

Desplegament del servei Web i definició de protocols

Nil Blanca
Roger Tarres
Javier Delgado
Josep Marches
Héctor Montesinos

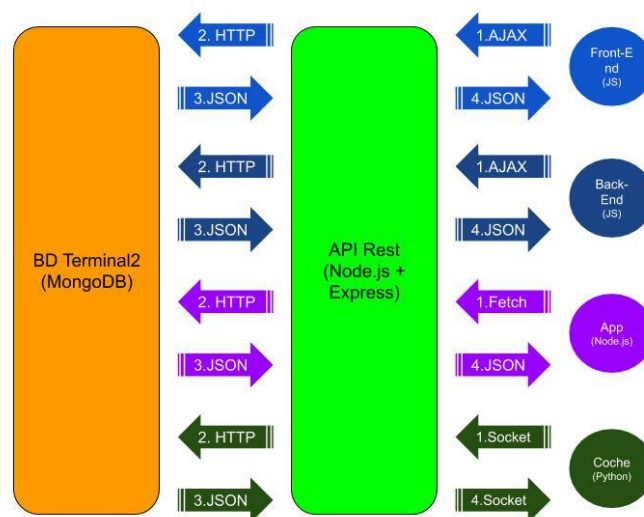
Índex...

Descripció general Sprint	2
Desplegament del servei web	13
Descripció general	13
Explicació software/hardware utilitzat	14
Manual d'usuari	15
Conclusions	15
Definició de protocols	16
Descripció general	16
Explicació/Justificació	16
Conclusions	16

Descripció general Sprint 3

//Descripció global de les tasques realitzades i l'organització del grup. Part del projecte realitzada, estructuració del treball...

En aquest tercer Sprint un dels increments ha sigut definir l'ús de protocols per a la connexió tant dels serveis dels quals som responsables, com els serveis dels altres grups amb els que estarem connectats:

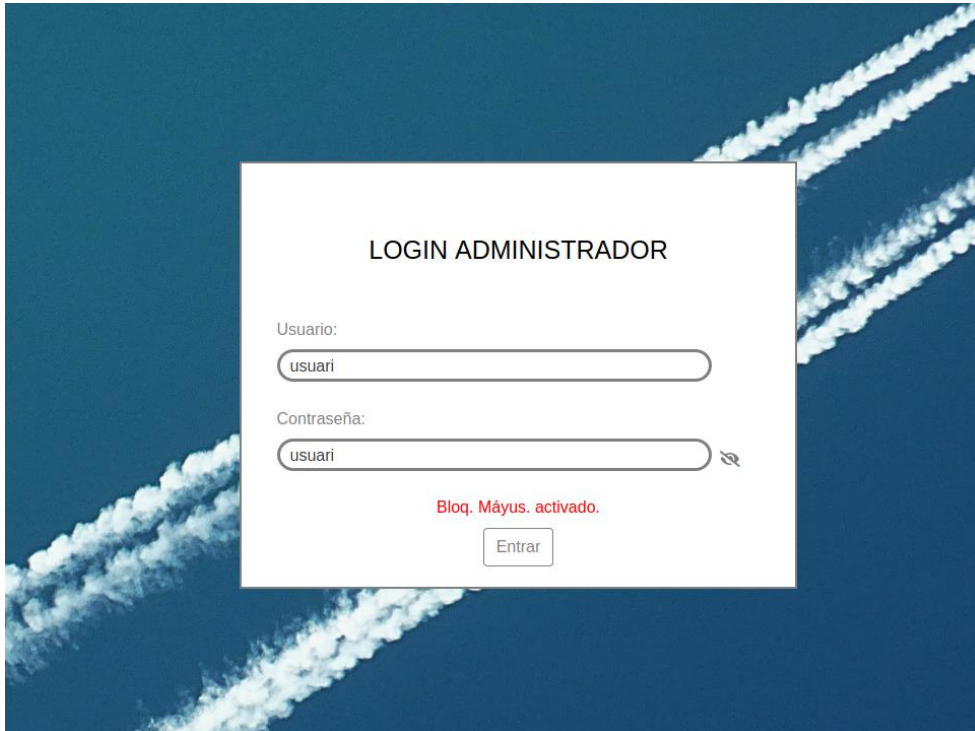


Les peticions de la API Rest a la base de dades seran HTTP (TCP) i les peticions de cada servei client a la API seran d'un tipus concret indicat al diagrama. En aquest cas els dos clients web fan peticions AJAX, l'aplicació mòbil fa peticions FETCH i el cotxe es connectarà per un WebSocket a la API mitjançant el mòdul socket.io de Node. Per l'aplicació mòbil i per al cotxe, els protocols estan definits però encara no implementats.

