



ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ ΠΡΟΟΔΟΥ ΠΡΑΚΤΙΚΗΣ ΑΣΚΗΣΗΣ

Όνομα ασκούμενου: Κων/νος Νικολούτσος

Αρ. Μητρώου: p3170122

Φορέας Υλοποίησης Π.Α.: Myconstructor IKE

Επιβλέπων Καθηγητής: Κ.Ξυλωμένος

Εργασιακός Επιβλέποντας: Χρήστος

Φραγκουλάκης

Ακαδημαϊκό Εξάμηνο: Εαρινό 2020 - 2021

Λήξη πρακτικής 28 φεβρουαρίου



Επιχειρησιακό Πρόγραμμα
Ανάπτυξη Ανθρώπινου Δυναμικού,
Εκπαίδευση και Διά Βίου Μάθηση
Ειδική Υπηρεσία Διαχείρισης
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης



ΕΠΑνΕΚ 2014-2020
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΑΝΤΑΓΩΝΙΣΤΙΚΟΤΗΤΑ • ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΤΗΤΑ • ΚΑΙΝΟΤΟΜΙΑ

Περιεχόμενα

- Περιγραφή εταιρίας.....σελ. 3
- Υπάρχων εμπειρία για συγκεκριμένη θέση.....σελ. 4
- Οργάνωση χρόνου.....σελ. 4
- Στόχοισελ. 5
- Ρόλοι και καθήκοντα.....σελ. 5
- Αναλυτικά καθήκοντα.....σελ. 7
- Εργασία από το σπίτι λόγω covid.....σελ. 16
- Αποτελέσματα στόχων.....σελ 17
- Προβλήματα και αντιμετώπιση.....σελ. 18
- Επίλογος.....σελ. 19
- Χρήσιμοι σύνδεσμοι..... σελ 20

Η συνεργασία μου με την εταιρία Myconstructor αποδείχθηκε αρκετά ευχάριστη καθώς και οι δυο πλευρές μείναμε ικανοποιημένοι.
Ως αποτέλεσμα αυτού, προς το τέλος της πρακτικής μου έγινε κάποια προσφορά από την εταιρία καθώς επίσης ο CEO μου έκανε και review στην επαγγελματική πλατφόρμα LinkedIn



Fragkoulakis Chris

Ceo at MyConstructor.gr
January 4, 2021, Fragkoulakis managed Konstantinos directly

It's was a pleasure to work with Kosta! He is a tech enthusiast, passionate to learn new technologies and always perform his best to deliver a top-notch product! Kosta is a talented young professional and will be an asset for any tech team he'll work with.

Περιγραφή εταιρίας

Στο MyConstructor μπορείτε να βρείτε εξειδικευμένους και αξιόπιστους επαγγελματίες για περισσότερες από 50 τεχνικές εργασίες και υπηρεσίες, να συγκρίνετε τιμές και αξιολογήσεις και να διαλέξετε αυτόν που σας ταιριάζει.

Καθημερινά, βρίσκουμε και αξιολογούμε τους καλυτέρους συνεργάτες ώστε να είμαστε βέβαιοι από πρώτο χέρι ότι οι υπηρεσίες που παρέχουμε είναι εγγυημένα οι καλύτερες.

Σήμερα, δεκάδες χιλιάδες χρήστες από Ελλάδα και Αγγλία χρησιμοποιούν την πλατφόρμα μας με αποτέλεσμα να φέρνουμε κοντά τους καλυτέρους επαγγελματίες με ανθρώπους που έχουν ανάγκη τις υπηρεσίες τους.



Υπάρχων εμπειρία για συγκεκριμένη θέση

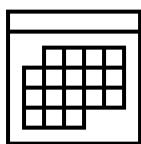
Κάτι το οποίο μπορώ να πω ότι με βοήθησε αρκετά ήταν ότι είχα ξαναδουλέψει στο παρελθόν σε παρόμοια θέση mobile developer με αποτέλεσμα πάνω-κάτω να γνωρίζω κάποια από τα τεχνολογικά εργαλεία/τεχνικές που χρησιμοποιούνταν στην συγκεκριμένη εταιρία.

Επίσης πολλά από αυτά τα έχουμε διδαχθεί και στην σχολή όπως τα παρακάτω:

- Polymorphism and OOP (Java 2)
- Sockets and deadlocks (λειτουργικά συστήματα)
- Websockets (κατανεμημένα)
- Git and design patterns (τεχνολογία λογισμικού)

Αυτό κέρδισε χρόνο με αποτέλεσμα να μπούμε κατευθείαν στο business logic με στόχο να ξεκινήσω να φτιάχνω και νούργια features μαζί με την ομάδα μου.

Οργάνωση χρόνου



1-3^η εβδομάδα : Onboarding – Εισαγωγή στο business logic

3-7^η εβδομάδα : android & iOS bug fixes

7-12^η εβδομάδα : android & iOS & backend implement new features

12=17^η εβδομάδα : backend δημιουργία τηλεφωνικού κέντρου

Στόχοι

Έπειτα από τις πρώτες 2-3 εβδομάδες, που αφιερώθηκαν στο onboarding δηλαδή στην ομαλή ένταξη μου στην ομάδα, θέσαμε τους παρακάτω στόχους που μπορεί να αποτελέσουν μετρικές αξιολόγησης:

- Αύξηση του user experience της mobile εφαρμογής (iOS, Android)
- Δημιουργία υποδομών για τηλεφωνικό κέντρο (Real time chat, Rest API)

Αξίζει να υπογραμμίσουμε ότι επιτεύχθηκαν όλοι οι στόχοι. Στα επόμενα sections μπορείτε να διαβάσετε περισσότερα για αυτούς.

