



UNIVERSITAT POLITÈCNICA DE CATALUNYA

BARCELONATECH

Escola Politècnica Superior d'Enginyeria
de Manresa



MEMÒRIA DEL PROJECTE

INTEGRACIÓ DE SISTEMES

Ivan Chamero

Alejandro Prieto

Manuel Ángel Román

Jaume Serra

Gener 2022

Índex

1 INTRODUCCIÓ	2
2 OBJECTIUS	3
3 ESTRUCTURA DEL SISTEMA	4
4 DISPOSITIU HARDWARE	6
4.1 CONTROL DE HARDWARE	7
4.2 ESTRUCTURACIÓ DEL SOFTWARE	9
5 SOFTWARE EN EL HOSTING	18
5.1 SERVIDOR WEB	18
5.2 BASE DE DADES	21
5.3 WEBSITE	22
5.3.1 Part sense login	22
5.3.2 Rol Administrador	26
5.3.3 Rol d'usuari	35
5.4 CONTROL INTEL·LIGENT	36
6 CONCLUSIONS	38
7 ANNEXOS	39
7.1 INTERFÍCIE GRÀFICA	39
7.2 CONFIGURACIÓ A NIVELL LOCAL	41
7.3 PLACA	42
7.3.1 Esquemàtic del dispositiu	42
7.3.2 Disseny 3D de la caixa	43
7.4 ORGANITZACIÓ DEL GRUP DE TREBALL	45
7.4.1 Calendari	45
7.4.2 Repartiment de Tasques	46

1 INTRODUCCIÓ

En aquest document es descriu la realització i desenvolupament del nostre projecte d'Integració de Sistemes. Es tracta de la gestió, disseny i implementació del sistema de control d'un laboratori tecnològic obert a la ciutadania, gestionat per l'EPSEM. El sistema de control del TechLab ha de proporcionar un control remot de les màquines del laboratori controlant l'activació i desactivació d'aquestes.

El nostre sistema parteix de la idea de l'assignatura d'Enginyeria de Sistemes on vam fer un sistema de control d'aforament a les aules a causa de la COVID-19. Aquest sistema permetia controlar l'accés a les aules a través d'un dispositiu encastrat que es connectava a un servidor local, i al comunicar-se amb la base de dades donava accés o no als usuaris. El mètode de funcionament del dispositiu aprofitava la idea del NFC, els missatges per pantalla i la comunicació amb el servidor a través de WiFi.

2 OBJECTIUS

L'objectiu principal del projecte és crear un site adaptatiu a qualsevol dispositiu, accessible i intuitiu per tothom per poder gestionar remotament totes les funcions del TechLab, ja sigui la gestió i informació de les reserves, consultar l'estat i consum de les màquines actuals, modificar l'estat de les diferents màquines així com els seus paràmetres de connexió o característiques, limitació dels temps de reserva dependent del rol, juntament amb més aplicacions que anirem veient al llarg de la memòria. Una característica important que ha de tenir aquest site és que es trobarà en un servidor amb IP pública i protocol HTTPS de seguretat.

A més a més d'aquest site web, l'altre gran objectiu del projecte és, com bé s'ha comentat, és programar la placa proporcionada a través de la programació en Micropython del microcontrolador ESP-32. Aquesta configuració es durà a terme a partir de la programació dels diferents mòduls com són el NFC, la OLED, entre altres i la programació d'un programa principal que gestionarà tots els diferents estats del sistema. Tenim com a finalitat fer d'aquest sistema fàcil de configurar i d'usar, que disposi d'una intel·ligència per detectar possibles anomalies de funcionament. Respecte al tema energètic, el nostre sistema gestionara el consum de les diferents màquines en temps real.

En termes més específics, un dels objectius grans interrelacionant les dues parts del projecte és el control d'accés/aforament al TechLab. En aquest apartat ens sorgeix la necessitat de realitzar un sistema de control dels usuaris amb la modificació dels permisos de cada rol different d'accés diferenciant entre administrador i usuari normal.

Un tema en comú en tot el projecte és la necessitat de la seguretat, així que farem del nostre sistema un sistema robust. La robustesa en la web la trobarem a partir de la gestió dels errors de les possibles reserves o de la necessitat d'emplenar els camps obligatoris en alguns templates, en la part de la placa la trobarem en la gestió de tot tipus d'errors i necessitat de recuperar els estats anteriors als problemes sorgits. Finalment, per protegir la placa de cops o agents externs, es dissenyarà una caixa 3D adaptada a les dimensions de la placa.

3 ESTRUCTURA DEL SISTEMA

El sistema global està separat en diferents parts que interactuen les unes amb les altres a través de protocols *http* per sobre d'una capa de seguretat *TLS*, interconsultes a la base de dades, entre altres. El següent esquema mostra l'estructura del sistema complet:

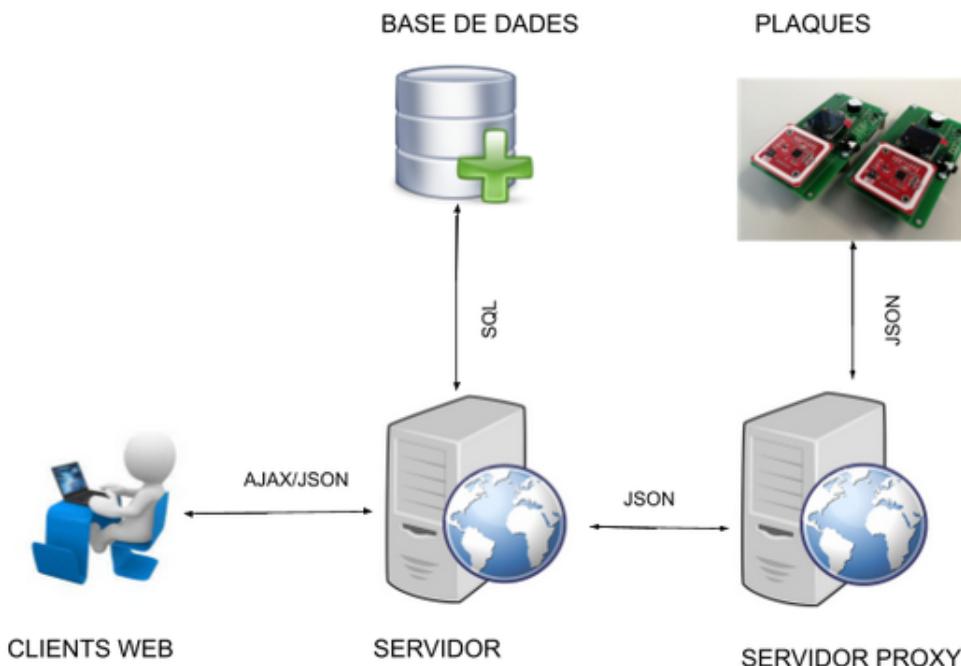


Figura 1: Estructura del sistema

Com es pot observar en la figura anterior, les plaques connecten amb un servidor proxy local a la xarxa wifi on estàn connectades a través d'*http*. Aquest servidor proxy envia les peticions al servidor que està en un hosting i aquest fa el tractament de dades necessari connectant amb la base de dades a través de sentencies *SQL* hostejades en *python3* gràcies a les funcionalitats que ofereix *sqlite3*. La comunicació es bidireccional de manera que el servidor ofereix una resposta en *JSON* encapsulada dintre d'una trama *http* amb un xifrat *TLS*. Per altre part, estan els clients web que connecten al servidor a través d'*http* xifrat amb una capa *TLS* i per fer un intercanvi dinàmic de les dades s'utilitza *AJAX*. També s'ha fet una aplicació d'escriptori portàtil per canviar el Wifi de les plaques que comunica directament amb el *api restful* del servidor.

El servidor està fet en *python3* amb el framework *flask*. Aquest es troba en un hosting amb IP pública que permet la interconnexió dels dispositius clients a través de la xarxa d'internet. El hosting ofereix un servei web amb apache que fa una redirecció del port 5000 (el de flask) sobre el port 443 (que ofereix una capa de seguretat) a través d'un proxy pass reverse.

El servidor proxy es un intermediari entre la comunicació de la placa i el servidor flask del hosting. Les plaques es connecten aquest servidor a través de la seva direcció IP i el port on s'ofereix el servei (5000). El servidor proxy està fet en flask també. Aquest servei s'ha de fer en un client de la xarxa on estiguin connectades totes les màquines i ha de tindre una direcció IP estàtica perquè les plaques puguin connectar sempre amb ell. Si volem traslladar el sistema perquè estigui connectat en un altre punt d'accés, s'ha de posar el servidor proxy amb la IP que comuniquen totes les plaques.

La estructuració de la placa s'explicarà a continuació però en resum, està programada amb el framework de *micropython* sobre un *ESP32* com a nucli del sistema. Aquest interactua amb el hardware que incopora la PCB a través de les llibreries de micropython i altres mòduls que s'han fet apart. La connexió amb el servidor es fa a través de la llibreria *request* amb http xifrat sobre una capa TLS. Les màquines es connecten als connectors de la PCB i es pot tenir el control de la màquina a través de la intercomunicació amb el servidor en mode administrador. Les plaques envíen valors d'estat i consum constant al servidor gràcies a un sensor de corrent no intrusiu amb sortida *JACK* per l'entrada de la placa.

El servidor ofereix un servei web que està connectat a una base de dades SQL. El servidor està dividit en *blueprints* de manera que està tot més ordenat. Conté un *api restful* interconectada a la base de dades que permet la intercomunicació i tracta les dades de les peticions dinàmiques dels clients web com les de les plaques del Techlab i les aplicacions gràfiques per canviar de Wifi.

4 DISPOSITIU HARDWARE

Una part fonamental en el projecte és el control del propi hardware que controla les màquines. El nucli del hardware es basa en un microcontrolador *ESP32* que actua com a cervell i es connecta al servidor proxy per enviar dades de les màquines cap al servidor. La *PCB* va ser proporcionada per l'escola com a continuació de l'assignatura passada on controlavem l'aforament en les aules.

El *ESP32* va ser programat amb *micropython* aprofitant les llibreries que aquest incorpora. Abans de tot, es va flashejar la placa perquè sigüés compatible amb el firmware de *micropython*. Com a editor de codi i per interactuar amb el *ESP32* es va utilitzar el software de *Thonny*.

La funció fonamental de la placa és permetre l'accés a la màquina per als usuaris apartir de la comunicació bidireccional amb el servidor i la lectura NFC. Per això, els usuaris del TechLab han hagut de reservar la màquina tenint els permisos d'utilització i ús d'aquesta. La placa manté una comunicació constant amb el servidor a través de peticions http encapsulades sobre una capa TLS de seguretat. La informació del servidor es rep en format *JSON* i segons el contingut del missatge fa una acció o un altre.

La *PCB* compta amb un lector *PN532 NFC* amb el qual es llegeixen les targetes. També té una pantalla *OLED* per permetre la comunicació visual amb l'usuari. També compta amb un brunzidor per possibles alarmes, led per interactuar amb l'usuari, un relé per tindre control sobre la màquina, connectors per connectar la màquina i tindre el control i una entrada *JACK* per mesurar el corrent de la màquina gràcies a un sensor de corrent no intrussiu *SCT013*. La següent imatge mostra la placa:



Figura 2: PCB del nostre sistema

4.1 CONTROL DE HARDWARE

Per controlar el led, el brunzidor, rele i detector de corrent vam crear el mòdul *hard_aux* que està estructurat en classes. Cada mòdul hardware representa una classe diferent i tots aquests pertanyen a la classe *Hardware* on podem controlar tots ell a la vegada amb una instància d'aquesta classe en el programa principal *main*. Vam decidir fer-ho en classes ja que cada mòdul es independent de l'altre i així ho tindríem més organitzat. Per altre banda, tenir una espècie de super classe *Hardware* ens permetia tenir el control d'aquests elements amb només una sola instància en el programa principal. Els atributs de la classe hardware són instàncies de les altres classes i d'aquesta manera l'accés a cadascún d'ells desde el programa principal es trivial.

La classe *Led* representa una abstracció d'un led. Permet interactuar amb el led de manera directa posant-lo en ON o OFF i de diversos colors. Es basa en activar o desactivar el pin físic de sortida del ESP32 que està connectat al led en la PCB.