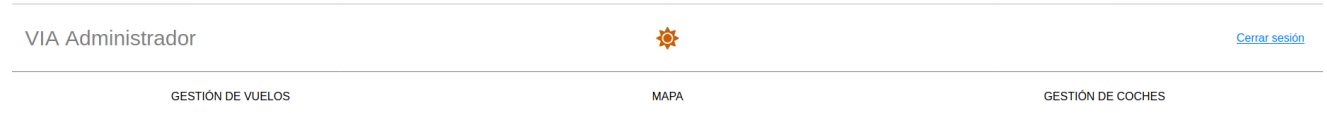
Parlem ara de l'increment de la web d'administrador.

En primer lloc, s'ha programat un Login mitjançant usuari i contrasenya, que fa una consulta a la base de dades. Si l'usuari existeix a la base de dades, pot accedir a la pàgina, en cas contrari mostra un missatge d'alerta. Aquest login fa una petició AJAX a la API i des de la API es fa una consulta a la base de dades buscant aquella dada.

A més a la pàgina de Login s'hi han inclòs funcionalitats noves com el canvi de fons, unificat amb el de la pàgina web d'usuaris, la detecció de majúscules, l'ocultació de la contrasenya i un ull per mostrar-la:

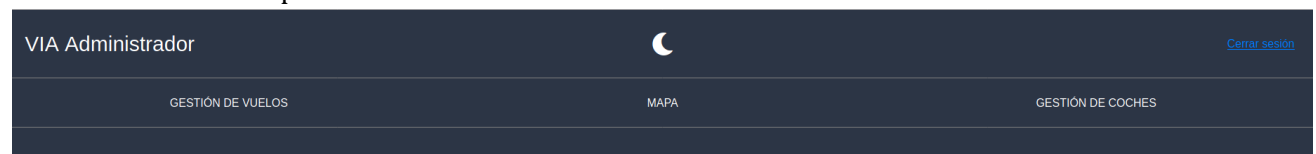
A screenshot of the 'LOGIN ADMINISTRADOR' form. The form is white and centered on a dark blue background with white diagonal streaks. It contains two input fields: 'Usuario:' with the text 'usuari' and 'Contraseña:' with the text 'usuari'. To the right of the password field is a small eye icon. Below the fields is a red message 'Bloq. Máys. activado.' and an 'Entrar' button.

Un cop s'accedeix a la pàgina amb unes credencials correctes, s'han unificat les tres pestanyes:



Ara, la gestió de vols, el mapa i la gestió de cotxes, son la mateixa pàgina i l'administrador es pot moure entre elles simplement clicant a sobre del nom, però no es carregarà una altra pàgina, simplement farà un lleuger moviment a dreta o esquerra.

L'Icona que es pot veure al mig de la imatge anterior, es altra nova implementació de mode nocturn, si l'administrador treballa de nit o simplement vol rebaixar la il·luminació, pot clicar a sobre i s'obscura la pantalla:



La primera part de la pestanya, gestió de vols, també està connectada amb la API i mostra els vols que estan entrats a la base de dades, encara que no amb totes les dades de la col·lecció *Flights* ja que seria molt enyorós veure una taula amb tantes dades:

GESTIÓN DE VUELOS	MAPA	GESTIÓN DE COCHES
-------------------	------	-------------------

Gestión de Vuelos

Vuelo	Aerolínea	Hora de salida	Hora de llegada
QA1274	FA3	01:05	12:30
VL206	FA2	19:05	20:30
VL207	FA1	11:05	11:00
VL211	FA41	10:05	10:30

[Modificar](#) [Añadir](#)

No obstant, com que també s'ha implementat la opció d'afegir nous vols des de la pàgina, quan l'administrador clica sobre el botó afegir, si que se li demanen tots els camps de la col·lecció *Flights*, ja que son imprescindibles a la base de dades:

Gestión de Vuelos

Vuelo	Aerolínea	Hora de salida	Hora de llegada
QA1274	FA3	01:05	12:30
VL206	FA2	19:05	20:30
VL207	FA1	11:05	11:00
VL211	FA41	10:05	10:30

Nombre de vuelo	<input type="text" value="p.ej VL054, FR675..."/>
Aerolínea	<input type="text" value="p.ej Vueling, SkyAirlines..."/>
Origen	<input type="text" value="p.ej VilanovaT1, VilanovaT2..."/>
Destino	<input type="text" value="p.ej Dinamarca, Francia..."/>
Hora de salida	<input type="text" value="Formato: HH:MM"/>
Hora de llegada	<input type="text" value="Formato: HH:MM"/>
Hora de embarque	<input type="text" value="Formato: HH:MM"/>
Fecha	<input type="text" value="Formato: AAAA-MM-DD"/>