Ρόλοι και καθήκοντα

Το target audience που επηρεάζονταν από τις αλλαγές και τις προσθήκες μου ήταν κυρίως οι επαγγελματίες που συνεργάζονταν με την εταιρία. Με λίγα λόγοι ασχολήθηκα κυρίως εφαρμογές B2B.

Ο ρόλοι που μου ανατέθηκαν κατά την διάρκεια της πρακτικής μου ήταν οι παρακάτω:

- Android developer
- iOS developer
- Backend developer

Οι τεχνολογίες που χρησιμοποιήθηκαν για τα παραπάνω ήταν αυτές που ήδη ήταν υλοποιημένος ο κώδικα της Myconstructor. Φυσικά αξίζει να σημειωθεί ότι εκτος από της ήδη υπάρχουσες τεχνολογιες(frameworks) που χρησιμοποιούσε η εταιρία προστέθηκαν και έξτρα τεχνολογίες που υπόσχονται καλύτερο performance.

Για το Android development έγινε χρήση και ασχολήθηκα με τα παρακάτω:

- Java
- Android SDK
- Firebase
- Jetpack architectural components
- 3rd party libraries
- MVVM architectural design pattern

Για το iOS development έγινε χρήση και ασχολήθηκα με τα παρακάτω:

- Swift
- Objective C
- iOS SDK
- Firebase
- APNS
- 3rd party libraries
- MVP architectural design pattern
- Crashlytics
- Fabric
- Pods

Για το backend development έγινε χρήση και ασχολήθηκα με τα παρακάτω:

- nodeJS
- Vanilla PHP
- MySQL
- Rest API
- Ratchet PHP
- Socket IO

Αναλυτικά καθήκοντα

Real time chat (Android, iOS, Backend):

Στην υπάρχουσα εφαρμογή έπειτα από συνεννόηση με τους stakeholders αποφασίστηκε να φτιαχτεί ένα real time chat από το οποίο θα μπορούν οι επαγγελματίες να στέλνουν μηνύματα τα οποία θα τα λαμβάνουν οι agents μέσω της ειδικής πλατφόρμας. Ο σκοπός του συγκεκριμένου feature ήταν να αυξηθεί το user experience των χρηστών καθώς θα υπάρχει ένα αμεσότερο κανάλι επικοινωνίας

Το συγκεκριμένο feature χτίστηκε με τεχνολογία WebSocket τα οποία είναι γνωστό ότι παρέχουν ένα bi-directional communication μεταξύ του client και του server. Διαφορετικές λύση (workarounds) ήταν το polling το οποίο συζητήθηκε κατά την διάρκεια των meetings με την ομάδα μου αλλά αποδειχθεί όχι τόσο αποδοτική λύση.

Για να φτιαχτεί λοιπόν ο WebSocket server έκανα τα παρακάτω βήματα:

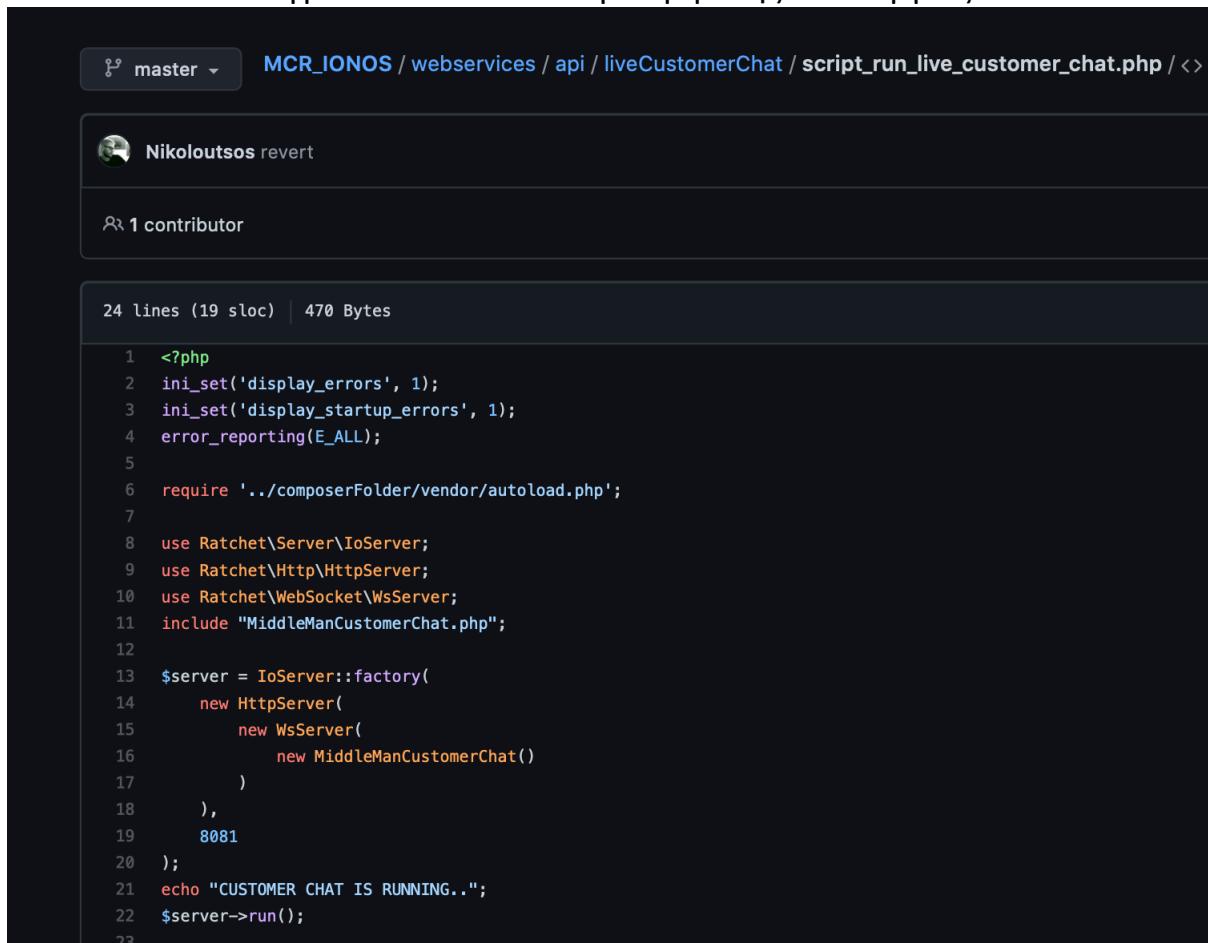
- NodeJS server με την χρήση της βιβλιοθήκης SocketIO
- Δημιουργία SSL certificate για να γίνεται η χρήση του https και να μην μπορεί κάποιος να κάνει spy. (security)
- Παραμετροποίηση κάποιον ρυθμίσεων του apache ώστε να επιτρέπονται περισσότεροι παράλληλοι χρήστες.
- Υλοποίηση των domain κλάσεων (message, user etc)
- Υλοποίηση της λογικής για την ροή εκτέλεσης (flow)
- Αρκετό manual και unit test για να επιβεβαιωθεί ότι η λειτουργία είναι αυτή που θέλουμε