La classe *Brunzidor* representa una abstracció d'un brunzidor. Permet generar tons per avisar a l'usuari durant un cert temps. Es basa en generar polsos per fer vibrar la membrana del brunzidor durant un temps. Per fer-ho s'activa i es desactiva el pin corresponent durant un cert temps.

La classe *Rele* representa una abstracció del relé. Permet commutar la tensió dels connекторs per encendre o apagar la màquina. Per fer-ho, s'activa i es desactiva un pin.

La classe *Sensor_Corrent* reprensenta una abstracció d'un lector de corrent. Aquesta part va ser molt difícil d'elaborar ja que la PCB té un circuit que afegeix un offset i està fet per una tensió de referència d'1.5V quan el *ESP32* té una tensió de referencia del ADC de 1.1V. Per llegir els valors del sensor vam tindre que atenuar al màxim el senyal per poder situar-lo correctament sinó amb l'offset per defecte sortien malament les dades.

Per poder llegir els valors de corrent vam tindre que rectificar el senyal fent un mostreig ja que arriba una tensió alterna en la entrada JACK. Vam observar que hi havia molt de soroll en la entrada del ADC així que vam optar per fer un filtre de mitjana mòvil per reduir-lo.

Per fer les proves vam utilitzar una bombeta de 40W de potència i vam observar que per càrregues menors a 100W les lectures tenien error ja que el sensor es capaç de llegir càrregues fins a 30.000A aproximadament. També vam probar-ho amb una pistola d'aire calent de 300W i vam obtindre mesures molt precises. Aquest va ser el motiu principal pel qual vam optar per posar un factor de calibració per càrregues inferiors a 100W.



Figura 3: Transformador de corrent SCT013

El sensor de corrent fa un filtre de mitjana mòvil de valors que llegueix l'ADC restant-li l'offset que aquest incorpora. Per saber el valor total de tensió calculem el valor de la tensió RMS. Un cop obtinguda fem la conversió a corrent i calculem sabent que a l'entrada del ADC hi ha una resistència de 33Ohm i la relació de transformació del transformador de corrent *SCT013*.



Figura 4: Bombeta de prova de 40W

Per connectar al Wifi s'utiliza les funcionalitats de la llibreria *Network*. La connexió és mode client i té un timeout per connectar-se a la xarxa. Pel que fa el lector NFC i la pantalla s'utilitzen les seves respectives llibreries.

4.2 ESTRUCTURACIÓ DEL SOFTWARE

El programa principal de la placa es basa en una màquina d'estats que interactua amb els diversos elements apartir de variables globals. El següent diagrama mostra el graf de la màquina d'estats:

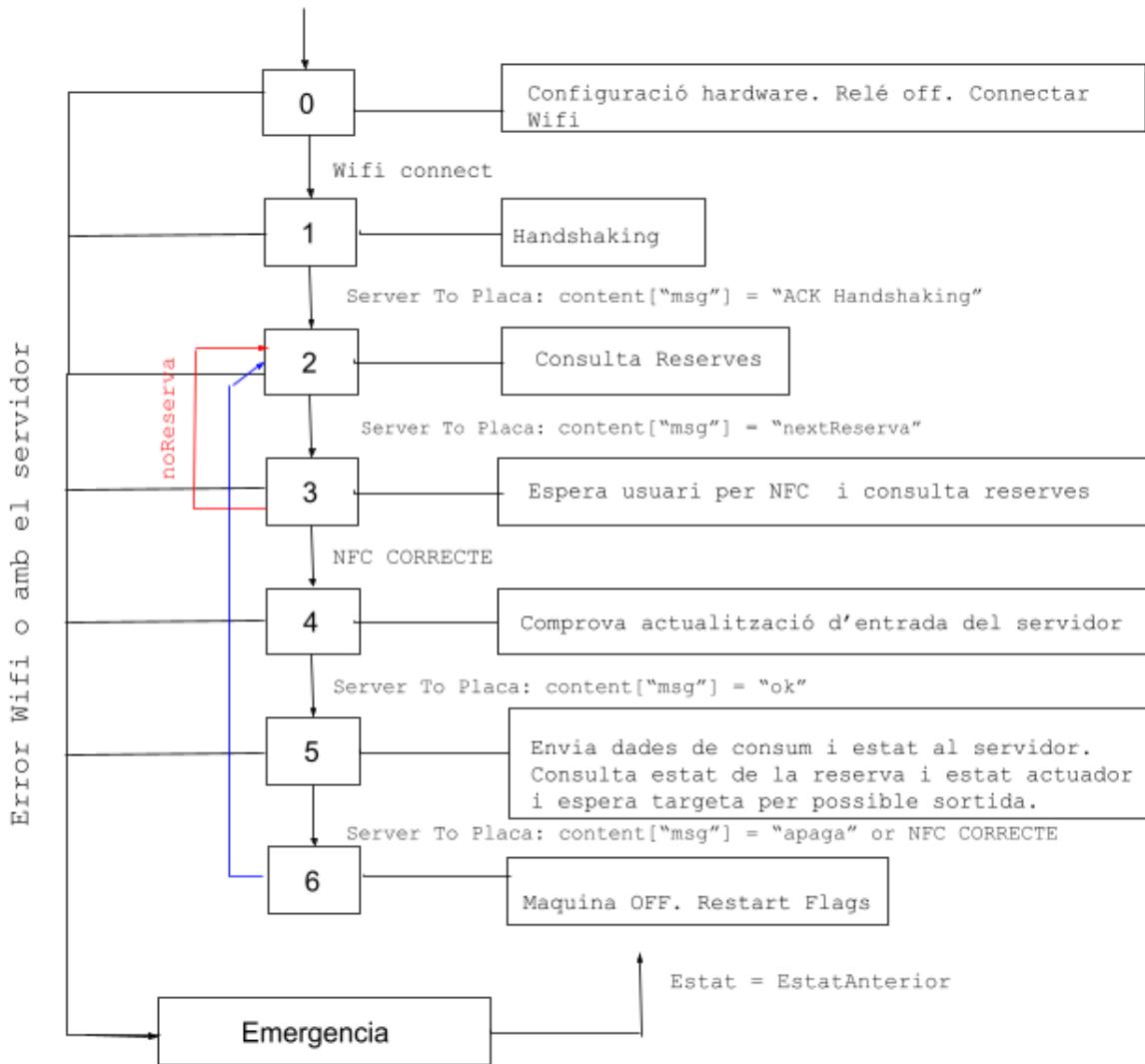


Figura 5: Màquina d'estats de la placa

Com podem observar en la figura anterior, l'autòmat comença en **Estat0** on configura el Hardware (posant el relé a off, inicialitzant brunzidor, leds...) i es connecta al punt d'accés. Durant aquest estat el led de la placa està en color vermell. Per la pantalla es mostra que la màquina es troba apagada mostrant el missatge *MAQUINA OFF* i que s'està intentant connectant a la xarxa Wifi. L'SSID de la xarxa i el password es declaren com a variables globals de manera que podem accedir desde qualsevol part del programa per modificarles així com la pantalla, l'NFC, diversos flags i l'instància de la classe *Hardware* per interactuar

amb el relé, leds, brunzidor i detector de corrent. En aquest estat la màquina es troba apagada ja que no hi ha ninguna reserva ni ningú usuari. És un estat d'inicialització. En aquest estat l'ESP32 es configura com a mode client del punt d'accés. Si després d'un timeout no s'ha pogut connectar, ho intenta de nou fins que ho aconsegueix i canvia d'estat. El canvi d'estat es produeix quan s'ha pogut connectar al punt d'accés. D'altra banda, en cas que hi hagi un error en el Wifi anirà a l'estat d'**Emergència** on informarà per la pantalla que hi ha hagut un error wifi i la màquina està apagada. Al entrar a l'estat d'emergència per primera vegada farà un so pel brunzidor informant que hi ha hagut una emergència. En l'estat d'emergència el led de la placa es posarà de color blau. Després intentarà reconectar al Wifi i ho mostrerà per pantalla com està reconnectant. La màquina es quedarà en aquest estat mentre no es pugui connectar al Wifi. En el moment que la màquina es connecta al Wifi ho mostra per pantalla. A continuació veurem algunes imatges del missatges i l'estat del led:



Figura 6: Placa intentant connectar al Wifi



Figura 7: Placa connectada al Wifi. Transició d'estat

Un cop connectada al Wifi, en l'**Estat1** la placa fa la primera connexió amb el servidor per identificar-se com a client. Es tracta d'un estat de handshaking amb el servidor on aquest li envia el missatge en *JSON* "*ACK Handshaking*" amb el factor de calibració de la màquina i el Wifi i password del punt d'accés que té. Si la màquina no existeix el servidor enviarà "*NACK Handshaking*". La transició d'estat es provoca quan rep el *ACK* del servidor. En cas que no pugui connectar amb el servidor anirà a l'estat d'emergència on es mostrerà per pantalla un error de servidor i el led de la placa estarà en blau i el brunzidor per primera vegada sonarà. Després tornarà a l'estat anterior i intentarà reconnectar amb el servidor. Si el servidor li envia una xarxa diferent a la que està connectada s'intentarà connectar amb un timeout i tant en cas exitós com de fracàs se li informarà al servidor en el següent estat. En cas de fracàs, la placa s'intentarà connectar al Wifi anterior que funcionava correctament. Fins que no es pugui connectar aquest Wifi es mantindrà l'estat. En la pantalla es mostrerà el missatge que la placa s'està reconnectant al Wifi. Quan es rep el factor de calibració s'estableix en el detector de corrent perquè posteriorment faci les lectures correctes. En aquest estat el led de la placa segueix de color vermell. Es un estat important perquè es tracta de la primera comunicació amb el servidor i aquest inicia el sistema de threads de timer de comunicació per controlar l'enviament de dades continu (s'explicarà en l'apartat de control intel·ligent). En cas que hi hagi un problema amb el Wifi anirà a l'estat d'emergència on informarà del problema per la pantalla, intentarà reconnectar-se i quan ho hagi fet tornarà a l'estat anterior.



Figura 8: Error en el servidor. Led blau i estat d'emergència

El següent estat és l'**Estat2**. En aquest estat es consulta constantment si hi ha alguna reserva disponible per aquella màquina. En la pantalla apareix que la màquina esta apagada i no hi han reserves amb el led de color vermell. En el moment que es rep el missatge "*nextReserva*" es canvia d'estat i es guarda el valor de l'identificador de targeta d'usuari que ha de pasar per la reserva que li envia el servidor. En cas d'error Wifi o de servidor anirà a l'estat d'emergència seguint el procediment anterir. D'igual manera si es canvia el Wifi i el servidor li notifica farà el procediment anteriorment descrit.

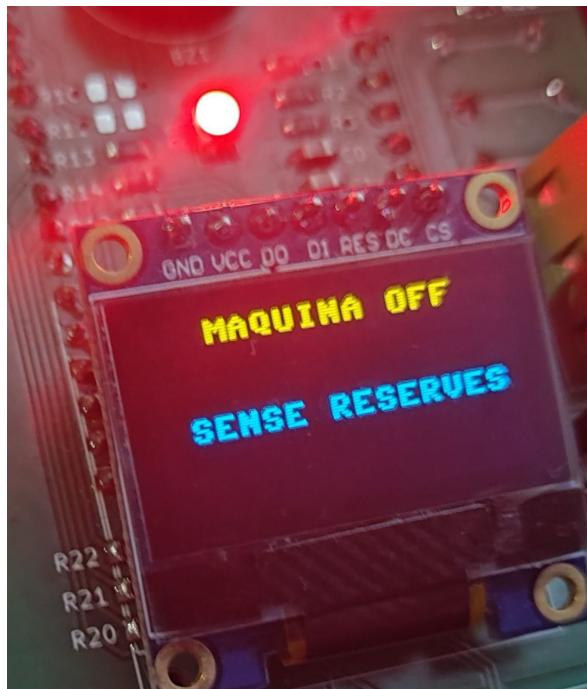


Figura 9: Sense reserves amb led vermell. Estat 2

Un cop el servidor ha informat que hi ha reserva i del l'ID de la targeta NFC que ha de pasar es pasa a l'**Estat3**. En aquest estat es mostra per pantalla que s'està esperant la targeta d'un usuari perquè hi ha reserva. S'activa el lector NFC per llegir targetes mentre que es manté un comunicació amb el servidor per veure si la reserva continua activa. Si la reserva no continua activa perquè ha passat un temps de 10 minuts i no ha vingut l'usuari o perquè s'hagi cancel·lat la reserva el servidor informarà amb el missatge "*noReserva*" que produirà el canvi d'estat a l'estat anterior de consulta de reserves. En aquest estat també poden haver emergències de servidor o de Wifi i es contemplen com en els cassos anterior. També es pot canviar de Wifi seguint el procediment anterior. La transició d'estat es produceix quan es passa la targeta i es correcte. En aquest cas, s'informa per la pantalla i es canvia l'estat. En cas que la targeta llegida sigui incorrecte també s'informa. Cada cop que es pasa una targeta el brunzidor fa un to suau i curt. El led de la placa continua de color vermell.



