ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΜΑΤΙΚΗΣ
ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΑΣ

Αναστασίου Θεοδώρα itp19103
Λίχας Άγγελος itp19123
Μητσοκάπας Κωνσταντίνος itp19124

Πληροφοριακά Συστήματα Ιστού
https://github.com/kostas00/docker-app

Σεπτέμβριος 2020

Βάση Δεδομένων :

Η βάση δεδομένων μας φτιάχτηκε σε PostgreSQL σύστημα, αφού έγιναν οι απαραίτητες ρυθμίσεις του postgresql server.

Η βάση δεδομένων μας αποτελείται από τέσσερις πίνακες:

Users : Περιέχει τα δεδομένα που χρειαζόμαστε για τους χρήστες της εφαρμογής μας
userid serial PRIMARY KEY,

role smallint NOT NULL : Ουσιαστικά καθορίζει τι τύπος user είναι ο συγκεκριμένος user - για 0 θεωρούμε ότι είναι απλός user για 1 θεωρούμε ότι είναι χρήστης που διαχειρίζεται ραντεβού φορέων)

name varchar (25) NOT NULL,
surname varchar (25) NOT NULL,
username varchar (15) UNIQUE,
email varchar (25) NOT NULL,
dateofbirth timestamp NOT NULL,
password varchar (50) NOT NULL

Carriers : Τον χρησιμοποιούμε για αποθήκευση των απαραίτητων στοιχείων των φορέων ώστε να έχουμε άμεση πρόσβαση σε αυτά.

id serial PRIMARY KEY,
name varchar (25),
userid int, (το id του χρήστη που είναι διαχειριστής στον συγκεκριμένο φορέα)
CONSTRAINT fk_userid FOREIGN KEY (userid) REFERENCES myuser (userid)

Appointments : Ο πίνακας που περιέχει όλα τα απαραίτητα δεδομένα και σχέσεις για τα ραντεβού. Συνδέεται με τον πίνακα carries για να ξέρουμε σε ποιό φορέα αναφέρεται το ραντεβού αλλά και με τον πίνακα των users για να ξέρουμε ποιός το έκλεισε.

id serial PRIMARY KEY,
number smallint,
isSelected boolean,
isReserved boolean,

carrierid int NOT NULL,
dateofapp date,
userid int,
CONSTRAINT fk_carrierid FOREIGN KEY (carrierid) REFERENCES carriers (carrierid)
CONSTRAINT fk_userid FOREIGN KEY (userid) REFERENCES myuser (userid)

Notifications : Ο πίνακας αυτός κρατάει τις ειδοποιήσεις για τα ραντεβού. Ενώ επίσης συνδέεται με τον πίνακα ραντεβού για να έχει πρόσβαση αργότερα στα απαραίτητα στοιχεία.

id serial PRIMARY KEY,
dateandtime timestamp,
appointmentid int,
CONSTRAINT fk_appointmentid FOREIGN KEY (appointmentid) REFERENCES appointments (id)

Βάση Δεδομένων Cache (Redis):
Στην cache βάση δεδομένων αποθηκεύουμε τα ραντεβού τα οποία είναι προς επιβεβαίωση, με key το id του ραντεβού ώστε αργότερα όταν αυτό επιβεβαιωθεί να σβηστεί.

Backend και FastAPI της εφαρμογής ανα microservice:

Γενικά : Το κάθε microservice τρέχει ξεχωριστά πάνω σε διαφορετικό port, παρόλα αυτά χρειάζονται να τρέχουν όλα την ίδια ώρα, αφού το ένα καλεί το άλλο ώστε να γίνονται οι λειτουργίες απο το microservice που πρέπει. Τα microservices γραφτηκαν σε python 3.6.9 και χρησιμοποιούν sqlAlchemy queries.

