prement l'enllaç **Dades personals** del menú principal.
Totes les dades són modificables, però cal tenir en compte que els canvis només es reflectiran en les factures no emeses.

Sortir

El botó **Sortir** del menú principal ens permet sortir de l'aplicació com a usuari. Si no sortim, quan tornem a obrir l'aplicació aquesta ens recordarà com a usuari i no caldrà que tornem a entrar les nostres credencials. Pe seguretat, l'aplicació només ens recordarà com a usuaris durant 24 hores.

Annex 6. Bibliografia

Imatges

Les imatges utilitzades en l'aplicació i les presentacions estan extretes de https://pexels.com

Arxiu: pexels-allan-mas-5383758.jpg
Crèdits: Foto de Allan Mas de Pexels
https://www.pexels.com/ca-es/foto/saludable-home-paret-corda-5383758/

Arxiu: pexels-polina-zimmerman-3782134.jpg
Crèdits: Foto de Polina Zimmerman de Pexels
https://www.pexels.com/ca-es/foto/escrivint-notes-negoci-paper-3782134/

Arxiu: pexels-pixabay-262438.jpg
https://www.pexels.com/ca-es/foto/blanc-colpejar-concentrar-se-conceptual-262438/

Arxiu: pexels-mathias-pr-reding-5466436.jpg
Crèdits: Foto de Mathias P.R. Reding de Pexels
https://www.pexels.com/ca-es/foto/ciutat-encreuament-de-camins-carretera-carrer-5466436/

Arxiu: pexels-matthias-groeneveld-4200740.jpg
Crèdits: Foto de Matthias Groeneveld de Pexels
https://www.pexels.com/ca-es/foto/assolellat-vermell-sort-taula-4200740/

Arxiu: pexels-eberhard-grossgasteiger-1670187.jpg
Crèdits: Foto de eberhard grossgasteiger de Pexels
https://www.pexels.com/ca-es/foto/alba-paisatge-naturalesa-cel-1670187/

Arxiu: pexels-adam-borkowski-5068329.jpg
Crèdits: Foto de Adam Borkowski de Pexels
https://www.pexels.com/ca-es/foto/escales-ciutat-edifici-abstracte-5068329/

Arxiu: pexels-energepiccom-313690.jpg
Crèdits: Foto de energepic.com de Pexels
https://www.pexels.com/ca-es/foto/cabell-cabells-disseny-dona-313690/

Arxiu: pexels-ketut-subiyanto-4560092.jpg
Crèdits: Foto de Ketut Subiyanto de Pexels
https://www.pexels.com/ca-es/foto/home-portatil-internet-parc-4560092/

Arxiu: pexels-photomix-company-101808.jpg
Crèdits: Foto de PhotoMIX Company de Pexels
https://www.pexels.com/ca-es/foto/bloquejar-bloquejat-claus-concentrar-se-101808/

Arxiu: pexels-nastyasensei-335393.jpg
Crèdits: Foto de NastyaSensei de Pexels
https://www.pexels.com/ca-es/foto/terra-univers-viatjar-negoci-335393/

Arxiu: pexels-ksenia-chernaya-5691615.jpg
Crèdits: Foto de Ksenia Chernaya de Pexels
https://www.pexels.com/ca-es/foto/construccio-taula-desenfocament-disseny-5691615/

Arxiu: pexels-snapwire-310983.jpg
https://www.pexels.com/ca-es/foto/accio-adult-anant-amb-bici-anant-en-bici-310983/

Arxiu: pexels-andrea-piacquadio-826349.jpg
Crèdits: Foto de Andrea Piacquadio de Pexels
https://www.pexels.com/ca-es/foto/adult-beguda-bellesa-beure-826349/

Arxiu: pexels-rodolfo-clix-1036936.jpg
Crèdits: Foto de Rodolfo Clix de Pexels
https://www.pexels.com/ca-es/foto/articles-de-vidre-blau-bombeta-brillant-1036936/

Música

La música de la presentació pública està extreta de https://freemusicarchive.org

Arxiu: acoustic-indie-folk-by-scott-holmes-music.mp3
Pista: Acoustic Indie Folk
Artista: Scott Holmes Music
Llicència: attribution, non commercial. https://creativecommons.org/licenses/by-nc/4.0/

Annex 7. Vita

El meu nom és Carles Canellas Crusellas. Vaig néixer fa 49 anys a Artés, un poble del Bages. Vaig estudiar EGB, batxillerat i COU. Posteriorment vaig fer una diplomatura d'enginyeria elèctrica especialitat electrònica a l'EUPM de Manresa. Vaig treballar en diferents sectors, destacant l'etapa que vaig treballar a l'empresa Naturgest, SL entre els anys 1999 i 2003 com a delineant i programador de SIG. Després d'uns anys sense treballar a causa d'un accident de moto, vaig començar a estudiar enginyeria informàtica a la UOC el 2011. Més tard vaig començar el màster de desenvolupament de llocs i aplicacions web, que ara finalitzo amb aquest treball. Actualment treballo com a autònom de programador, desenvolupador d'aplicacions i llocs web i en el sector audiovisual. Combino la meva feina amb les meves aficions, sobretot el ciclisme de carretera.