Figura 10: Esperant targeta. Led vermell



Figura 11: Usuari correcte. Led vermell



Figura 12: Usuari incorrecte. Led vermell

El següent estat és **Estat4**. En aquest estat s’informa al servidor que l’usuari de la reserva acaba de passar la targeta i per tant la seva hora d’entrada. El canvi d’estat es produueix quan el servidor li envia el missatge *OK* referenciant que tot es correcte. Mentre no li envii aquest missatge l’autòmat segueix en aquest estat. També contempla les emergències i el canvi de Wifi tal i com s’ha explicat anteriorment. Un cop l’usuari ha passat la targeta i el servidor es conscient es pasa a l’estat **Estat5** on prèviament s’encén la màquina amb el relé i es posa el led de la placa de color verd. Per la pantalla s’informa que la màquina està ON.

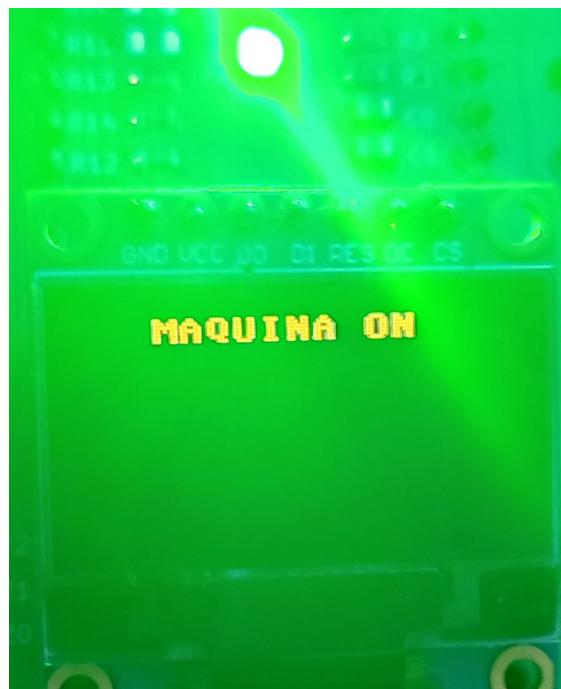


Figura 13: Màquina ON. Led verd ON. L'usuari té reserva i és l'adequat



Figura 14: Màquina ON

Un cop la màquina s'ha iniciat es pasa a l'estat **Estat5** on s'envia constantment els valors de consum i estat de la màquina. La màquina es troba en aquest estat mentre estigui en el temps de reserva o no s'hagi passat la targeta per sortir. En el moment que es passa la targeta s'informa al servidor i es canvia l'estat. D'altra banda si el servidor li envia el missatge

"apaga" canvia al següent estat i es finalitza la reserva. En aquest estat la màquina canvia els valors de l'actuador segons el que li diu el servidor (ja que els administradors del Techlab desde el website poden canviar-lo). D'altra banda, es contemplen errors de wifi i servidor com s'ha comentat anteriorment. En el moment que es passa la targeta per sortir s'informa al servidor i es mostra per la pantalla. Quan es finalitza una reserva ja sigui passant la targeta de l'usuari correcte o enviant un missatge desde el servidor es mostra el missatge de sortint. Un aspecte important a comentar en aquest estat es que vam fer una implementació extra de manera que el sistema de control avisés quan quedessin 5 minuts per acabar la reserva. Quan es dona aquest cas el servidor envia un missatge i la pantalla mostra que queda poc temps de reserva i el brunzidor fa un so suau informant a l'usuari.



Figura 15: Sortint

Un cop s'ha finalitzat la reserva o s'ha passat la targeta per acabar es pasa a l'estat **Estat6**. En aquest estat s'apaga la màquina posant el relé a off i es restauren els flags. Es torna a l'estat de consultar reserves.

Com podem observar, es una màquina d'estat complexa i robusta que dona solucions a tot tipus de casos d'error (tant en el servidor com caiguda de Wifi, control de reserves, etc ...). La intenció es mantindre una comunicació constant amb el servidor perquè conegui l'estat de la màquina i poder localitzar averies gràcies al sistema de control que porta incorporat.

5 SOFTWARE EN EL HOSTING

El cervell de tot el nostre sistema s'allotja en el servidor Flask que es troba en un hosting d'internet amb IP pública i sistema operatiu Linux amb el següent nom de domini:[EPS21]

El servidor web és el cervell de totes les operacions ja que informa als dispositius de les accions que han de fer, recull dades i les interpreta. El servei web en el hosting es fa a través d'un dimoni del sistema operatiu de UNIX, apache. En el hosting hi han parts molts diferenciades i, per això s'explicaràn cadascuna en detall en les següents seccions. El servidor connecta amb la base de dades, el control intel·ligent de reserves, el mòdul per enviar correus a través d'STMP, etc.

5.1 SERVIDOR WEB

Està programat amb el framework *flask* que proporciona el servei d'un servidor http que per defecte s'executa en el port 5000. La part de *backend* està programada amb *python3* amb flask. S'utilitzen les llibreries que flask incopora per separar el servidor en blueprints i tenir una estructura de rutes més organitzada.

La elecció de separar-ho en blueprints va ser per tindre-ho més ordenat i no tindre un mòdul d'alt tamany de rutes i mètodes. D'aquesta manera podem diferenciar les rutes segons si son per els clients web, per els dispositius hardware o les aplicacions gràfiques. El servidor compta amb una *apirestful* per poder tractar i processar dades dels dispositius físics, aplicacions gràfiques i clients web. En aquesta *apirest* es diferencien les rutes segons el tipus de client. L'*apirest* en sí és un blueprint del servidor que s'importa en el procés principal (*init*). Permet la comunicació bidireccional en format *JSON* encapsulat en trames http sobre una capa TLS comunicant amb la base de dades. Els clients web interactuen amb ella per obtindre dades dinàmicament com l'aforament actual, gràfics de potència en temps real, estat d'una màquina, etc. Ho fan a través de peticions http amb *AJAX* contra les rutes del *apirest* que retornen la informació en format *JSON*. D'igual manera, tant les interfícies gràfiques com els dispositius físics interactuen fent peticions contra l'*apirest* que els hi envia la informació en format *JSON*.

El servidor manté les sessions actives dels clients un cop han fet el login en la pàgina d'entrar. D'aquesta manera si els clients surten de la web i tornen a entrar, la sessió continua activa durant un temps. Per fer-ho s'utilitzen les llibreries que ofereix *flask* enviant una cookie als clients. Per referenciar una sessió d'un usuari dintre del servidor es fa pel seu correu electrònic.

La part de *front end* està feta amb *HTML5*, *CSS3*, *JavaScript* i diversos frameworks com *Bootstrap* per donar estils al portal web, *chartjs* per fer gràfiques, *AJAX* per fer la pàgina web

dinàmica fent peticions contra l'api restful i es recullen les dades en format *JSON*. Els fitxers de HTML es troben com a templates i el servidor flask el que fa es renderitzar-los segons convingui amb les dades que obté de la base de dades.

Els clients web navegen per la pàgina i el que és fan són peticions *GET* contra el servidor porque lis proporcioni informació de la pàgina enquestió. El servidor agafa tota la informació necessària per la pàgina que demana l'usuari i gràcies al framework de *Jinja* renderitza els templates amb flask.

El següent esquema mostra un diagrama de dependències de mòduls al servidor:

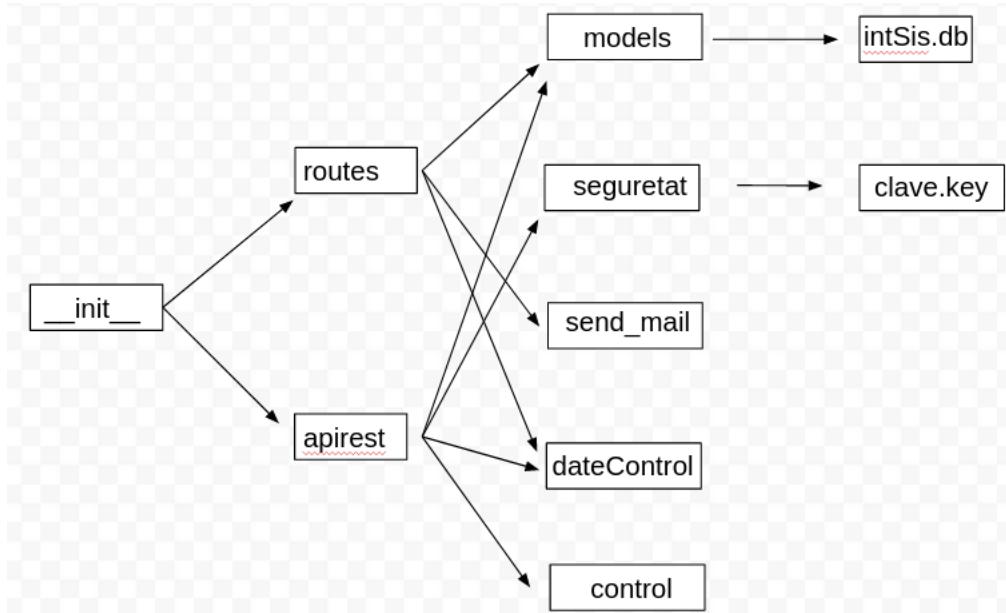


Figura 16: Diagrama de mòduls al servidor

Com es pot observar en la figura anterior, el procés principal és el mòdul `init` on es llença el servidor `flask` i s'importen els mòduls `routes.py` i `apirest.py`. Aquest dos fitxers son fitxers de rutes en el servidor.

El fitxer `routes` s'encarrega de renderitzar el components pel website i el control de les sessions. Interactua amb la base de dades a través del mòdul `models` que implementa totes les consultes i modificacions sobre la nostre base de dades, `intSis.db`.

El mòdul `models` està programat amb `python3` amb les llibreries de `sqlite3` de manera que podem hostejar sentències `SQL` en el nostre fitxer `python` per interactuar amb el fitxer binari de base de dades. El fitxer `models` es troba en el mateix directori de flask i per poder accedir a la base de dades en local s'ha de posar correctament el path de la base de dades (director `DATABASE` dintre de flask del computador enquestió).

Un altre funcionalitat important del fitxer de `routes` és l'enviament de correus electrònics

via *SMTP* per avisar de que s'ha realitzat una reserva, s'ha cancel·lat, recuperar la contrasenya i rebre el formulari de contacte. Envia correus via *SMTP* gràcies al mòdul *send_mail*.

També s'utilitza el mòdul *dateControl* que serveix per a controlar les hores en les reserves (temps mínim i màxim) i per obtindre el temps que queda per avisar a l'usuari que la seva reserva s'està acabant.

D'altra banda està l'apirest. Aquest mòdul implementa un apirestful com a un blueprint de servidor que permet la gestió i tractament de dades dels clients web per dinamitzar les seves sollicituts, per guardar els valors i proporcionar-los a les aplicacions gràfiques i per interactuar amb els dispositius físics.

Està dividida en tres parts: hardware, part web i aplicació gràfica. Es defineixen una sèrie de rutes amb diferents mètodes que sempre retornen les dades en format *JSON* encapsulat sobre una trama http sobre una capa TLS. L'apirest també contacta amb la base de dades per interactuar amb els dispositius per exemple informant que s'ha iniciat una màquina. Una part molt important de l'apirest es el control intel·ligent que porta incorporada per controlar emergències i avisos per als usuaris (s'explicarà en l'apartat de control intel·ligent).