[Atrás](#) [Añadir](#)

Continuem amb nous increments del producte en quant al Back-End, si ens movem a l'apartat gestió de cotxes, exactament el mateix. Podem veure la taula dels cotxes que hi han a la base de dades, degut a una petició AJAX a la API i una consulta de la API a la base de dades, completament transparent per a l'administrador:

Gestión de Coches

Coche	Estado	Localización
CH0000	Disponible	264,33
CH0001	Disponible	123,45
CH1234	Disponible	256,12

Modificar

Añadir

Com a l'apartat gestió de cotxes, podem afegir des del botó afegir, un nou cotxe i aquest s'afegirà a la base de dades i apareixerà a la pàgina:

Gestión de Coches

Coche	Estado	Localización
CH0000	Disponible	264,33
CH0001	Disponible	123,45
CH1234	Disponible	256,12

Nuevo coche

p.ej CH545, CH2581...

Nuevo estado

p.ej Averiado, Cargando...

Nueva localización

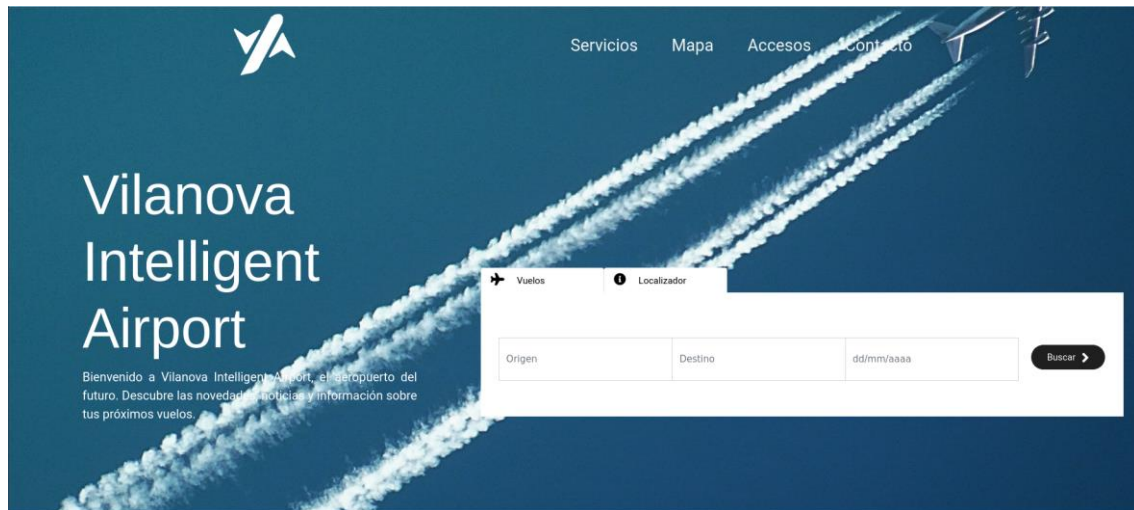
Formato: 111,11

Atrás

Añadir

Podem observar com les dades que ens demanen, tornen a ser les mateixes que les de la col·lecció *Nodes*, la col·lecció indicada per als cotxes a la base de dades amb els seus atributs *Id*, *State* i *Location*.

Passem ara a comentar els increments de la pàgina destinada a usuaris de l'aeroport:



De la pàgina principal s'han millorats alguns detalls del disseny que a l'anterior Sprint estaven sense terminar, però el gran increment ha sigut la creació de la pestanya informació de vols i localitzador. Clicant sobre el botó Buscar > podem accedir a la pestanya d'informació de vols:



Llegadas de

Vilanova i la Geltrú - Aeropuerto De Vilanova
Terminal 2

Horario: De 06:00 a 07:00

Filtros:

Aerolíneas: ☐ IBERIA ☐ Vueling ☐ AirEuropa ☐ Ryanair (disabled)

Origen: ☐ BCN ☐ MAD ☐ VLC ☐ MAH

Actualizar Actualizado hace 2 seg

Miércoles 13/03/2020


Hora	Vuelo	Destino	Aerolínea
9:00	IB6535	Valencia(VLC)	IBERIA
10:00	IB6534	Barcelona-El Prat (BCN)	IBERIA


En aquesta pàgina, la que encara es merament disseny, podem veure la informació dels vols d'arribades a l'aeroport, properament es connectarà amb la API per mostrar les arribades que estan presents a la base de dades.

També podem veure la pestanya de localització:

Vuelo: IBE8335

Vilanova i la Geltrú- Aeropuerto de Vilanova (VIA) → Badajoz(BJZ)

 Salida: Vilanova International Airport (VIA)

 Llegada: Badajoz(BJZ)

Aerolínea	Fecha	Hora	Terminal	Puerta	Estado
Vueling	27/04/2020	15:10	2	54	En hora

De moment, també es merament disseny i no es connecta amb dades reals, però properament ho farà i es podrà buscar per un número de vol concret i es rebran les dades del mateix.