Εφόσον λοιπόν φτιάχτηκε ο WebSocket server έπρεπε να γίνει consume από τους clients οπου στην συγκεκριμένη περίπτωση ήταν το iOS, android και web.

Εγώ ήμουν υπεύθυνος για το iOS και το Android και για την υλοποίηση τους δεν χρησιμοποιήθηκε κάποια έτοιμη βιβλιοθήκη αλλά έγινε χρήση των μεθόδων που έδινε το λειτουργικό.

Τέλος να υπογραμμίσουμε ότι οι συνολικές συνομιλίες που έχουν γίνει μεχρι τώρα **ξεπερνούν τις 3000** κάτι το οποίο δείχνει ότι η υλοποίηση είναι stable και με χαροποιεί ιδιαίτερα που τόσοι χρήστες βρήκαν χρήσιμο το live chat.

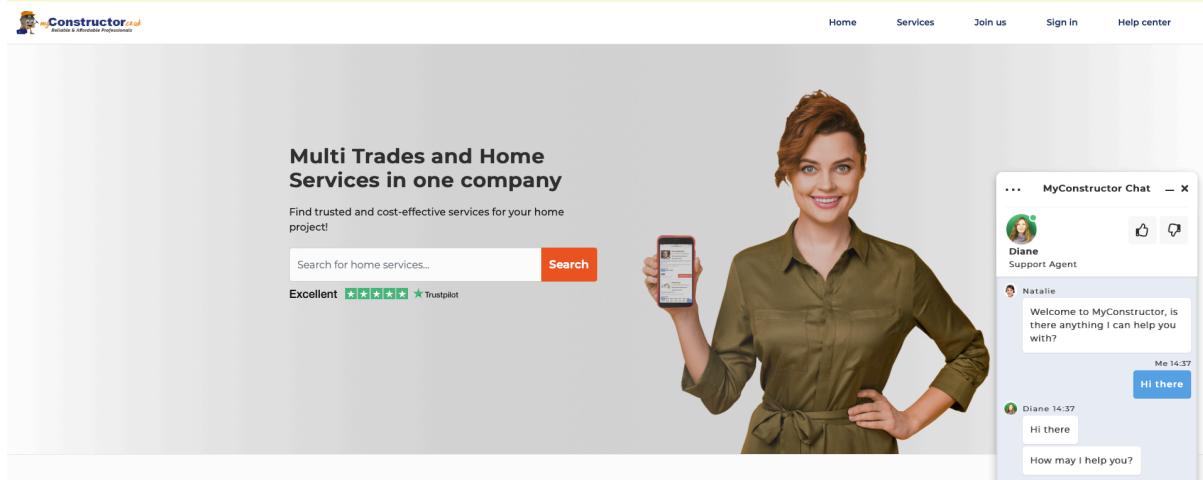
Στιγμιότυπο κώδικα συγκεκριμένης λειτουργίας