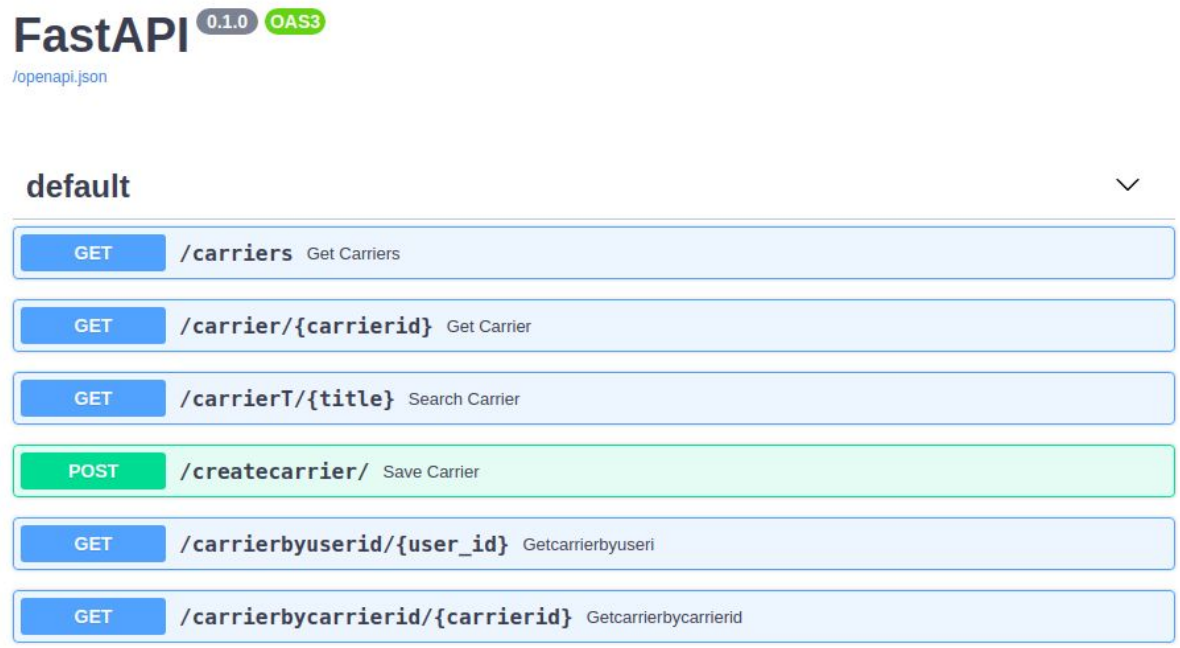
Κάθε microservice είναι σε δικό του φάκελο, και αποτελείται από ένα φάκελο app που περιέχει μέσα το αρχείο db.py(στο οποίο υπάρχουν τα απαραίτητα δεδομένα για την σύνδεση με την βάση μας) , db_manager.py (υπάρχουν οι συναρτήσεις που χρησιμοποιούν αρχότερα οι μεθόδοι της main.py) , main.py/app.py (περιέχει τις μεθόδους που καλούμε από το fastAPI και το front-end), models.py(στο αρχείο αυτό ορίζονται οι πίνακες και οι σχέσεις που χρησιμοποιεί το συγκεκριμένο service), schemas.py (ορίζονται τα σχήματα και οι κλάσεις που χρησιμοποιεί το συγκεκριμένο service)