Un altre gran canvi ha sigut prescindir de la API de GMaps.

Si l'usuari es mou a la pestanya Accessos, ara ja no veu GMaps, si no altres indicacions per arribar a l'aeroport:

Medios de transporte

Viaja al aeropuerto de Vilanova de múltiples formas



Coche

Puedes llegar a la Terminal 2 del aeropuerto a través de la autopista AP-7 y la carretera C-32. El aeropuerto está suficientemente señalizado.



Tren

Si quieres usar el transporte público puedes llegar a la Terminal 2 mediante la línea R2 o Regional Express de la compañía de trenes Renfe.

[Ver parada de tren](#)



Coche autónomos

Disponemos de una gran flota de coches autónomos que puedes solicitar mediante nuestra App gratuita. Ves y vuelve del aeropuerto sin tener que preocuparte de nada.

[Descarga App](#)



Taxi

El aeropuerto está en taxi a unos 40 minutos desde Barcelona.



Bus

Para ir en Bus, diversas compañías ofrecen servicio de transporte al aeropuerto como Autocars Plana i Monbus. La parada más cercana es Plaça d'Eduard Maristany.

[Ver parada de Bus](#)



A pie

Si por otra parte quieres ir a pie porque vives en las cercanías del aeropuerto, no hay ningún problema. La terminal 2 se encuentra en Av. Víctor Balaguer, 1. 08800 Vilanova i la Geltrú.

[Ver aeropuerto](#)

Aquest canvi s'ha realitzat per a evitar que el client pagui GMaps, i la versió gratuïta era massa reduïda en quant a nombre de peticions per abastar les necessitats de l'aeroport, a més, així ens estalviem problemes d'usuaris amb intencions dolentes. En cada tipus de mitjà de transport (excepte taxis i cotxes) tenim un botó que ens manarà a GMaps per tenir una ubicació exacta de l'aeroport i la seva parada de tren o autobús més propera.

Passem ara a parlar de la API.

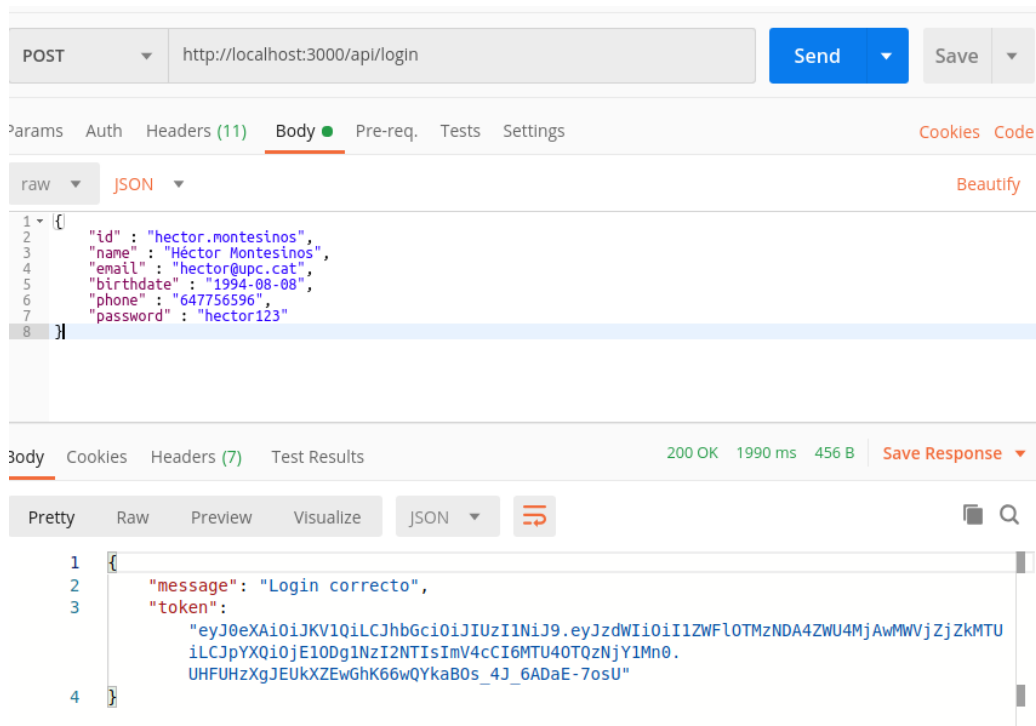
S'han adaptat peticions per a totes les col·leccions de la base de dades real de la Terminal, i s'han fet proves amb una base de dades MongoDB que la simulava.

```
> show dbs;
admin          0.000GB
config         0.000GB
local          0.000GB
restapi-local  0.001GB
> use restapi-local;
switched to db restapi-local
> show collections;
administrators
flights
nodes
operators
passengers
shops
>
```