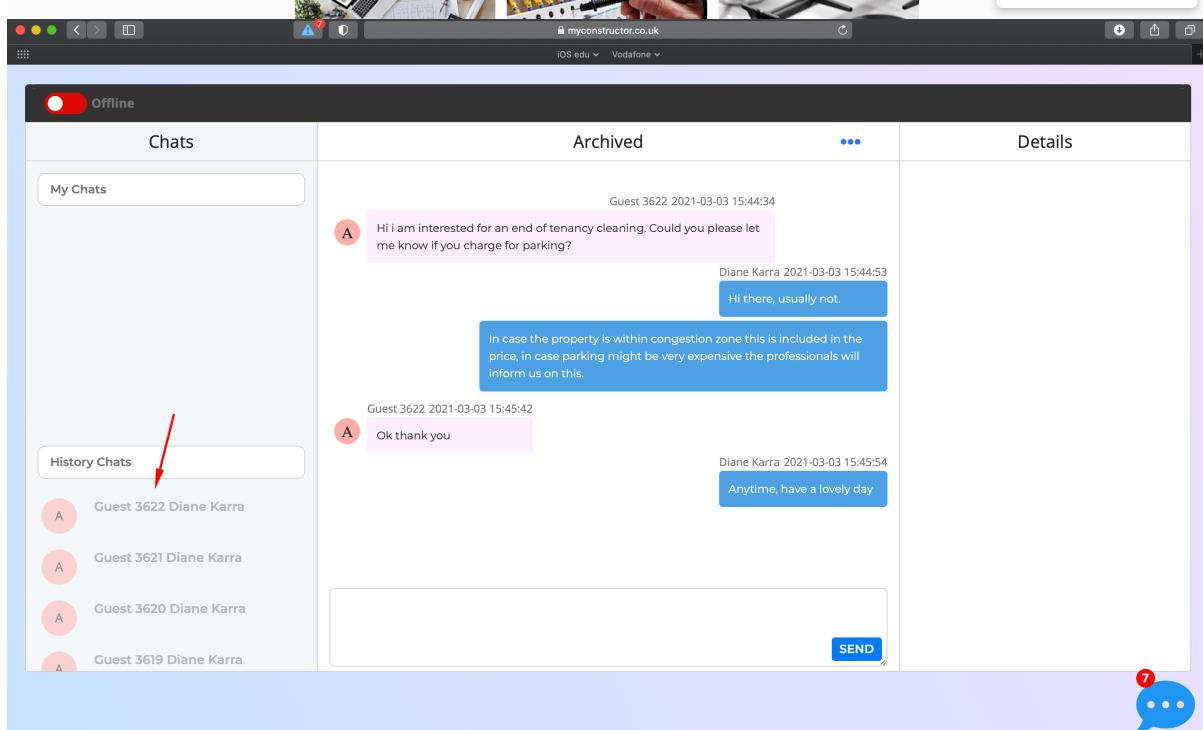
The screenshot shows a GitHub commit page for the file `script_run_live_customer_chat.php`. The commit was made by `Nikoloutsos revert` on the `master` branch. It has 1 contributor. The code is as follows:

```
1 <?php
2 ini_set('display_errors', 1);
3 ini_set('display_startup_errors', 1);
4 error_reporting(E_ALL);
5
6 require '../composerFolder/vendor/autoload.php';
7
8 use Ratchet\Server\IoServer;
9 use Ratchet\Http\HttpServer;
10 use Ratchet\WebSocket\WsServer;
11 include "MiddleManCustomerChat.php";
12
13 $server = IoServer::factory(
14     new HttpServer(
15         new WsServer(
16             new MiddleManCustomerChat()
17         )
18     ),
19     8081
20 );
21 echo "CUSTOMER CHAT IS RUNNING..";
22 $server->run();
23
```

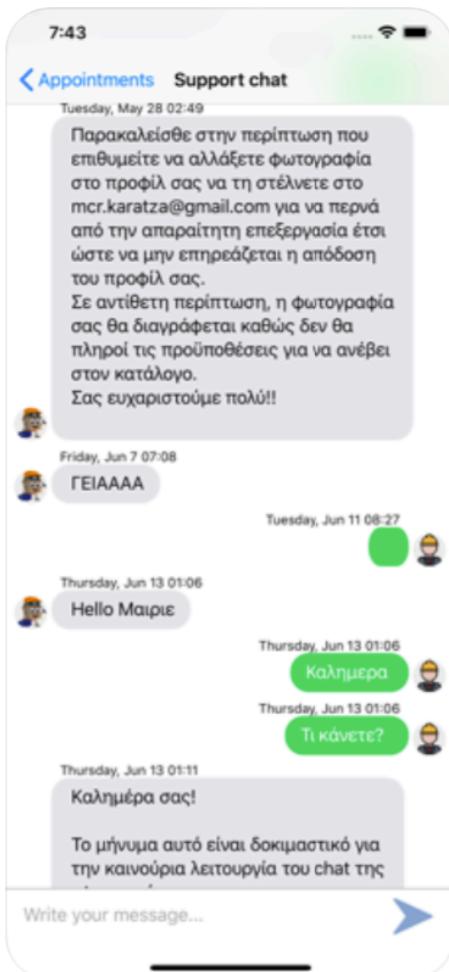
Παρακάτω μπορείτε να δείτε κάποια screenshot με την υλοποίηση στους clients:



The screenshot shows the homepage of the MyConstructor website. At the top, there's a navigation bar with links for Home, Services, Join us, Sign in, and Help center. Below the navigation is a large banner featuring a woman in a green jumpsuit holding a smartphone. The phone screen displays a live chat interface between a support agent named Diane and a user named Natalie. The banner also includes the text "Multi Trades and Home Services in one company" and a search bar. Below the banner are three small images of construction tools: a laptop, a hard hat, and a hand holding a tool.