Pel que fa la part web, es divideix en diferents rutes amb la majoria de mètodes *GET* i *PUT*. Un exemple de mètode *PUT* és per canviar l'estat de l'actuador d'una màquina concreta. El que farà serà modificar el valor de la base de dades. D'igual manera per el canvi de Wifi desde la Web, informar d'una averia o que l'averia ha sigut reparada. Tot i així la majoria de mètodes són de tipus *GET* on el client web gràcies a Javascript i *AJAX* interactuen amb l'apirest per consultar l'estat d'una màquina, l'aforament actual, la potència de les màquines, etc. Totes aquestes consultes són realitzades cada cert temps amb Javascript per fer una web dinàmica. L'apirest consulta els valors de la base de dades a través del mòdul *models* i retorna en format *JSON* una resposta pels clients web.

D'igual manera passa amb els dispositius físics i les aplicacions gràfiques que comuniquen amb el servidor les ordres que han de fer s'envien en *JSON* encapsulat sobre trames http sobre una capa TLS de xifrat.

El servidor incorpora un sistema de control de Wifi de manera que si una màquina s'intenta connectar a una xarxa i és un fracàs recupera el Wifi anterior que funcionava correctament. Si la connexió al punt d'accés és un èxit guarda aquells valors a la base de dades.

5.2 BASE DE DADES

Aquest projecte utilitza una base de dades relacional utilitzant SQL i específicament, utilitzen el framework Sqlite3 per Python. La bbdd està formada per set taules que defineixen les característiques del sistema. La base de dades segueix el següent diagrama E/R:

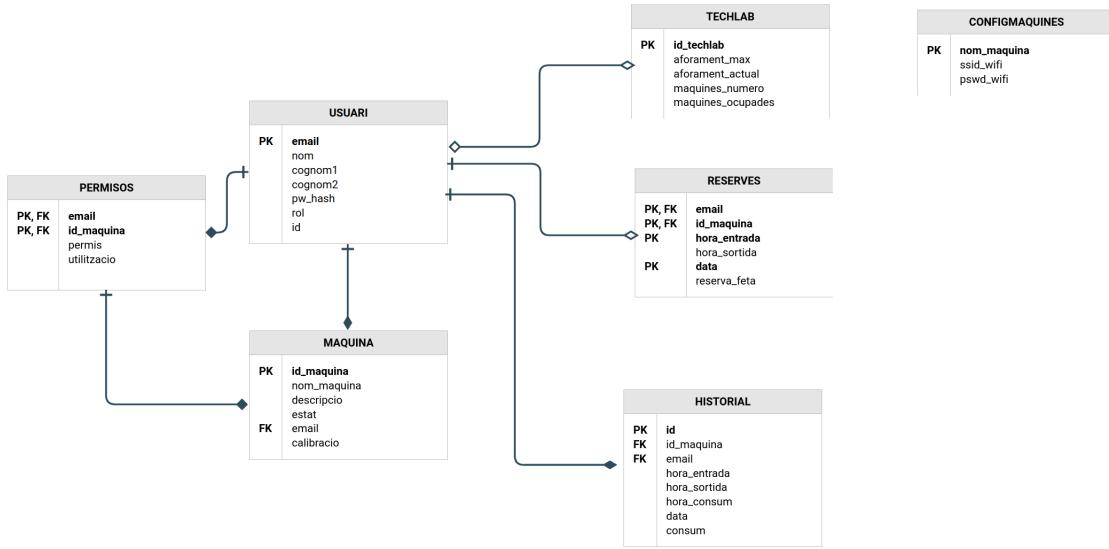


Figura 17: Model E/R de la nostre base de dades

Primer de tot, tenim una taula on guardem tota la informació dels usuaris registrats al Techlab. En aquesta tenim dos grans grups, els usuaris estàndard i els administradors del sistema. Dependent del tipus de rol, la interfície web ofereix unes característiques o permisos diferents. A continuació tenim una taula on es defineixen les màquines. A part de les característiques d'aquesta, també hi ha un camp per saber l'estat actual i qui l'està fent servir. Perquè un usuari pugui fer ús d'una màquina ha de tenir permís i utilització sobre aquesta. Aquesta informació es guarda a la taula *Permisos* i serveix per prevenir i arreglar problemes o avaries en una màquina.

Per poder utilitzar una màquina, els usuaris l'han d'haver reservat prèviament. En la taula *reserves* es guarda tota la informació quan es realitza una reserva. Per poder tenir un registre dels usuaris i quines màquines han utilitzat, la taula *Historial* guarda tota la informació per poder-la consultar en un futur. En el cas que l'usuari no tingui el rol d'administrador, només es podrà visualitzar informació del mateix usuari.

La taula Techlab té un control sobre les accions dels usuaris i l'estat de les màquines. Finalment, la taula *configmaquines* s'utilitza per configurar el ESP perquè es pugui connectar al punt d'accés.

5.3 WEBSITE

En aquest apartat procedirem a explicar el nostre website i ho farem a partir d'imatges del funcionament dels diferents templates realitzats i explicant les funcions que realitzen per darrere les possibles tecnologies com puguin ser JavaScript, Jinja o la BBDD i amb estructura de HTML5, CSS3 amb el framework de Bootstrap [W3S],etc. Aquest website permet l'entrada de qualsevol persona per poder conèixer l'equip o per preguntar informació, però requereix un login que diferencia entre rol d'usuari normal o rol d'administrador. L'administrador és capaç de gestionar totes les reserves(cancel·lar,modificar...) , canviar l'estat de les màquines, poder conèixer el consum en temps real d'aquestes, modificar permisos dels usuaris, modificar els propis usuaris(creació, eliminació...), visualitzar tots els historials de les màquines i usuaris. L'usuari normal, només pot gestionar les seves reserves i visualitzar l'històric seu, a més a més de poder veure l'aforament actual per si pot entrar o no, i les reserves de la última setmana. El nostre website es responsive de manera que es pot adaptar a qualsevol dispositiu.

5.3.1 Part sense login

La pàgina principal del nostre website presenta una estructura merament informativa, on la part superior ens redirigeix a la pàgina d'iniciar sessió a través de renderitzar el template d'entrar si es clica el botó de Reservar. A la següent imatge també podem apreciar que disposem d'un menú on podem accedir a veure el template de l'equip de treball, el de contactar amb els administradors, l'apartat de reserves i el d'iniciar sessió.



Figura 18: Menú superior pàgina inici

A la part inferior de la pàgina d'entrar, podem observar uns apartats informatius resumits de les funcionalitats del nostre TechLab. A més a més d'aquests miniapartats informatius, també incloem una redirecció a la pàgina de cancel·lar, passant primer també per iniciar sessió.

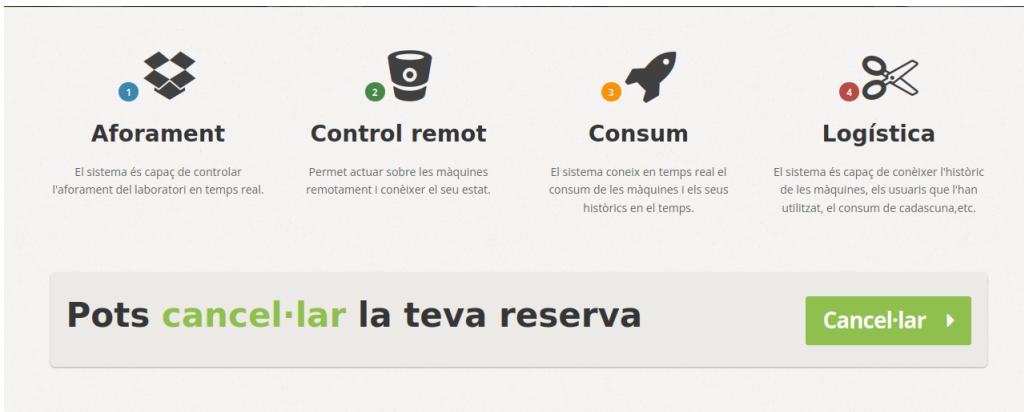


Figura 19: Menú inferior pàgina inici

Quan ens redireccionem a la pàgina d'entrar, veiem que mostra una imatge de la nostra universitat i tot just al costat tenim el quadre d'iniciar sessió on s'ha d'introduir correu d'usuari (per entrar com a usuari normal o com a admin segons el rol que presenti a la BBDD) i la contrasenya.

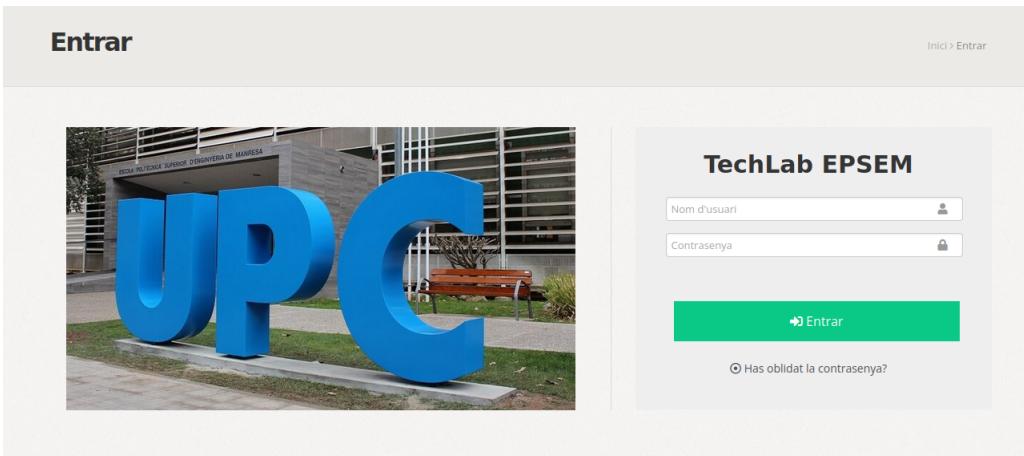


Figura 20: Pàgina entrar bàsica

Per el tractament d'errors com puguin ser correu o contrasenya escrits malament, correu no existent, o simplement intentar accedir sense complir tots els camps, t'enseanya que les credencials són incorrectes. Per fer això, ens basem en la interconnexió que ens proporciona la part dinàmica del site web a partir de les llibreries de *flask* amb la BBDD i el servidor, per realitzar les peticions http per evaluar els valors guardats a la nostre BBDD. Un altre apartat interessant que li dona robustesa al sistema final és la opció per si l'usuari ha olvidat la seva contrasenya. En aquest cas, introduint el seu correu electrònic s'envia un correu via *SMTP* amb un codi aleatori per poder accedir i canviar la contrasenya.



Figura 21: Error en la introducció de credencials

Un altre part que pot ser vista per tothom abans d'iniciar sessió és la que ensenya el nostre grup de treball amb les nostres fotos i les funcions de cadascú a l'interior del projecte.

Figura 22: Equip de treball del TechLab

El nostre grup es va informar sobre termes de privacitat i vam crear una declaració de política de privacitat del TechLab com a empresa que és una pàgina merament informativa.



Figura 23: Política de privacitat

El següent template a explicar, encara en la part sense sessió, és la pàgina web de contacte amb els administradors. Aquesta mostra un mapa de Google de la ubicació exacta de la nostra universitat, situada a Manresa. Després d'això trobem un formulari, que a partir de recopilar les dades en *flask*, envia un correu als administradors amb el camp del formulari que ha fet l'usuari. Un altre funció important d'aquesta pàgina es per informar als administradors del TechLab que es vol crear un nou usuari. Quan aquests rebin el correu, el crearàn i li enviarà un correu amb les seves credencials.