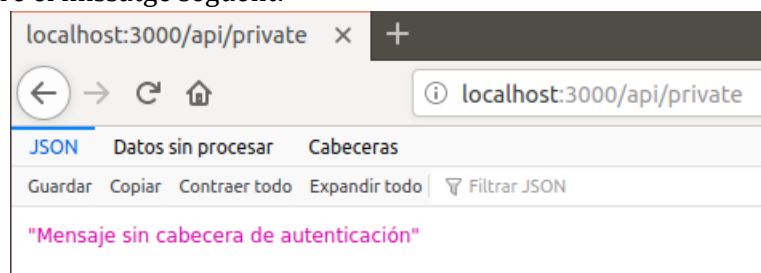
S'ha implementat un middleware d'autenticació i un sistema de Tokens mitjançant JsonWebToken (JWT), pel moment aquesta implementació es fa des de el client POSTMAN, ja que la web d'administradors encara no té una pestanya dissenyada per aquesta funcionalitat, però pel proper Sprint dos dels increments seran, poder entregar el Token al fer el Login a la web d'administradors i poder afegir/esborrar administradors des de la pròpia pàgina, cosa que ja es pot fer des de POSTMAN ja que n'hi ha una petició destinada a aquesta funcionalitat.

La funcionalitat de JWT per fer entrega del Token només estarà disponible a l'hora de fer el Login. Un administrador amb les seves credencials s'autenticarà a la web i se li entregarà el Token, el qual serà necessari per a poder veure qualsevol pàgina de la web, ja que sense ell, aquest informació serà totalment oculta.

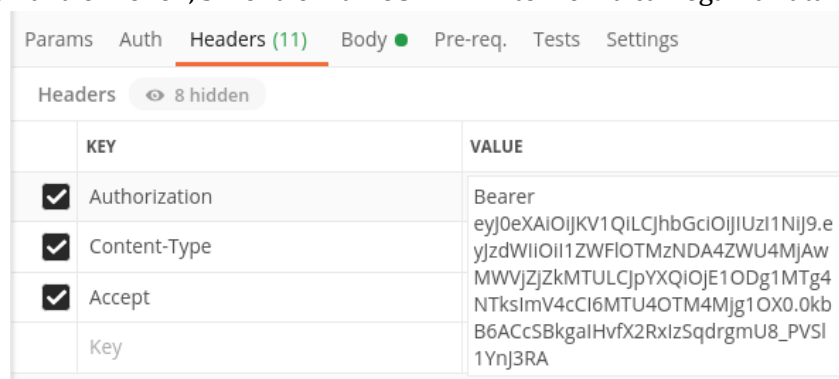
Ara mateix, només s'ha adaptat el middleware d'autenticació per a una ruta específica per a fer proves i mostrar-les. La ruta creada serà la `/private`. Des de POSTMAN podem provar a fer un Login amb credencials correctes:

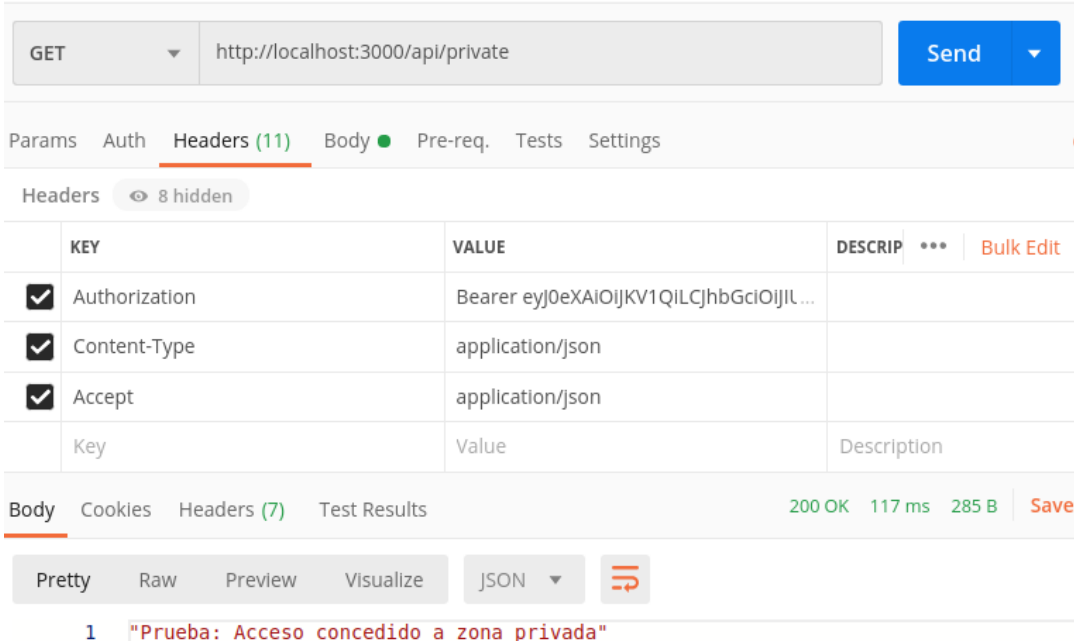


Comprovem que es rep el Token, un cop entrem a la ruta privada (des de Postman o des de la web) podem veure el missatge següent:



La ruta ens demana el Token, si l'entrem a POSTMAN i tornem a carregar la ruta:





GET http://localhost:3000/api/private Send

Params Auth **Headers (11)** Body Pre-req. Tests Settings

Headers 8 hidden

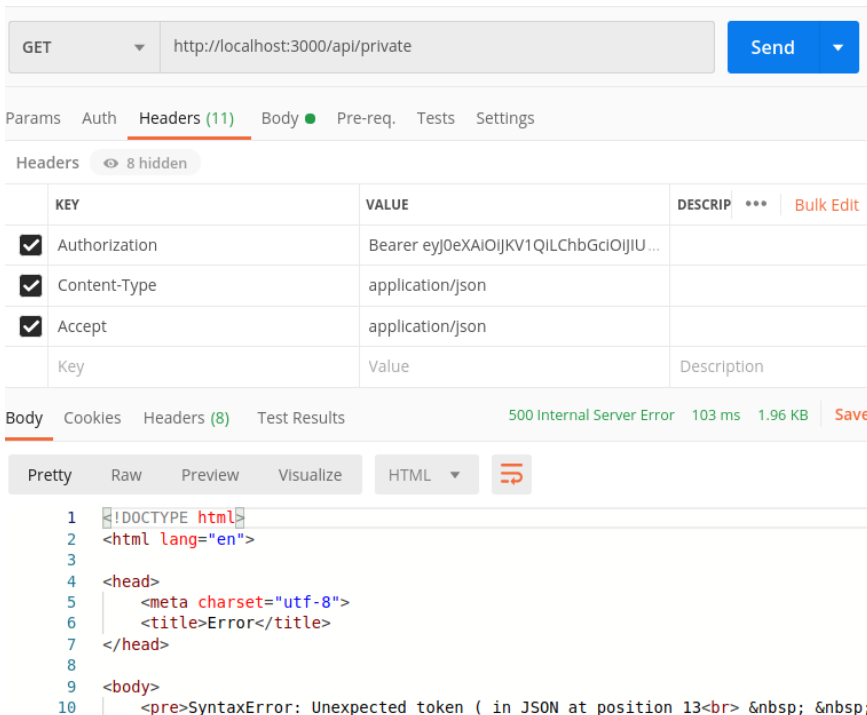
	KEY	VALUE	DESCRIP	...	Bulk Edit
<input checked="" type="checkbox"/>	Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJI...			
<input checked="" type="checkbox"/>	Content-Type	application/json			
<input checked="" type="checkbox"/>	Accept	application/json			
	Key	Value	Description		

Body Cookies Headers (7) Test Results 200 OK 117 ms 285 B Save

Pretty Raw Preview Visualize JSON

```
1 "Prueba: Acceso concedido a zona privada"
```

Veiem que ara rebem un missatge de 200 OK amb autorització d'entrada a la ruta /private. Això properament es el que passarà a totes del rutes del Back-End quan tinguem autorització, però en comptes de mostrar aquest missatge, mostrarà el contingut de les pàgines amb seguretat. Si introduïm un Token que no existeix, apareix un error.



GET http://localhost:3000/api/private Send

Params Auth **Headers (11)** Body Pre-req. Tests Settings

Headers 8 hidden

	KEY	VALUE	DESCRIP	...	Bulk Edit
<input checked="" type="checkbox"/>	Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJI...			
<input checked="" type="checkbox"/>	Content-Type	application/json			
<input checked="" type="checkbox"/>	Accept	application/json			
	Key	Value	Description		

Body Cookies Headers (8) Test Results 500 Internal Server Error 103 ms 1.96 KB Save