The screenshot shows the MyConstructor chat history interface. It has a header with an offline status indicator and tabs for Chats, Archived, and Details. The Chats tab is active, showing a list of recent conversations. One conversation is highlighted with a red arrow pointing to it. The message history shows a guest asking about end-of-tenancy cleaning and parking charges, and a support agent responding that parking is usually not included unless the property is in a congestion zone. The guest then thanks the agent. A blue 'SEND' button is visible at the bottom right of the message input field. The interface is designed with a light purple background and white cards for each message.



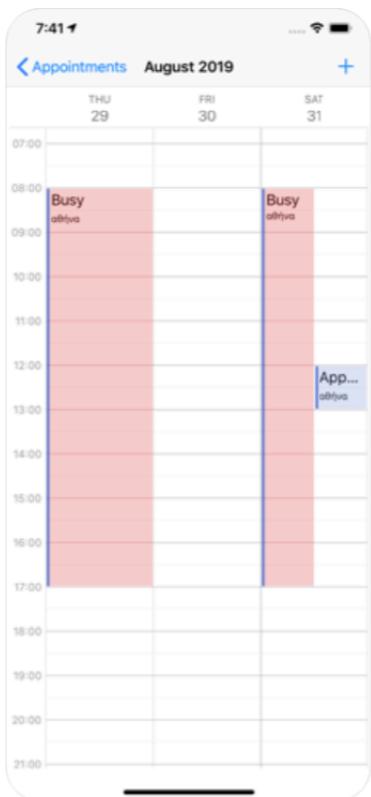
Google API calendar integration (Android, iOS):

Έπειτα από δίκη μου πρωτοβουλία κατά την διάρκεια του καθημερινού meeting που κάναμε πρότεινα να υλοποιηθεί μια καινούργια λειτουργία η οποία θα έδινε τεράστιο value στο business. Για να καταλάβουμε, η εταιρία για να παρέχει τις υπηρεσίες της πρέπει να γνωρίζει την διαθεσιμότητα των επαγγελματιών που συνεργάζονται μαζί της. Πριν υλοποιηθεί το feature αυτό ο μόνος τρόπος να γνωρίζουμε την διαθεσιμότητα ήταν ο επαγγελματίας έπρεπε να έβαζε manually μέσω της εφαρμογής ότι δεν θα μπορεί να δουλέψει σε ένα συγκεκριμένο timeslot.

Τώρα όμως με το **Google API calendar integration** αρκεί ο επαγγελματίας να συνδέσει στην εφαρμογή του το google account του και αυτόματα έπειτα από έγκριση του επαγγελματία η εταιρία θα είναι σε θέση να γνωρίζει την διαθεσιμότητα του. Για παράδειγμα αν ο Γιώργος είναι επαγγελματίας και προσθέσει στο google calendar ότι θα πάει στον οδοντίατρο στις 5-7 μ.μ. η βάση δεδομένων MySQL της εταιρίας θα ανανεωθεί αυτόματα γνωρίζοντας έτσι πλέον την διαθεσιμότητα του Γιώργου. Για να επιτευχθεί η αυτόματη ανανέωση χρησιμοποιήθηκαν web-hooks.

Για την υλοποίηση του παραπάνω στις εφαρμογές χρησιμοποιήθηκαν οι clients που παρέχει οι google και μπορούν να βρεθούν στο παρακάτω link:

<https://developers.google.com/calendar/overview>



Στιγμιότυπα κώδικα backend

The screenshot shows a GitHub repository page for the file `has_google_calendar.php`. The repository is `MCR_IONOS/webservices/api/app`. The file has 53 lines (39 sloc) and is 1.3 KB in size. The code is as follows:

```
1 <?php
2
3     include_once('../config/core.php');
4     header("Access-Control-Allow-Origin: \"$site_url\"");
5     header("Content-Type: application/json; charset=UTF-8");
6     header("Access-Control-Allow-Methods: GET");
7     header("Access-Control-Max-Age: 3600");
8     header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");
9
10    include_once('../config/utilities.php');
11    include_once('../config/validation.php');
12    include_once('../config/database.php');
13    include_once('GoogleApiClass.php');
14
15    ini_set('display_errors', 1);
16    ini_set('display_startup_errors', 1);
17    error_reporting(E_ALL);
18
19    // Get parameters
20    $prof_id = $professional->id;
21
22
23
24    switch ($http_method) {
25
26        case 'GET':
27            if ( $success ){
28                if ( $validated['data']['type'] == 'professional' ){
29
30                    $database = new Database();
31                    $db = $database->getConnection();
32                    $googleApi = new GoogleApiClass($db);
33
34                    $result = $googleApi->hasGoogleCalendarAccount($prof_id);
35
36                    if($result){
37                        echo json_encode(array("has_gc_account" => "1"));
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
```

Fix bugs and improve performance in mobile apps (android, iOS):

Φυσικά πέρα από την δημιουργία νέων λειτουργιών υπάρχει και η διόρθωση bugs γνωστό και ως maintenance.

Παρακάτω ακολουθούν κάποια από τα bugs που διόρθωσα:

- Η android εφαρμογή κράσαρε γιατί δεν γινόταν σωστό parsing JSON

- Αντικατάσταση linked-list με HashMap για να μειωθεί το big O complexity σε O(1) από O(n)
- Τα push notifications στο iOS δεν δουλεύαν σωστά. Για να λύσω αυτό χρειαζόταν να κάνω export ένα νέο certificate και να το κάνω sign με την πλατφόρμα της apple.
- Η εφαρμογή δεν δούλευε όταν δεν υπήρχε ιντερνέτ. Για την επίλυση αυτού εφαρμόστηκε cache έτσι ώστε να θυμάται παλιά responses και να τα δείχνει στους επαγγελματίες
- Refactor στον κώδικα γιατί θύμιζε spaghetti code, για αυτό χρησιμοποιήθηκαν good design patterns της google και της apple.