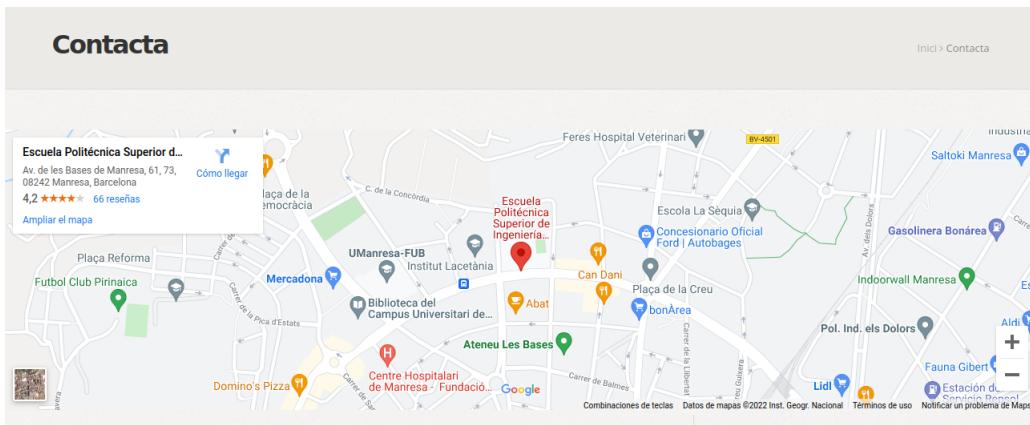


Figura 24: Mapa de la pàgina de contacte

Poseu-vos en contacte amb nosaltres emplenant el formulari de contacte següent

* Introdueix el nom complet * Introdueix el teu email

* Introdueix el tema de contacte

* El teu missatge aquí...

Envia

* Siusplau, ompli totes les entrades del formulari. Gràcies

Informació de contacte

Direcció :
Av. de les Bases de Manresa, 61-73,
Manresa

Telefon :
(+34) 93 877 72 00

Email :
techlabepsem21@gmail.com

Figura 25: Formulari de contacte amb els administradors

En el cas de deixar algun camp en blanc, seria impossible continuar degut a la característica dels inputs del formulari que són required i, per tant, tots han de presentar dades per poder ser enviat el correu.

5.3.2 Rol Administrador

Un cop hem introduït les nostres credencials a la pàgina d'entrar identificant-nos com a administradors segons els rols que estan a la base de dades, trobem que la pàgina principal d'aquest rol és un template que mostra l'aforament actual del TechLab, el consum de les diverses màquines en temps real, l'estat d'aquestes (si es troben enceses, apagades o en emergència, mostrant els diferents colors que es canvien a partir del JavaScript fent peticions contra l'apirest), les reserves d'avui i un menú lateral amb les diferents opcions a realitzar per l'administrador. A continuació es mostren algunes de les imatges quan hi ha connectada una màquina.



Figura 26: Estat de les màquines en temps real

Aquí podem observar que la primera màquina es troba encesa, és a dir, s'ha realitzat una reserva i a partir de la placa ha entrat la persona que havia d'entrar i s'ha engegat el relé per començar a calcular el consum. La màquina número 3, en color groc, vol dir que ha patit una averia i, com bé comentarem més endavant, s'ha modificat l'estat a partir de la pàgina de control de màquina. La resta de màquines estan apagades. Per consultar l'estat de les

màquines es fan peticions *AJAX* amb Javascript contra la ruta del apirest del servidor i recull les dades en format *JSON*. Per defecte, en el nostre Techlab tenim 5 màquines de moment i es poden afegir més com veurem a continuació. El gràfic d'estat de les màquines és dinàmic de manera que si s'esborra o s'afegeix una màquina es modifica. El nostre màxim de màquines que es poden veure en el gràfic d'estat és de 10 (2 files) ja que vam pensar que si hi havien moltes màquines ocuparia molt espai a la pàgina. Igualment en l'apartat màquines es mostren totes les màquines actuals del TechLab. Els icones són clicables de manera que quan cliques sobre ells et redirigeixen a la pàgina de la màquina enquestió.

Seguint en el mateix template, trobem també un gràfic, creat a partir de les llibreries *Chartjs* de JavaScript, que mostra en temps real la potència que està consumint cada màquina del TechLab. El gràfic dinàmic s'actualitza cada 5 segons fent peticions al servidor amb *AJAX* sobre l'historic de les 15 ultimes mostres de totes les màquines. Les màquines tenen colors diferents i es generen de forma aleatòria de manera que si es creen màquines noves mai tindràn el mateix color. Podem diferenciar les màquines a través de la història que tenim, mostrant diferents colors pertanyents a cada màquina.



Figura 27: Potència en temps real de les màquines

Una funcionalitat més és l'aparició d'un gràfic circular (també a través de *Chartjs* de JavaScript), que mostra l'aforament actual vers el total possible del TechLab. Quan es realitza un hover t'indica el número exacte de persones que es troben a dins. El gràfic es dinàmic i segueix el mateix procediment anterior preguntant al servidor cada 5 segons.

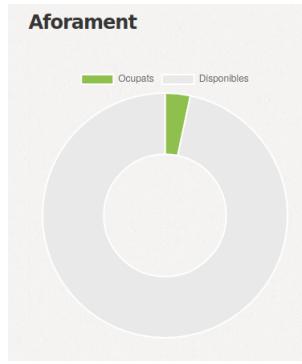


Figura 28: Aforament en temps real del TechLab

Després d'aquesta part base del rol d'administrador, anem a comentar el template de control de màquines. Com gairebé a totes les pàgines d'administrador, és una barreja entre pàgina informativa i interactiva. Aquí podem apreciar també la potència en temps real com a l'inici d'admin, però a més a més, ens dóna molta informació. A la part informativa trobem l'ID de la màquina, el nom, l'estat d'aquesta, si hi ha reserva o no, el SSID del Wifi al que està connectat aquella placa, i un indicador de màquina avariada, tot això captant les dades a través de peticions contra l'apirest del servidor. La part interactiva es basa en poder canviar l'estat de la màquina (ON/OFF), finalitzar reserva, consultar l'historial a través de l'acció en JavaScript de clicar un botó i, per últim, dos botons que accedeixen a dos pop-ups (un per modificar els paràmetres de configuració de xarxa i l'altre per modificar l'estat de la màquina si s'ha de posar en averia o si està arreglat).



Figura 29: Menú màquina

El primer pop-up, com hem comentat, serveix per poder modificar la configuració de xarxa de la màquina (SSID i password). D'aquesta manera fem que el sistema sigui escalable i es pugui posar en diferents lloc amb connexió Wifi. Les dades es recullen amb Javascript i

s'envíen amb un *PUT* contra el servidor que modificarà la taula de la base de dades de Wifi d'aquella màquina.



Figura 30: Pop-up de configuració de xarxa

El segon serveix per poder modificar l'estat amb dos botons que canvien a emergència o a normal segons si presenta una anomalia o no. Aquesta implementació serveix per si s'avaria una màquina posar-la en emergència de manera que no es puguin fer reserves d'aquella màquina informant que està averiada. Si la màquina s'arregla l'averia podem recuperar l'estat. Un exemple clar per aquesta aplicació apart de l'anterior es quan hi ha una tallada de llum o caiguda de Wifi, el servidor a través de threads de timer (s'explicarà en control intel·ligent) localitzarà l'averia i posarà la màquina en emergència mostrant el triangle d'emergència de la pàgina de color taronja. Si es soluciona el problema, a través d'aquest pop-up podrem recuperar l'estat anterior i el triangle d'emergència es posarà de color gris.



Figura 31: Pop-up de canvi d'estat

Per tractar la gestió de reserves, podem accedir a l'apartat d'introduïr una reserva on ens demana el nostre correu d'usuari, el dia, la hora de entrada i sortida i per últim, la màquina que volem reservar. Aquí veurem com surt un missatge d'error informant-nos que no existeix la màquina en cas de què no estigui a la base de dades. El nostre sistema permet reserves de 08:00 a 20:00 per tot el tipus d'usuari. El temps mínim de reserva ha de ser d'una hora tant per l'administrador com per l'usuari i màxim de 2 hores per l'usuari i més de 2 hores per l'administrador. L'administrador pot fer més d'una reserva al dia mentre que un usuari normal només una màxima de 2 hores i si té permisos sobre la màquina. En cas que un usuari necessiti un dia una màquina més de 2 hores, enviarà un correu a través de la pàgina de contacte informant de la seva situació i els administradors podràn fer la reserva enquestió. Tots els requeriments estan gestionats com a errors també (temps mínim, màxim de reserves per l'usuari, permisos, etc).

Figura 32: Error de no existència d'una màquina

Un altre cas d'error és en el cas que la màquina a reservar estigui averiada (estat emergència a la base de dades), i per tant es mostra aquest error.

Figura 33: Error de màquina averiada

Presenta una altra funció bloquejant que es tracta de que si intentes reservar una màquina a un dia i hora concret i ja està reservada, no pots accedir-hi.

Figura 34: Error de màquina ja reservada

Una mica semblant és el comportament del template de cancel·lar reserva. En aquest cas, no importa qui la tingui reservada, només es selecciona màquina, dia i hora. Podem veure el funcionament tot seguit:

Màquina a cancelar:

dia dd / mm / aaaa

Hora -- : --

Eliminar reserva

Success! Reserva eliminada

Figura 35: Missatge de reserva cancel·lada

Màquina a cancelar:

dia dd / mm / aaaa

Hora -- : --

Eliminar reserva

Error! Cap usuari ha reservat aquestes hores.

Figura 36: Error de màquina ja reservada

El missatge és diferent segons si s'elimina una reserva existent, o què no coincideixen els paràmetres amb cap reserva actual.

Quan una reserva es realitza correctament, ja sigui en rol d'administrador o rol d'usuari, s'envia un correu via SMTP al correu d'usuari que s'hagi introduït a la reserva. Aquí podem veure un exemple arribat a un dels administradors

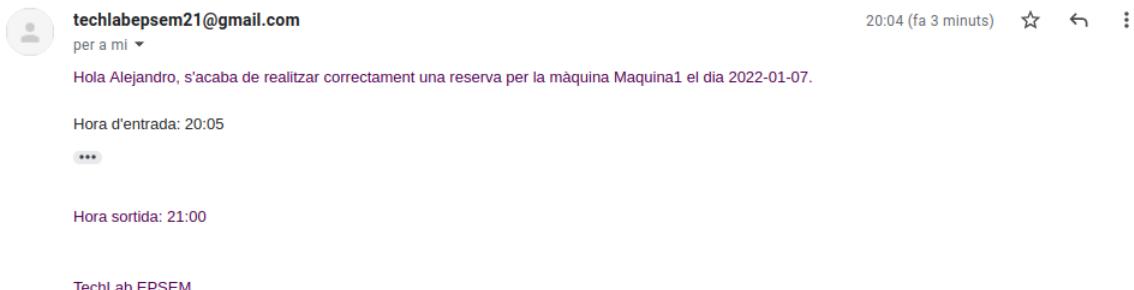


Figura 37: Correu rebut al Gmail de l'escola

En la pàgina d'historial d'usuari, podem buscar l'historial de qualsevol usuari que hagi reservat màquines indicant la data inicial i final del rang de dies que es vol estudiar, tot i estar loguejat com a altre administrador. Per fer-ho, s'omple el formulari i al recollir-lo en el servidor es renderitza el template amb la informació de la base de dades corresponent sobre el template amb *Jinja*. En la part inferior observem les reserves la última setmana.

Dia	Hora inici	Hora final	Usuari	Màquina
07-01-22	16:57	20:00	manuel.angel.roman@estudiantat.upc.edu	Maquina1

Figura 38: Historial d'un usuari concret

També disposem d'una pàgina per buscar l'historial d'una màquina d'un dia a un altre seguint el mateix procediment anterior com observem en la següent imatge:

Dia	Hora inici	Hora final	Usuari	Potència	Màquina
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4699094	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4655824	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4583804	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4628221	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4695759	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4758588	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4742199	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.466876	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4688581	Maquina1
07-01-22	19:38	22:00	alejandro.prieto@estudiantat.upc.edu	0.4693924	Maquina1

Figura 39: Historial d'una màquina concreta

Per poder-nos informar de les nostres dades d'usuari, hem de redirigir-nos a la pàgina de perfil, on ens mostra el nostre nom, mail i rol dins el website. També disposa de dos inputs per poder modificar en la base de dades el nostre mail i contrasenya i aquí podem veure un missatge d'avís que s'ha completat correctament el canvi de contrasenya.

The screenshot shows a 'Personal Information' form. It displays user details: Manuel angel Roman Ramos,manuel.angel.roman@estudiantat.upc.edu, and Administrador. There are fields for 'Nova contrasenya' (New password) and 'Nou email' (New email). A green button labeled 'Efectua canvis' (Update changes) is present. Below the form, a yellow success message reads: 'Success! Canvis efectuats amb èxit.' (Changes made successfully).