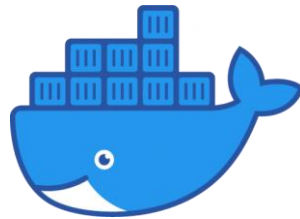
Pretty Raw Preview Visualize HTML

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <title>Error</title>
7 </head>
8
9 <body>
10  <pre>SyntaxError: Unexpected token ( in JSON at position 13<br> &nbsp; &nbsp; &nbsp;
```

La gestió d'errors quan no el sistema no reconeix el Token o quan està caducat, la millorarem també properament per a que mostri un missatge concret i no l'error, encara que no ens sembla massa urgent.

Parlem per últim de l'increment de connexió de serveis.

A diferència de l'anterior Sprint, on tots els nostres serveis es van mostrar per separat, ara tot està connectat mitjançant contenidors Docker.



Les dues webs estan cadascuna en un contenidor Docker on corre un servidor web Apache al seu interior i tant la API com la base de dades estan en un contenidor Docker independent.

Tots els contenidors estan connectats entre si mitjançant Docker-Compose. El seu fitxer de configuració es docker-compose.yml i el contingut es el següent:

```
GNU nano 2.9.3 docker-compose.yml
version: "2"
services:
  frontend:
    image: httpd:2.4
    ports:
      - "8080:80"
    volumes:
      - ./VIA:/usr/local/apache2/htdocs
  backend:
    image: httpd:2.4
    ports:
      - "8081:80"
    volumes:
      - ./Administrador_definitivo:/usr/local/apache2/htdocs
  api:
    build: .
    ports:
      - "3000:3000"
    depends_on:
      - db
  db:
    image: mongo
    ports:
      - "27017:27017"
```

Aquesta connexió anirà dins del servidor Debian que properament es migrarà al CRAAX, amb una configuració més avançada en quant a paràmetres de xarxa.

Desplegament del servei web i API

Descripció general

//Descripció més detallada i concreta d'aquesta part del treball.

Web Administrador (Back-End):

Pàgina d'inici: Modificada amb un fons comú amb el de la pàgina d'usuaris.

Login: Funcional amb dades reals, detecció de majúscules, contrasenya oculta i opció de visualitzar-la amb un botó que fa d'ull.

Gestió de cotxes: Veiem les dades dels cotxes presents a la base de dades i podem afegir-ne nous, properament, també modificar-los.

Mapa: Continua en el mateix estat que l'SPRINT2, properament mostrarà dades en temps real de passatgers.

Gestió de vols: Veiem les dades dels vols presents a la base de dades i podem afegir-ne nous, properament, també modificar-los.

Web Usuaris (Front-End):

Pàgina principal: S'han fet millores al codi CSS per a una interacció més àgil.

Informació de vols: Nova pestanya que mostra els vols, properament, mostrarà dades reals de la base de dades, ara mateix, merament disseny.

Localitzador: Nova pestanya que permet cercar un vol concret, properament, farà una cerca a la base de dades, ara mateix, merament disseny.

Serveis: Continua en el mateix estat que l'anterior Sprint, on estava molt avançada, apareixen els comerços de l'aeroport.

Mapa: Apareix el mapa de l'aeroport de les dues plantes, continua en el mateix estat que l'anterior Sprint. Properament es situaran els comerços.

Accessos: S'ha prescindit de GMaps, ara mostra diverses opcions d'arribada i localització de l'aeroport. Cada localització està mapejada a la web de GMaps per a que l'usuari la pugui ubicar i arribar a ella.

Contacte: Pendent d'implementar per a que mostri un formulari de contacte.

API Rest:

Funcionalitats: S'han adaptat peticions GET, POST, PUT i DELETE per a les col·leccions reals de la base de dades *terminal2*.

Seguretat: S'ha implementat la gestió de Tokens mitjançant JWT, creant un middleware d'autenticació i està funcionant per a fer un login.

Connexió dels serveis:

S'han interconnectat tots els nostres serveis mitjançant Docker-Compose.

Cada servei es troba a un contenidor Docker mapejat a un port concret.

Les dues webs estan als ports 8080 i 8081, la API al port 3000 i la base de dades al port 27017.

Explicació software/hardware utilitzat

Node.js: Entorn que treballa en temps real d'execució, de codi obert, multi plataforma que permet al programador crear tota classe d'eines del costat servidor (BackEnd) i aplicacions en Javascript.

Docker: Projecte de codi obert que automatitza el desplegament d'aplicacions dins de contenidors software.

POSTMAN: Software que simula peticions a la API de rutes, des d'una aplicació web/mòbil.

MongoDB: Sistema de base de dades NoSQL, orientat a documents i de codi obert.

Mòduls necessaris de Node.js:

Express: Framework de Node, crea l'estructura del servidor i ens permet escriure codi de manera senzilla.

Body-Parser: Mòdul que ens permet convertir les dades que ens arriben en les peticions al servidor, en objectes amb format JSON.

Mongoose: Mòdul que ens proveeix mètodes i funcionalitats per a treballar millor amb MongoDB.

Nodemon: Mòdul que farà que amb qualsevol canvi en el servidor, aquest es reiniciï (fa com una compilació), a més de que mostra per consola totes les peticions HTTP (GET, POST, PUT, DELETE...) que rep el servidor (200, 404, 500, etc.)

Cors: Mòdul que permet que es puguin sol·licitar recursos exclusius.

JsonWebToken: Mòdul que permet el sistema d'autenticació mitjançant Tokens.

JavaScript: Llenguatge de programació.

HTML5: Conjunt de tecnologies Web

CSS3: Llenguatge de disseny gràfic enfocat en Web.

Bootstrap: Conjunt d'eines per a disseny Web.

AJAX (JavaScript And XML): Tècnica de desenvolupament web per a crear aplicacions interactives.

W3.CSS Icons: Llibreria d'icones gratuïts de W3Schools.

Ionic icons: Pàgina web d'icones.

Flaticons: Pàgina que ofereix fotos en format png.

Pexels: Servei que ofereix fotos de stock.

Weatherwidget: Widget responsive sobre el clima en temps real de qualsevol ubicació.

Manual d'usuari

//Interaccions que pot realitzar l'usuari amb el projecte actual.

Amb el codi Web, codi de l'API i els fitxers Dockerfile per muntar la API i docker-compose.yml per interconnectar tots els serveis en un mateix directori, qualsevol podria entrar la comanda *docker-compose up* i poder gaudir dels quatre serveis interconnectats funcionant i poder veure totes les funcionalitats que hem indicat a l'apartat anterior:

```
Héctor (B3) > pwd
/home/hector/Escritorio/APIRest
Héctor (B3) > ls -l
total 232
drwxr-xr-x  5 hector hector  4096 may  3 12:08 Administrador_definitivo
-rw-r--r--  1 hector hector   470 may  2 20:24 docker-compose.yml
-rwxr-xr-x  1 hector hector   238 may  1 19:04 dockerfile
-rwxr-xr-x  1 hector hector   267 may  1 20:02 dockerfile.save
drwxr-xr-x 271 hector hector 12288 may  1 11:20 node_modules
-rw-r--r--  1 hector hector   749 may  1 11:20 package.json
-rw-r--r--  1 hector hector 190541 abr 30 19:36 package-lock.json
-rw-r--r--  1 hector hector   134 may  2 09:13 Readme.md
drwxr-xr-x  8 hector hector  4096 may  2 16:28 src
drwxr-xr-x  6 hector hector  4096 may  3 10:16 VIA
Héctor (B3) >
```

```
hector@hector:~/Escritorio/APIRest$ docker-compose up
Starting apirest_backend_1 ...
Starting apirest_db_1 ...
Starting apirest_backend_1
Starting apirest_frontend_1 ...
Starting apirest_db_1
Starting apirest_db_1 ... done
Starting apirest_api_1 ...
Starting apirest_api_1 ... done
Attaching to apirest_db_1, apirest_frontend_1, apirest_backend_1, apirest_api_1
db_1      | 2020-05-04T05:39:22.672+0000 I CONTROL [main] Automatically disabled
db_1      | 2020-05-04T05:39:23.444+0000 W ASIO [main] No TransportLayer con
db_1      | 2020-05-04T05:39:23.445+0000 I CONTROL [initandlisten] MongoDB start
```

Conclusions

Estem aprenent molt en quant a programació Web i també en quant a infraestructura degut a l'ús de Docker i els moviments de dades que fa la API entre els diferents serveis.

Definició de protocols

Descripció general

//Descripció més detallada i concreta d'aquesta part del treball.

Els clients Web es connecten a la API mitjançant peticions AJAX (JavaScript And XML).

La API fa peticions HTTP (Protocol de transport TCP) a la base de dades MongoDB. La base de dades MongoDB retorna dades en format JSON (format de text orientat al intercanvi de dades) i aquest JSON es retorna als clients Web.

En quant a l'aplicació mòbil es quasi bé el mateix, la única diferencia es que les peticions son FETCH.

En quant als cotxes, s'obrirà un WebSocket a la API i es connectaran.

Explicació/Justificació

//Explicació de la raó per la qual s'utilitzaran els protocols definits

Trobem que son les connexions mes utilitzades per aquests tipus de serveis, i en quant a la connexió mitjançant socket amb els cotxes, creiem que es la millor opció per a fer una connexió entre Node.js (API) i Python (cotxes).

Conclusions

/**Cal afegir un manual o be un link amb documentació dels protocols utilitzats.

Els protocols utilitzats es poden veure a l'esquema general inclòs al primer apartat.