Microservice Login/SignUp

Το microservice αυτό αποτελείται κυρίως από τις τέσσερις παραπάνω λειτουργίες,

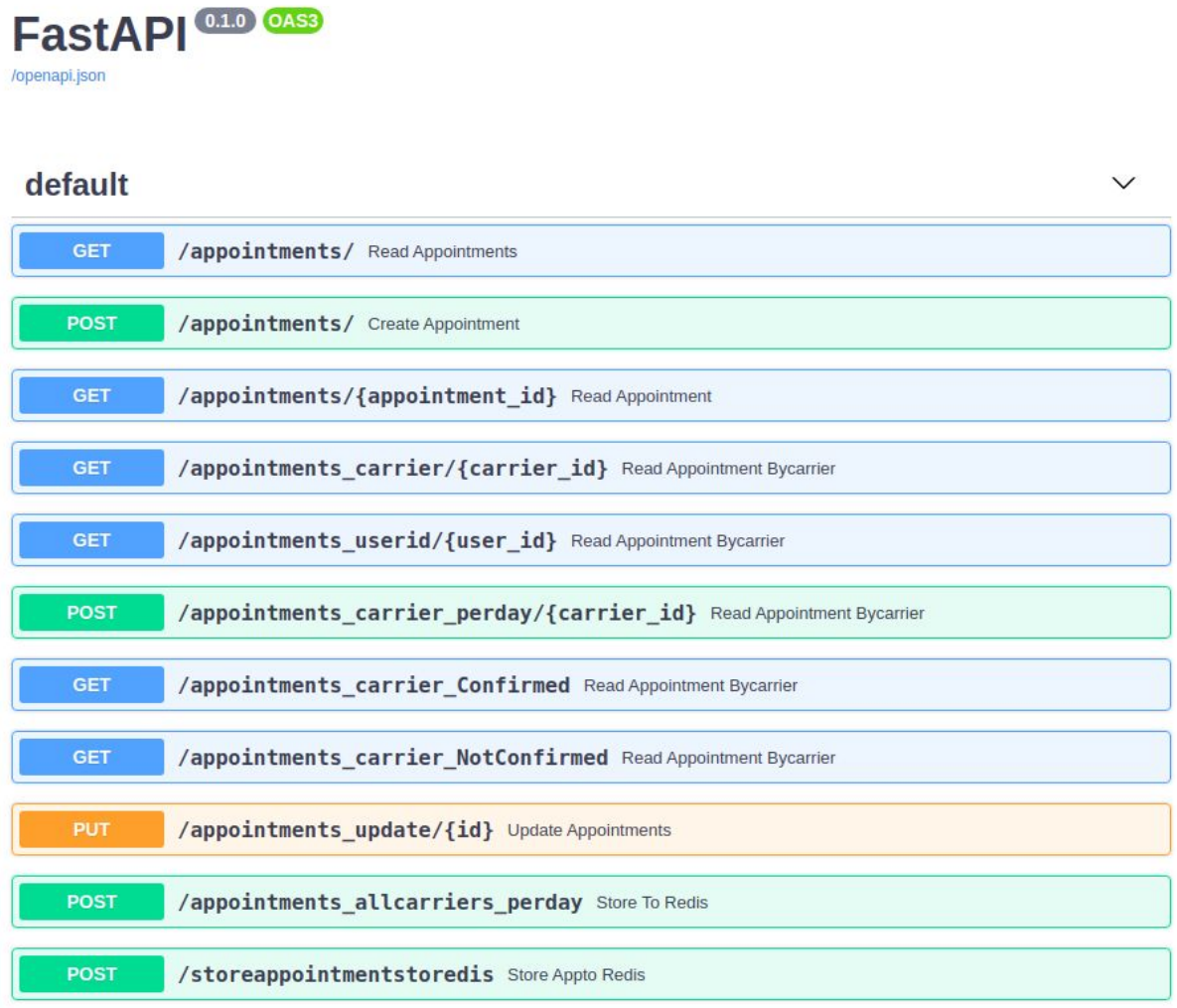
1) Βασική λειτουργία login, ελέγχει εάν ο χρήστης έχει δώσει σωστό email ή username αλλά και σωστό password (ο οποίος χρειάζεται αποκρυπτογράφηση ώστε να αυθεντικοποιηθεί) . Παράλληλα αφού ο χρήστης δώσει τα σωστά στοιχεία το login επιστρέφει ένα token το οποίο χρησιμοποιούμε αργότερα στην εφαρμογή ώστε να παραμένει συνδεδεμένος ή και όχι ο συγκεκριμένος χρήστης.
2) Διαβάζει όλους του χρήστες της βάσης
3) Βασική λειτουργία Register , ο χρήστης συμπληρώνει τα απαραίτητα στοιχεία, γίνονται οι απαραίτητοι έλεγχοι για το αν υπάρχει άλλος χρήστης με το ιδιο username/email, ενώ παράλληλα ο κωδικός που δίνει ο χρήστης κρυπτογραφείται και γράφεται στην βάση.
4) Διαβάζει τον χρήστη με το αντίστοιχο user_id

Microservice Carriers:



1) Εμφανίζει όλους τους φορείς απο το https://hr.apografi.gov.gr/api.html χρησιμοποιώντας requests
2) Εμφανίζει τους φορείς με id απο το https://hr.apografi.gov.gr/api.html χρησιμοποιώντας requests
3) Εμφανίζει τους φορείς με τίτλου απο το https://hr.apografi.gov.gr/api.html χρησιμοποιώντας requests
4) Δημιουργεί νέο φορέα, ουσιαστικά το χρησιμοποιούμε για να περνάμε τους φορείς που επιστρέφει τα πάνω requests στη βάση μας, για να έχουμε άμεση πρόσβαση.
5) Παίρνουμε από την βάση τον φορέα που έχει σαν χρήστη τον χρήστη με user_id
6) Παίρνουμε από την βάση τον φορέα που έχει σαν id το carrierid

Microservice Appointments:



1) Εμφανίζει όλα τα ραντεβού από τη βάση μας
2) Ο χρήστης συμπληρώνει τα απαραίτητα στοιχεία για την εισαγωγή νέου ραντεβού θεωρούμε ότι στην αρχή το ραντεβού καταχωρείται πάντα με isselected = false, isreserved=true
   (Θεωρούμε ότι το isselected μας δείχνει αν το ραντεβού έχει κλείσει, ενώ το isreserved ότι το ραντεβού περιμένει να γίνει confirm)
   Το ραντεβού εδώ απο