Figura 40: Missatge de contrasenya canviada

En el cas de deixar els dos camps en blanc i s'acció el botó, et surtirà un missatge per introduir dades, degut a que un o l'altre ha d'haver-hi alguna dada.

The screenshot shows a form with a field labeled 'Nou email' (New email) and a green 'Efectua canvis' (Update changes) button. Below the form, a yellow error message reads: 'Error! Introduix dades.' (Error! Enter data).

Figura 41: Error. Introduir dades

Un altre apartat important com a administrador, és que es pot modificar qualsevol permís de qualsevol usuari, per tant si es vol eliminar un permís concret d'una persona en la base de dades, hem d'introduir el seu correu identificatiu d'igual manera que si volem afegir permisos. En el cas que es vulguin eliminar permisos d'un usuari que no té s'informarà que no té dades d'igual manera que si es volen afegir permisos d'un usuari que els té tot. Les dades es recullen a través de flask segons la opció afegir o eliminar i es consulta l'usuari en qüestió en la base de dades. Un cop obté les dades renderitza el template amb *Jinja* i les dades corresponents.

The screenshot shows a 'Afegir/Eliminar permisos' (Add/Delete permissions) page. It lists users: alejandro.prieto@estudiantat.upc. There are 'Afegir' (Add) and 'Eliminar' (Delete) buttons. A table lists machines: Maquina1, Osciloscopi, Tester, Pc, Pantalla, each with 'Permis' (Permissions) and 'Utilització' (Usage) columns containing checkboxes. A green 'Modificar' (Modify) button is at the bottom.

Figura 42: Eliminar permisos

Per últim, també tenim disponible les opcions d'afegir/eliminar usuaris i màquines. Primer la gestió d'usuaris:

The screenshot shows two stacked sections of a web form. The top section, titled 'Informació personal', contains fields for 'Nom i cognoms' (Name and Surname), 'Email' (with value 'alejandro.prieto@estudiantat.upc'), 'Contrasenya' (Password), 'Repetir Contrasenya' (Repeat Password), 'Rol' (Role), and 'Codi Tarjeta' (Card Code). A green 'Crear usuari' (Create User) button is at the bottom. The bottom section, titled 'Eliminar usuari', contains a 'Email' field and a green 'Eliminar' (Delete) button.

Figura 43: Afegir/eliminar usuaris

Aquests dos apartats també gestionen tots els errors, com ja pugui ser si el formulari és invàlid perquè ja existeix el mail d'usuari, si no existeix usuari a eliminar o si hi ha algun paràmetre que no coincideix amb les possibilitats. Un cop es crea un usuari es donen permisos automàticament segons el rol que tingui. Si es un usuari normal no té permisos sobre ninguna màquina i si es administrador té tots els permisos en totes les màquines. D'igual manera, quan es crea una màquina s'assignen tots els permisos als administradors mentre que els usuaris no en tenen.

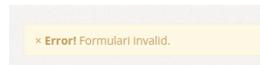


Figura 44: Error de formulari invàlid



Figura 45: Error d'usuari inexistent

Per últim, la gestió de màquines:

Figura 46: Afegir/eliminar màquines

Presenta un comportament molt semblant a l'anterior i a l'eliminar màquina, surten els mateixos errors que si volem eliminar un usuari que no toca

5.3.3 Rol d'usuari

Després d'analitzar la part sense login i el rol d'administrador, per acabar queda parlar del rol d'usuari sense permisos. Com podrem veure, presenta la mateixa estructura que en administrador, però només tenim la opció de reservar/cancel·lar i veure aquestes reserves, a més a més de poder saber l'aforament i visualitzar la teva pròpia informació. Igual que l'administrador té una pàgina de perfil per canviar dades personals com la contrasenya.

Per poder accedir a reservar qualsevol màquina, principalment el usuari normal no té permisos per accedir-hi així doncs han de ser modificats prèviament comunicant amb els administradors del TechLab per la pàgina de contacte.

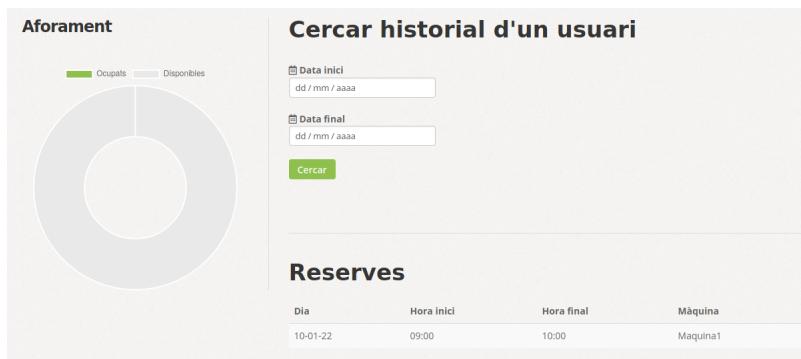


Figura 47: Historial d'un rang de reserves del propi usuari

A l'inici d'usuari també es pot visualitzar les reserves d'aquesta setmana degut a què un usuari té limitades les reserves a mínim 1h i màxim 2h per dia seguides.

5.4 CONTROL INTEL·LIGENT

Una part fonamental del sistema és el control intel·ligent que aquest incorpora. Un dels nostres objectius era fer reserves i si pasava un cert temps que no hi havia ningú usuari, cancel·lar la reserva de manera automàtica així com cancel·lar la reserva si l'usuari no passa la targeta al sortir i s'expira el seu temps d'ús. Per altre part, el control de la comunicació entre dispositius és un altre factor important. Una funcionalitat extra que hem afegit en aquest mòdul és l'avís quan queda poc temps per finalitzar la reserva per informar a l'usuari.

En aquesta part controlarem aquests aspectes. Per fer-ho de manera automàtica creem un servei que s'executa en el moment d'inicialitzar el servidor i actua paral·lelament a ell per no saturar-lo. Aquest servei interacciona amb la base de dades i amb l'apirest del servidor mitjançant threads de timer. El mòdul encarregat de fer aquesta feina és el **control.py**. El mòdul està programat en python dintre del servidor i s'executa paral·lelament al procés principal gràcies a threads.

El mòdul està estructurat en classes per així tindre un control més exhaustiu, robust i per les facilitats que aquestes incorporen. Per altre part, és més organitzat i fàcil d'accendir a cada thread. La classe *Scheduler* és el motor del sistema de threads. Aquesta classe s'encarrega de llençar threads de timer que executen una funcionalitat (funció handler). Quan es dona la interrupció de software de timer s'executa aquesta funció en el thread corresponent.

Per portar el control del sistema tant de control de reserves, timeout d'emergència amb la placa, timeout d'emergència de comunicació s'utilitza en el apirest una instància de la classe *control* que gestiona tots els cassos anteriors. Aquesta classe consulta cada segon les reserves disponibles i si hi ha alguna disponible on no s'ha fet el control d'aquesta es llença el thread de timeout de final de reserva i s'incorpora al diccionari de threads de reserves actives. En el moment que s'expira el valor del timer s'executa un handler que finalitza la reserva i decrementa l'aforament del Techlab.

En els atributs de la classe hi han diversos diccionaris on s'emmagatzemen el threads actius tant de control de reserves, emergència de timeout de comunicació i un similar a l'anterior per quan els usuaris estan utilitzant la placa. En el moment que es connecta la placa i es connecta amb el servidor en una ruta de l'apirest de hardware, es llença un thread de control d'emergència de comunicació i s'emmagatzema en un diccionari dintre d'una variable global de l'apirest. Aquesta variable es una instància de *control* on s'emmagatzemen tots els threads.

Cada cop que li arriba una petició al servidor de la placa es llença el thread de timeout de comunicació fins que arriba la següent petició on es cancel·la la anterior i es torna a llençar. En cas que el dispositiu no envii faci peticions al servidor s'executarà el handler de timeout i posarà la màquina en emergència (a la base de dades) de manera que a la web s'activarà

l'emergència per aquella màquina.

Per altre part, en el moment que un usuari està utilitzant una màquina es fa el mateix procediment que en l'anterior cas per controlar que es pasen correctament cada un cert temps els valors de consum i estat de la placa.

En el moment que un usuari comença la seva reserva, s'inicialitza un thread per avisar-li que queda poc temps per finalitzar la seva reserva (5 minuts) que ho veurà en la pantalla de la placa quan sigui el moment.

Aquest mòdul permet tindre un control exhaustiu de la comunicació i del control de reserves així com vam plantejar al contracte. En el cas que hi hagi una reserva i no entri cap usuari el sistema finalitzarà la reserva un cop acabi el temps mínim per entrar (10 minuts). Pensem que es un bon sistema ja que s'automatitza tot el servidor i s'inclouen funcionalitat per permetre que el Techlab pugui ser usat el màxim temps possible per el màxim nombre d'usuaris.

Per altre part, el control d'emergències es un aspecte que dona robustesa al sistema final ja que el sistema es capaç de localitzar emergències quan els dispositius no comuniquen amb el servidor de manera automàtica sense la necessitat que un usuari es doni compte que les dades no arriben realment.

Per altre part, la interacció amb la base de dades permet canviar les taules que aquesta incorpora i desde el portal web veure l'estat de la màquina en temps real tot i que tingui problemes de connexió amb el servidor (activant l'emergència de la màquina corresponent). Un altre aspecte important és la interfície d'usuari ja que el sistema avisarà a l'usuari que queda poc temps per finalitzar la seva reserva i ho farà de forma automàtica.

D'altra banda, el sistema decrementa l'aforament del Techlab automàticament sense la necessitat que l'usuari passi la targeta de sortida informant de la seva marxa.

Un altre factor important és la robustesa que li dona al sistema en cas de talls de llum, saturacions de bandes de freqüencies,etc. En alguns d'aquests casos la placa no enviarà informació al sistema de control intel·ligent i, aquest automàticament veurà que la màquina en questió està sofrint una averia.

6 CONCLUSIONS

Per finalitzar l'explicació del desenvolupament d'aquest projecte, volem mirar enrere i avaluar els objectius que ens vam decidir posar durant la fase d'organització del projecte. En termes generals, el nostre grup creiem que hem complert perfectament les expectatives que ens vam ficar, degut a què hem arribat a dissenyar un sistema completament funcional en tots els aspectes que es van tractar.

En termes de hardware, ens hem basat, com bé vam comentar a la primera acta entregada, en les llibreries de MicroPython dels diferents mòduls de la placa i el programa principal s'ha implementat com una màquina d'estats on pregunta al servidor en tot moment les necessitats de cada estat. Degut a això, vam decidir no implementar les cues de transmissió i recepció perquè no era necessari i més farragós estaravaluant les cues en tot moment. El que si s'ha de corroborar en termes de hardware és la implementació amb threads que es va comentar al principi, sobretot en la part de control intel·ligent del temps (sessions, esperes d'estat, reconfiguració, brunzidor, etc.), fent d'això un sistema molt més robust i segur. A més a més disposem de funcionalitat extres que no estaven a la primera idea i que no són estrictament necessàries en el projecte final, com una interfície gràfica en Tkinter de Python que pot canviar els paràmetres de configuració de xarxa del dispositiu.

En la part de la interfície web, també ens trobem bastant orgullosos del treball fet. Dispossem d'un website amb domini obert a tothom on s'han utilitzat totes les tecnologies necessàries per fer l'entorn molt més visualment agradable i accessible tant pels administradors com pels usuaris externs i adaptable a diferents dispositius. Com bé ens vam proposar, hem realitzat una interfície web amb totes les possibilitats proposades, com eren tota la gestió de les reserves (informació, eliminació i modificació), el possible control d'aforament i consum de les diferents màquines en temps real, diferència de permisos entre admin i user, entre d'altres.

Si evaluem la part de servidor, també ens hem basat en la primera idea, en dividir les peticions en diferents blueprints per la millor organització i accés al codi, tot i que també es van arribar a implementar per separat però creiem que no era tant factible. Disposem d'un api rest per fer de connexió directe entre el dispositiu, el portal web i la base de dades implementada i poder modificar aquesta.