Increase security with JWT (android, iOS):

Ο τομέας της ασφάλειας αποτελεί αναπόσπαστο στοιχείο των εφαρμογών. Έτσι λοιπόν πρότεινα να προσθέσουμε JWT verification στα υπάρχοντα webservice μας. Αυτό θα μείωνε τον κίνδυνο κάποιος κακόβουλος χρήστης να κάνει χρήση των webservice με σκοπό να προκαλέσει πρόβλημα στο κομμάτι του backend.
Έτσι λοιπόν πρόσθεσα σε κάθε webservice https calls το JWT token στην εφαρμογή iOS και Android.

```
// working
@GET(Prefix + "app/professional_suspend_account.php")
Call<ResponseBody> getTotalBalance(
    @Query("member_id") String member_id
);

// working
//http://upgrade.myconstructor.gr/webservices/api/professional/getCalendar.php
@GET(Prefix + "app/get_calendar_busy.php")
Call<ResponseBody> getGoogleCalendarEventList(
    @Header("Authorization") String authKey, ←
);

// working
//http://upgrade.myconstructor.gr/webservices/api/professional/getCalendar.php
@GET(Prefix + "professional/getCalendar.php")
Call<ResponseBody> getGoogleCalendarEventListRange(
    @Query("prof_id") String prof_id,
    @Query("startDate") String start_date,
    @Query("endDate") String end_date
);

// working
@FormUrlEncoded
@POST(Prefix + "app/add_busy_time.php")
Call<ResponseBody> createGoogleCalendarEvent(
    @Header("Authorization") String authKey, ←
    @Field("busy_date") String busyDate,
    @Field("busy_time") String busyTime
);

// working
@FormUrlEncoded
@POST(Prefix + "app/delete_busy_time_by_date.php")
Call<ResponseBody> deleteGoogleCalendarEventsByDate(
    @Header("Authorization") String authKey, ←
    @Field("date") String busyDate
);

// working
@FormUrlEncoded
@POST(Prefix + "app/is_connected_with_gc.php")
Call<ResponseBody> isConnectedWithG(
    @Field("prof_id") String profId
);
```

Rest API for telephone-center WebRTC(Backend):

Ένα από τα μεγαλύτερα κατορθώματα ήταν η δημιουργία ενός τηλεφωνικού κέντρου από το μηδέν. Η αλήθεια είναι ότι στην αρχή κάτι τέτοι φαινόταν δύσκολο ωστόσο με σωστό planning καταφέραμε σαν ομάδα και το βγάλαμε στο production code.

Στο WebRTC το οποίο όπως λέει και το όνομα του αποτελεί προϊόν του web browser οι agents δέχονται κλήσεις από πελάτες καθώς επίσης γίνονται και εξερχόμενες.

Ιδιαίτερο ενδιαφέρον ωστόσο είχε η λειτουργία κατασκόπου η οποία επέτρεπε σε κάποιον agent να ακούσει την συνομιλία κάποιου αλλού agent καθώς μιλούσε. Αυτό μπορεί να φανεί αρκετά χρήσιμο σε καινούργιους agents που δεν γνωρίζουν καλά την δουλειά και θέλουν να αποκτήσουν κάποια εμπειρία ακούγοντας άλλους.

Για την δημιουργία του χρειαστήκαμε :

- Google VM
- Asterisk server
- Rest API
- Html, php, JS, nodeJS, JSSIP

*Το Asterisk αποτελεί γνωστό server για την δημιουργία PBX(τηλεφωνικού κέντρου) καθώς αποτελεί open source και είναι από τα παλιότερα. Έτσι λοιπόν είναι αρκετά stable.

*Το google VM χρειάζεται γιατί πάνω σε αυτό γίνεται install το asterisk.

Για την υλοποίηση του WebRTC ήμουν υπεύθυνος για την δημιουργία REST API με το οποίο θα επικοινωνούσε τόσο ο asterisk server όσο και ο browser.

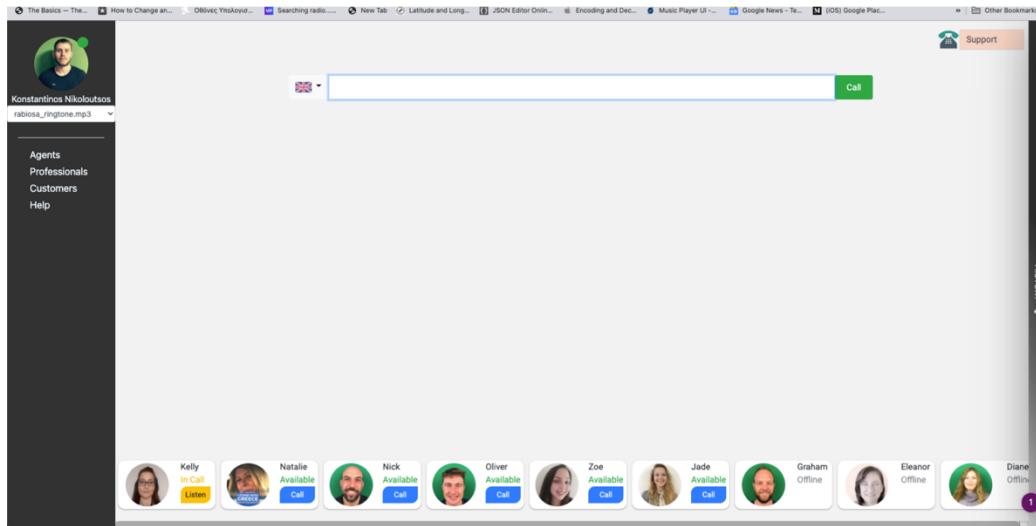
Για την υλοποίηση του χρησιμοποιήθηκαν:

- PHP
- NodeJS
- MySQL + Redis

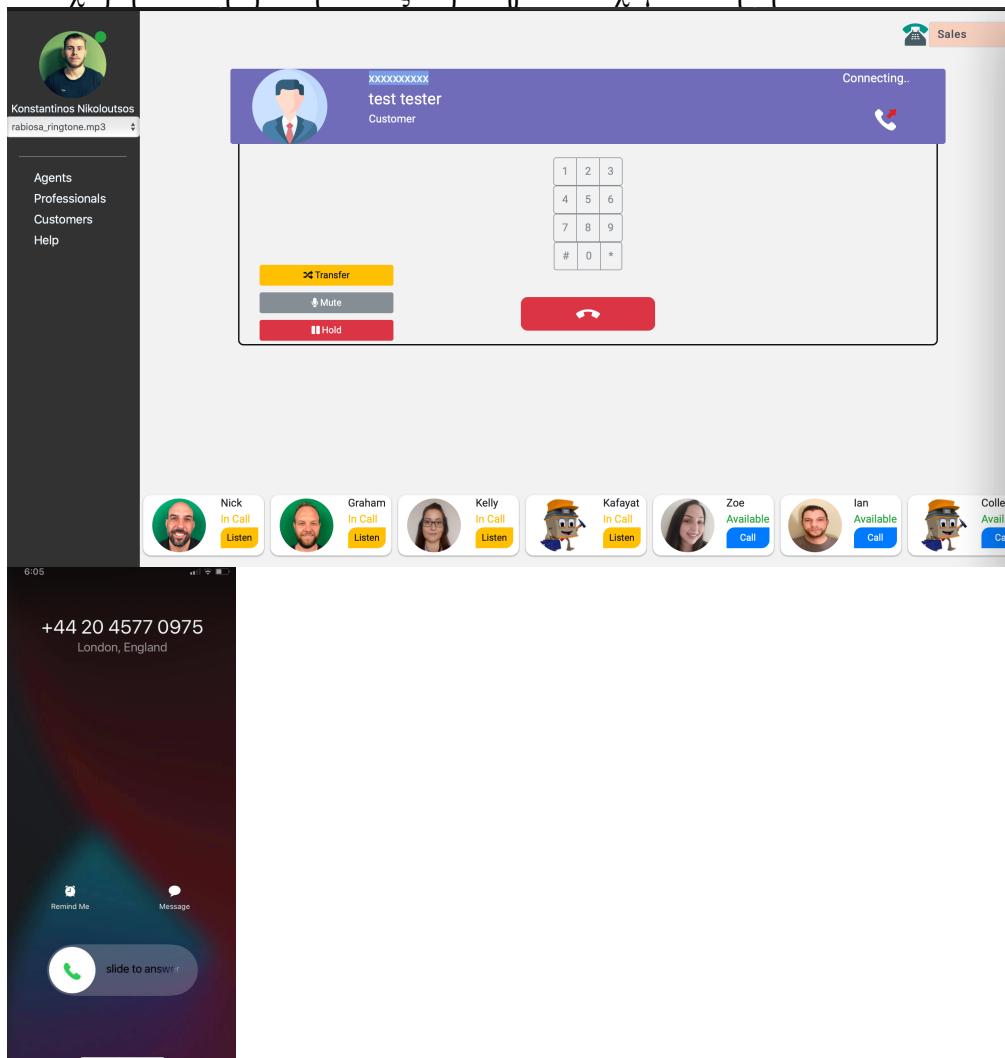
Το δύσκολο κομμάτι στην δημιουργία του Rest API δεν ήταν το routing αλλά πως θα φτιαχτεί η βάση δεδομένων για να είναι scalable και να μην επαναλαμβάνεται οι πληροφορία.

Με την αρκετή επικοινωνία μεταξύ των μελών της ομάδας ωστόσο αντιμετωπίστηκαν οι δυσκολίες και φτιάχτηκαν οι υποδομές για το WebRTC

Παρακάτω μπορείτε να δείτε κάποια screenshots από την χρήση του webRTC:



*Στο παρακάτω screenshot κάλεσα τον ενατό μου. Το xxxxxxxxx placeholder κρύβει το τηλέφωνο για λόγους ασφάλειας ενώ το test tester είναι το όνομα μου που με έχει καταχωρίσει στην βάση. Όπως παρατηρείτε δέχομαι κλήση από το webRTC



Εργασία από το σπίτι λόγω covid

Θεωρώ ότι πρέπει αν σχολιαστεί η συγκεκριμένη συνθήκη γιατί έπρεπε τόσο η εταιρία όσο και εγώ να προσαρμοστώ σε κάτι τέτοιο. Προτεραιότητα έχει η υγεία, γι' αυτό λοιπόν ένα μεγάλο διάστημα αναγκαστικά να δουλεύω από το σπίτι και τα meetings πλέον να γίνονται με virtual τρόπο αντί με φυσική παρουσία. Τα software που χρησιμοποιούσα για να γίνει αυτό πράξη ήταν:

- Microsoft teams
- Skype
- Slack

Καθημερινά κάναμε ένα σύντομο daily meeting στο οποίο συζητούσαμε για τα ενδεχόμενα προβλήματα που προέκυψαν αλλά και για το τι θα κάνουμε σήμερα. Με αυτόν τον τρόπο μπορούμε να πούμε ότι είμασταν όλοι συγχρονισμένη και ο καθένας γνώριζε τις αρμοδιότητες του.