3) Εμφανίζει το ραντεβού με appointment_id
4) Εμφανίζει τα ραντεβού του φορέα με id carrier_id
5) Εμφανίζει τα ραντεβού του συγκεκριμένου χρήστη
6) Παίρνει τα ραντεβού του συγκεκριμένου φορέα την συγκεκριμένη μέρα που παίρνει απο την φόρμα του frontend
7) Εμφανίζει όλα τα επιβεβαιωμένα ραντεβού για τον συγκεκριμένο φορέα
8) Εμφανίζει όλα τα μη επιβεβαιωμένα ραντεβού για τον συγκεκριμένο φορέα
9) Τροποποιεί το συγκεκριμένο ραντεβού ώστε στην περίπτωση που το εγκρίνει ο χρήστης του φορέα να είναι πλέον το isselected=true και το isreserved = false, ενώ ταυτόχρονα το διαγράφει από την μνήμη cache.
10) Εμφανίζει όλα τα ραντεβού ανα ημέρα για όλους τους φορείς
11) Ο χρήστης συμπληρώνει τα απαραίτητα στοιχεία για την εισαγωγή νέου ραντεβού θεωρούμε ότι στην αρχή το ραντεβού καταχωρείται πάντα με isselected = false, isreserved=true
(Θεωρούμε ότι το isselected μας δείχνει αν το ραντεβού έχει κλείσει, ενώ το isreserved ότι το ραντεβού περιμένει να γίνει confirm)
Το ραντεβού εδώ αποθηκεύεται στην cache-redis μέχρι να επιβεβαιωθεί και να διαγραφεί από την cache.

Microservice Notifications:

Χρησιμοποιούμε την συνάρτηση **send_confirm_email(email_1,email_2,password,number)** ώστε να στέλνουμε emails σαν ειδοποιήσεις προς του χρήστες μας. Φτιάξαμε ένα gmail ωστε να στέλνονται τα mails της εφαρμογής μας από αυτόν τον λογαριασμό. Έτσι αφού δώσουμε το email της εφαρμογής μας, το email του παραλήπτη , τον κωδικό του email της εφαρμογής μας και έναν αριθμό, ώστε αντίστοιχα να στέλνει τα πάρακάτω.

```
 if number == 1:
        message = """\
        Subject: Registration Confirmed!

        Your registration has been confirmed! """

    if number == 2:
        message = """\
        Subject: New Appointment Notify

        New appointment from user needs confirmation! Go check it out! """

    if number == 3:
        message = """\
        Subject: Appointment Confirmation Notify

        Your appoingment has been confirmed. Thanks for using our platform! """

    if number == 4:
        message = """\
        Subject: Appointment Submitted

        Your appointment has been submited,it may take some time to be confirmed
by host user. Thanks for using our platform! """
```

1) Προσθέτει νέα ειδοποίηση - μέσω email - αφού πάρει τα απαραίτητα στοιχεία.
2) Στέλνει email - ειδοποίηση στον χρήστη όταν κάνει εγγραφή στην εφαρμογή
3) Επιστρέφει όλες τις ειδοποιήσεις από την βάση
4) Επιστρέφει συγκεκριμένη ειδοποίηση από την βάση
5) Επιστρέφει τις ειδοποιήσεις για το συγκεκριμένο ραντεβού

Docker

Για τα παραπάνω api δημιουργήσαμε docker images και τα σηκώσαμε σε docker containers. Στήσαμε 6 συνολικά containers ένα για κάθε api μαζί με ένα για την βάση postgres και ένα για τον server του redis.
Τα containers επικοινωνούν μεταξύ τους και συνδέουν το ένα api με το άλλο. Στο front με την ίδια λογική στέλνουμε requests στα api που τρέχουν στα containers

Front-End React

Για την υλοποίηση του front-end συστήματος μας ώστε να έχει πρόσβαση ο χρήστης και ο υπάλληλος και να κλείνουν ραντεβού χρησιμοποιήσαμε React.

Ο χρήστης μπορεί να εισέλθει στο σύστημα αν έχει ήδη λογαριασμό ή να δημιουργήσει έναν κάνοντας register. Το login είναι ίδιο και για τους χρήστες και για τους υπαλλήλους. Αυτό το σύστημα ενώνεται με το microservice του Login/SignUp και μέσω requests επιτυγχάνεται ο έλεγχος και η πρόσβαση στο σύστημα.

Ανάλογα τον ρόλο του χρήστη εμφανίζεται διαφορετική σελίδα με τις επιλογές που μπορεί να πραγματοποιήσει.

Ο χρήστης μπορεί να δει τα ραντεβού που έχει κλείσει και να κλείσει εκ νέου καινούργιο ραντεβού.

Ο υπάλληλος βλέπει τα ραντεβού τα οποία χρειάζεται να επιβεβαιώσει και όλα τα ραντεβού για τον φορέα στον οποίο εργάζεται.