Per últim, resumint, estem molt orgullosos del treball fet entre tots els membres de l'equip, que tot i haver-hi parts més farragoses o époques més difícils de gestionar el treball i les diferents opinions per fer qualsevol cosa, hem completat l'objectiu de realitzar aquest projecte i hem après molt en termes de treball en equip i creació d'un projecte desde pràcticament zero.

7 ANNEXOS

7.1 INTERFÍCIE GRÀFICA

Per tal de fer un sistema escalable i portable vam desenvolupar una aplicació d'escriptori gràfica per poder canviar el Wifi de les màquines apart del sistema que tenim en el website. L'aplicació va ser desenvolupada amb *Python3* amb les llibreries de *tkinter* que ofereixen un entorn gràfic. Per intercomunicar amb el servidor vam usar la llibreria *requests* contra el servidor https. A part, vam xifrar les dades del SSID del Wifi i la seva contrasenya amb una clau privada que comparteixen les aplicacions gràfiques i el servidor per codificar i decodificar dades. La següent imatge mostra l'aplicació gràfica:

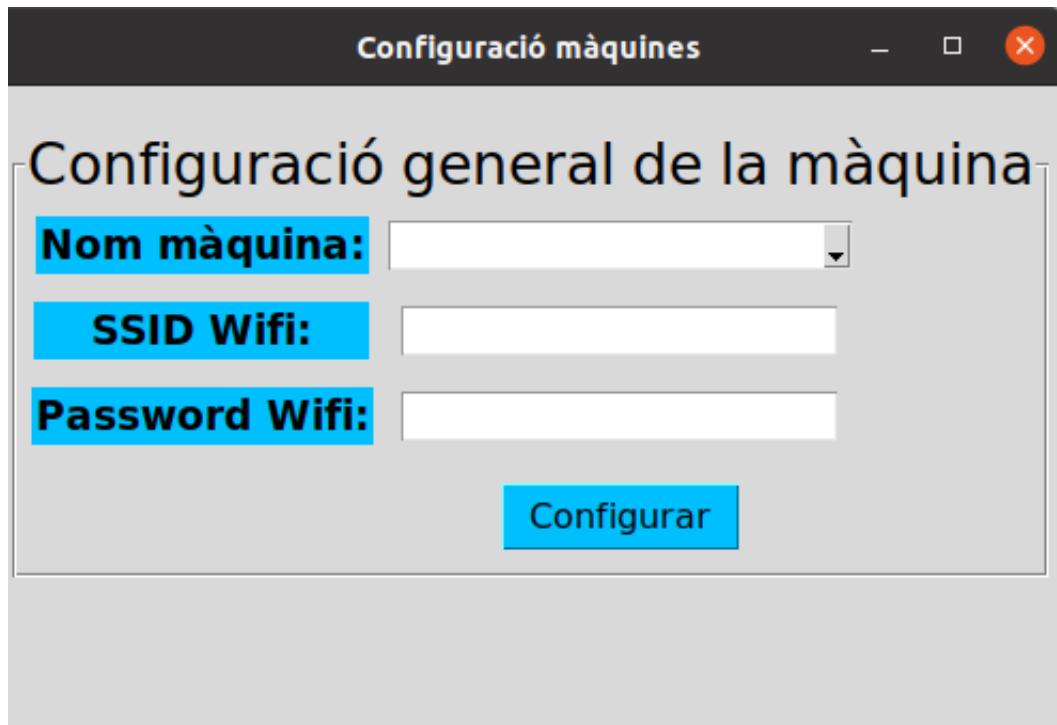


Figura 48: Aplicació gràfica

Un cop iniciada l'aplicació es fa una consulta al servidor de les màquines que té el TechLab i s'obté una resposta en *JSON*. Les màquines disponibles al TechLab són visibles per l'usuari de manera que pot canviar el Wifi de qualsevol. La següent imatge mostra un exemple:

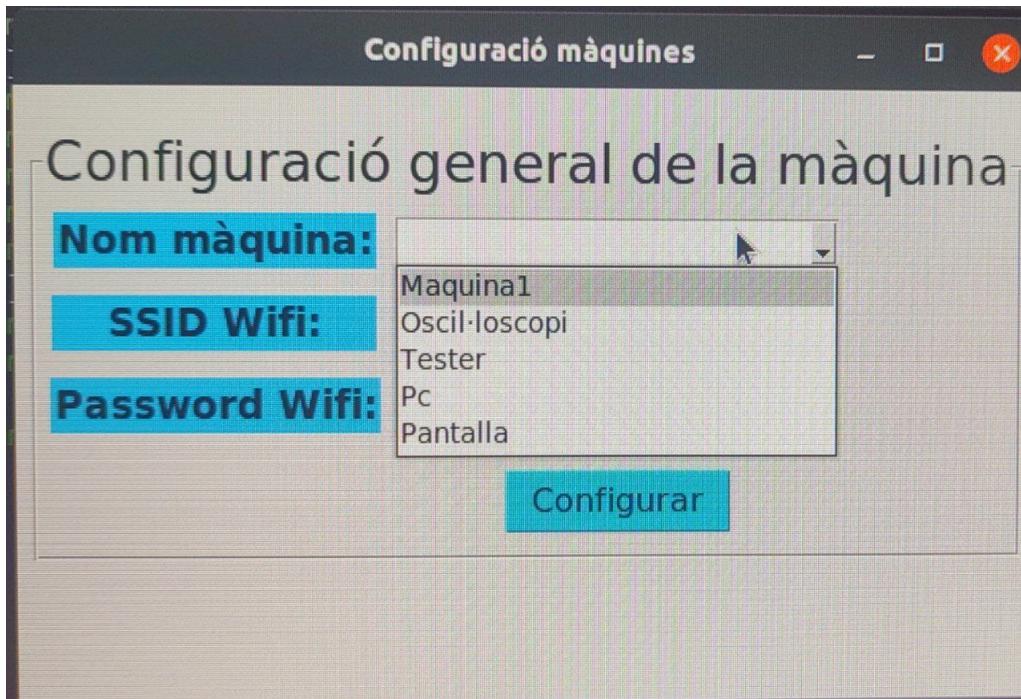


Figura 49: Màquines disponibles al TechLab en l'aplicació gràfica.

Un cop completats tots els camps, es xifren les dades amb la clau privada i s'envia la sol·licitud contra l'apirest d'aplicació gràfica del servidor. Aquesta canviarà els valors de la base de dades de la màquina en questió per quan li facin peticions es connecti al nou Wifi.

L'aplicació abans de ser iniciada comprova l'estat del servidor i en cas que no estigui operatiu informa a l'usuari amb un missatge d'error.

7.2 CONFIGURACIÓ A NIVELL LOCAL

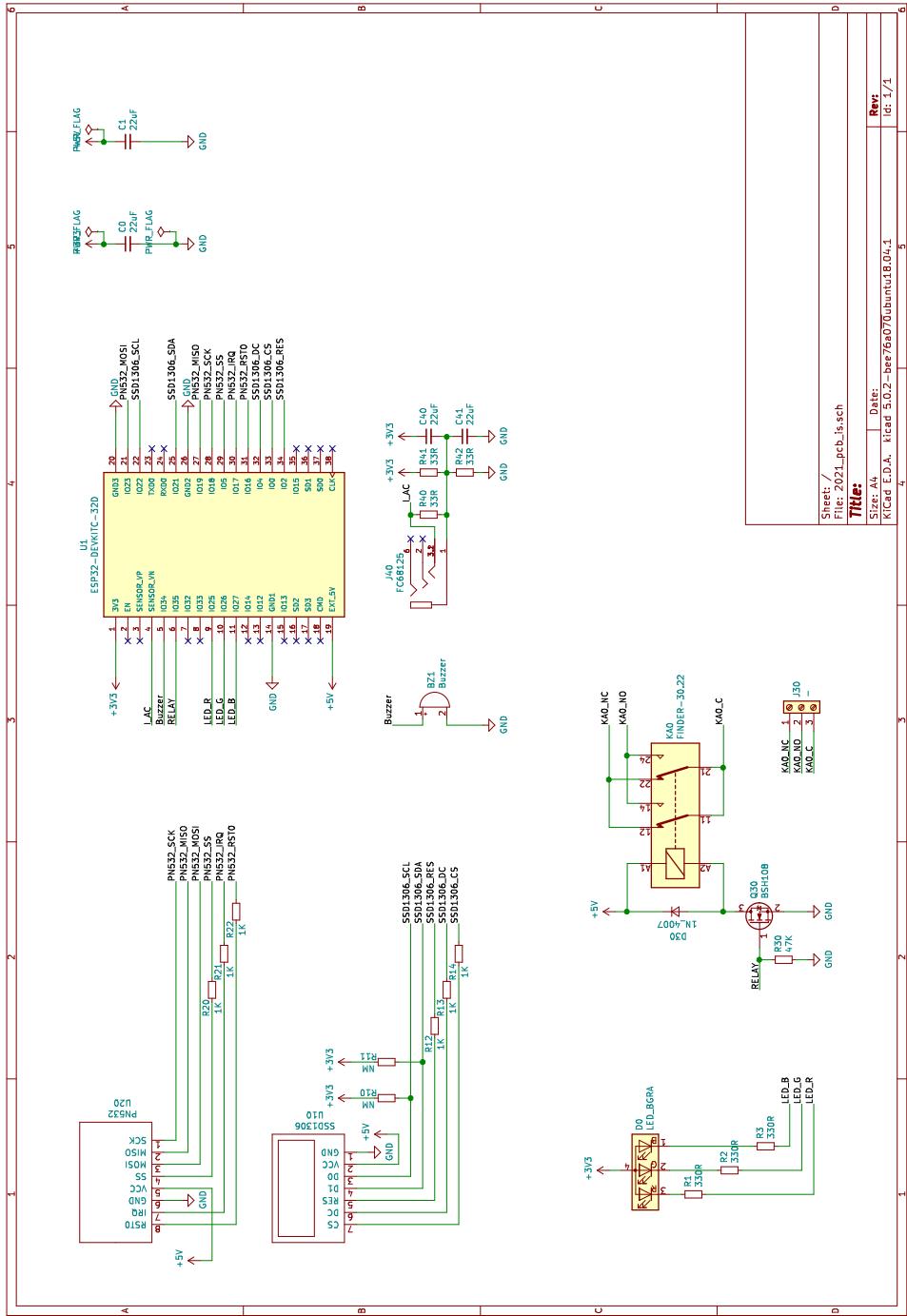
Per poder provar la placa en local, hem de seguir una sèrie de passos. Els passos són els següents:

1. Posar a la base de dades a quin wifi volem que és connectí la màquina i compilar la base de dades amb **python3 db.py**.
2. Posar al **models.py** la ruta de la base de dades del nostre ordinador.
3. Fem la comanda **iptables -I INPUT -p tcp --dport "port del flask" -j ACCEPT** per acceptar peticions http pel port 5000.
4. Executem el servidor "Flask" amb la comanda **python3 _init_.py**
5. **Flasehem** la placa amb el **Firmware de Micropython** amb el software **Thonny**.
6. Utilitzant el programa **Thonny**, modifiquem el fixer **main.py** indicant el wifi,el servidor flask local i l'ID de la màquina.
7. Carreguem els fitxers en la placa amb el programa **Thonny**.
8. Canviar el ServerName a l'aplicació gràfica.

7.3 PLACA

7.3.1 Esquemàtic del dispositiu

Figura 50: Esquema de la placa



7.3.2 Disseny 3D de la caixa

Per tal de complir amb el desenvolupament d'un sistema encastat vam elaborar una caixa protectora que seria fabricada amb una impresora 3D. La caixa està feta perquè sigui compatible de tamany amb els components de la *PCB*. La caixa va ser desenvolupada amb el software de *Freecad*.

Es tracta un disseny de dimensions reduïdes perquè ocipi el mínim tamany. La caixa té diversos forats perquè es vegi la pantalla i el led. També té un forat pel lector NFC. En aquest forat es colocaria un plàstic opac d'un material que no interferís el senyal del lector i permetés les lectures. D'altra banda, compta amb un forat inferior per connectar la placa a la seva alimentació i dos laterals per enxufar la màquina i connectar el JACK del sensor *SCT013*.