Στην αρχή ομολογώ ότι ήταν λίγο περίεργο και δύσκολη η εναλλαγή από φυσική παρουσία σε εικονική αλλά στην συνέχεια προσαρμοστήκαμε. Μπορώ να πω ότι νιώθω και πιο παραγωγικός με την εργασία από το σπίτι!



Αποτελέσματα στόχων

Τα αποτελέσματα στόχων ήταν άμεσα και όπως φαίνεται αρκετά ικανοποιητικά

Αύξηση μέσου όρου χρήσης εφαρμογής	1minutes 53seconds σε 3minutes 22seconds
Μείωση των bugs	<p>Filter Event type = "Crashes" X</p> <p>Crash-free statistics</p> <p>Crash-free users ?</p> <p>99.31%</p>
Τηλεφωνικό κέντρο	Παραπάνω από 200+ τηλεφωνήματα καθημερινά
Real-time chat	Παραπάνω από 30+ chat καθημερινά

Τα παραπάνω αποτελέσματα προκύπτουν από το firebase analytics το οποιο μας βοηθάει να έχουμε ένα insight για την συμπεριφορά των χρηστών στην εφαρμογή μας!

Προβλήματα και αντιμετώπιση

Γενικά δεν υπήρξαν πολλά critical προβλήματα κατά την διάρκεια της πρακτικής μου άσκησης. Παρόλα αυτά εκείνα που μου έμειναν στο μυαλό ήταν τα παρακάτω

Λάθος συνεννόηση για git push :

Κατά την διάρκεια της πρακτικής μου αντιμετώπισα ένα σοβαρό πρόβλημα που πρόκειται από λανθασμένη συνεννόηση μεταξύ των developers.

Το πρόβλημα ήταν ότι καταλάθος είχα ανεβάσει κώδικα στο GitHub ενώ θα έπρεπε να γινόταν merge με κάποιον άλλο κώδικα ώστε να είναι λειτουργικό.

Ευτυχώς όμως το αντιληφθήκαμε γρήγορα και με τις λειτουργίες που μας παρέχει το version control (git) έκανα reverse των κώδικα με αποτέλεσμα να αφαιρεθεί από το production code. Στην συνέχεια ανέβηκε αφού έγινε merge με τον υπόλοιπο κώδικα.

Απόκτηση νέων γνώσεων :

Κάποιες από τις τεχνολογίες που χρησιμοποίησα για να φτιάξω κάποια features απαιτούσαν διάβασμα και παρακολούθηση tutorial.

Κάποιες από τις πηγές που χρησιμοποίησα για να μάθω περισσότερα ήταν:

- Google
- Udemy
- Stack Overflow

Στην αρχή μάλιστα ένα μεγάλο μέρος του χρόνου μου το αφιέρωνα στο να μάθω καινούργιες για εμένα έννοιες τις οποίες επρόκειτο να χρησιμοποιήσω στον κώδικα.

Πολλές φορές όμως δεν αρκεί μόνο η παρακολούθηση και γι' αυτό αναγκαζόμουν να φτιάχνω ένα περιβάλλον playground στο οποίο εξασκούσα αυτά που έβλεπα στα tutorial.

Επίλογος

Η εμπειρία μου στην πρακτική ήταν αρκετά ευχάριστη γιατί είδα στην πράξη πως χρησιμοποιούνται διάφορα εργαλεία προγραμματιστικού χαρακτήρα αλλά και οργανωτικού όπως για παράδειγμα το JIRA.

Κατά την διάρκεια αυτών των 4 μηνών part-time κατάφερα να βγάλω εις πέρας αρκετά tasks με την βοήθεια των συναδέλφων μου!

Η εταιρία έπειτα από την πρακτική μου έκανε προσφορά πρότασης εργασίας το οποίο θα το σκεφτώ καθώς έχω μια προσφορά από την Vodafone στον τομέα του iOS development.

Τέλος, όχι μόνο έμαθα καινούργιες τεχνικές γνώσεις αλλά γνώρισα καινούργια άτομα με κοινούς στόχους και επιθυμίες.

Η ευκαιρία της πρακτικής που δίνεται σε φοιτητές αποτελεί χρήσιμο εργαλείο και χαιρομai που πήρα μέρος σε κάτι τέτοιο.

Χρήσιμοι σύνδεσμοι

Τα παρακάτω links οδηγούν στα προϊόντα τα οποία έκανα contribute είτε φτιάχνοντας καινούργια features είτε διορθώνοντας προβλήματα και bugs.

Website:

www.myconstructor.co.uk

Android app:

https://play.google.com/store/apps/details?id=com.myconstructor&hl=en_US&gl=US

iOS app:

<https://apps.apple.com/gr/app/myconstructor-pro/id1277061499>