Les següents imatges mostren el nostre disseny:

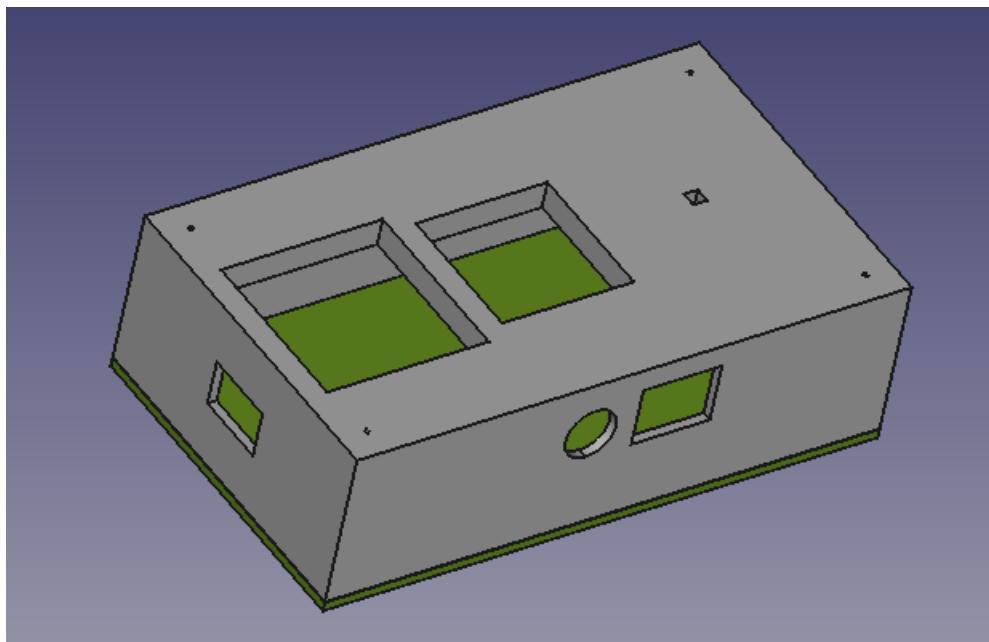


Figura 51: Esquema general caixa

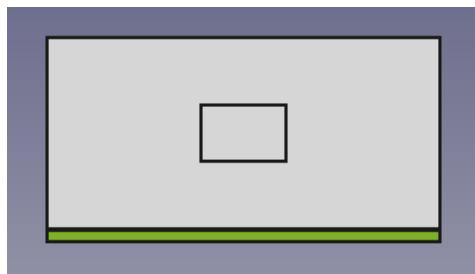


Figura 52: Forat connexió ESP

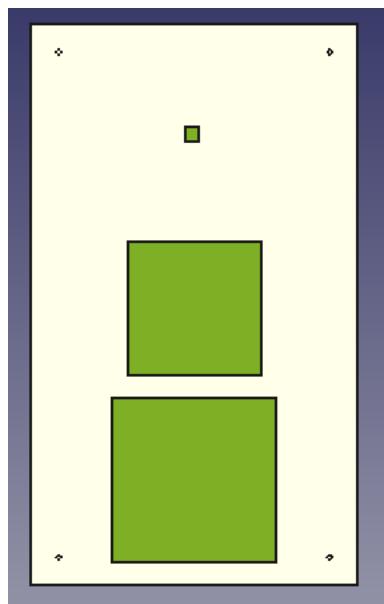


Figura 53: Forats NFC i Pantalla, LED

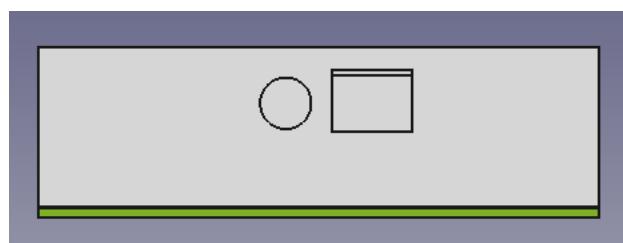


Figura 54: Forats JACK i Relé

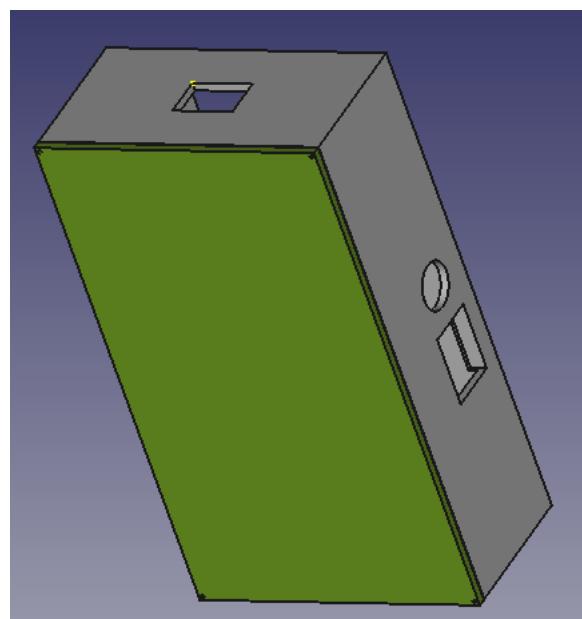


Figura 55: Tapa inferior per introduir placa

7.4 ORGANITZACIÓ DEL GRUP DE TREBALL

7.4.1 Calendari

Inicialment, vam plantejar que el disseny del front end s'encarregués el Manu i l'Alejandro durant el primer mes i el Jaume i l'Ivan s'encarreguessin de la base de dades i el back-end.

El període de desenvolupament del back-end era complicat de predir, però vam pensar que tardaríem un mes i mig. És a dir, vam planificar que a finals d'octubre o principis de novembre tindríem tant el front-end com el back-end acabat.

Per part del front-end, vam complir la planificació, ja que a l'octubre ja havíem acabat la majoria de desenvolupament. Tot i això, hem afegit i modificant certs aspectes del front-end al llarg de tot el projecte.

Per altra banda, el back-end ha sigut molt més llarg de desenvolupar del que ens pensàvem, ja que hem volgut implementar moltes funcionalitats. Vam planificar que acabaríem al novembre i hem acabat al desembre. L'endarreriment també és degut a certes problemàtiques que no vam preveure a l'inici del projecte.

A la planificació inicial, vam calcular que hauríem acabat el desenvolupament del hardware cap a mitjans de novembre. Aquest objectiu el vam complir parcialment, ja que a mitjans de novembre teníem la placa programada, però, com que no teníem la pàgina web pujada al servidor, no vam poder provar totes les seves funcionalitats fins a mitjans de desembre. Finalment, vam haver d'afegir el servidor proxy i vam acabar el desenvolupament de la placa a finals de desembre.

El servidor es va configurar a mesura que es desenvolupava el back-end de la pàgina web, és a dir, va configurar-se a mitjans de novembre. Inicialment, havíem plantejat utilitzar l'allotjament d'AWS, però finalment vam acabar per utilitzar GCC. Una vegada acabat el desenvolupament del back-end i del hardware, hem pujat la web al servidor per poder realitzar proves de funcionament.

Per acabar, la caixa ha sigut l'última cosa a dur a terme-se tal com s'havia planificat a començament de gener. Paral·lelament, també s'ha fet el testatge i la redacció de la documentació del projecte.

7.4.2 Repartiment de Tasques

El projecte el vam dividir en diferents parts i cada integrant del grup va elaborar la seva part. Cal dir que tothom va col·laborar en altres feines que no eren la seva responsabilitat total tot i així l'organització va ser la següent:

1. **Ivan Chamero.** Va desenvolupar la part de servidor tant de l'apirest com de la configuració del servidor en flask. També va desenvolupar funcions de la base de dades i es va encarregar de la interconnexió dinàmica amb Javascript del website. Va ajudar en el desenvolupament de Hardware sobretot en el desenvolupament del programa principal i les lectures de corrent.
2. **Manuel Ángel Román.** Va desenvolupar el front end de la interfície web i la interconnexió dinàmica amb Javascript per fer la pàgina dinàmica. Va desenvolupar el Hardware de la placa i va fer funcions per la base de dades, modificant-la per adaptar-la al hardware. Va desenvolupar el sistema de Control al servidor amb threads i va fer l'arquitectura de l'apirest. Va col·laborar en la interconnexió de les aplicacions gràfiques amb el servidor i va desenvolupar el servidor proxy.
3. **Alejandro Prieto.** Va desenvolupar el front end del website i les aplicacions gràfiques amb la seva interconnexió al servidor. Va desenvolupar funcions de hardware. Va desenvolupar la part de l'apirest per les aplicacions de hardware amb els respectius mètodes per la base de dades. Va desenvolupar el sistema de pop-ups al website. Va desenvolupar l'encapsulat de la placa en 3D i el sistema web amb interconnexió dinàmica per averies i averies solucionades.
4. **Jaume Serra.** Va desenvolupar la base de dades i l'estructura del servidor amb cookies, sessions i renderitzacions de templates. Va fer funcions per la base de dades i va configurar el hosting amb apache i certificats de seguretat TLS. Va elaborar el sistema d'enviament de correus. Va adaptar les plantilles de front per aprofitar-les amb Flask des de el servidor.

El grup es porta unes bones sensacions ja que sempre si teníem algun problema un altre company t'ajudava en la feina i, per tant, tots hem fet de tot.

Referències

- [EPS21] TechLab EPSEM. *TechLab*. <https://epsemtechlab.ddns.net/>. Accedit últim cop gener 2022. 2021.
- [Ope21] OpenProject. *Subversion Escriny*. <https://escriny.epsem.upc.edu/svn/isis-ivan-alejandro-manu>. Accedit últim cop gener 2022. 2021.
- [Boo] Bootstrap. *Bootstrap*. Ang. <https://getbootstrap.com/>.
- [Cat] Universitat Politècnica de Catalunya. *Atenea*. <https://atenea.upc.edu/>. Accedit últim cop gener 2022.
- [Dop] Radomir Dopieralski. *Basics*. Ang. <https://micropython-on-wemos-d1-mini.readthedocs.io/en/latest/basics.html#beepers>. Accedit últim cop novembre 2021.
- [Geoa] Damien P. George. *Analog to Digital Conversion*. Ang. <https://docs.micropython.org/en/latest/esp8266/tutorial/adc.html>. Accedit últim cop novembre 2021.
- [Geob] Damien P. George. *Getting started with MicroPython on the ESP32*. Ang. <https://docs.micropython.org/en/latest/esp32/tutorial/intro.html>. Accedit últim cop octubre 2021.
- [GIR] SERGIO ANDRÉS CASTAÑO GIRALDO. *LCD I2C Raspberry Pi Pico*. Cast. <https://controlautomaticoeducacion.com/micropython/lcd-i2c-raspberry-pi-pico/>. Accedit últim cop octubre 2021.
- [Git] GitHub. *GitHub*. Ang. <https://getbootstrap.com/>.
- [Ove] Stack Overflow. *Stack Overflow*. Ang. <https://github.com/>.
- [Pyta] Python3. *Flask*. Ang. <https://flask.palletsprojects.com/en/2.0.x/>. Accedit últim cop novembre 2021.
- [Pytb] Python3. *Requests: HTTP for Humans*. Ang. <https://docs.python-requests.org/en/latest/>. Accedit últim cop desembre 2021.
- [Pytc] Python3. *tkinter — Interface de Python para Tcl/Tk*. Ang. <https://docs.python.org/es/3/library/tkinter.html>. Accedit últim cop novembre 2021.
- [Pytd] Python3. *Tutorial sensor de corriente AC no invasivo SCT-013*. Ang. https://naylampmechatronics.com/blog/51_tutorial-sensor-de-corriente-ac-no-invasivo-sct-013.html. Accedit últim cop desembre 2021.
- [Ram] Manuel Ángel Román Ramos. *Video demostratiu*. Ang. <https://youtu.be/NNqtBiqqn8A>. Accedit últim cop gener 2022.
- [W3S] W3SCHOOLS. *W3Schools*. Ang. <https://www.w3schools.com/>.
- [WAR] WARLORD. *ESP32 and NFC over I2C*. Ang. <https://warlord0blog.wordpress.com/2021/10/09/esp32-and-nfc-over-i2c/>. Accedit últim cop novembre 2021.