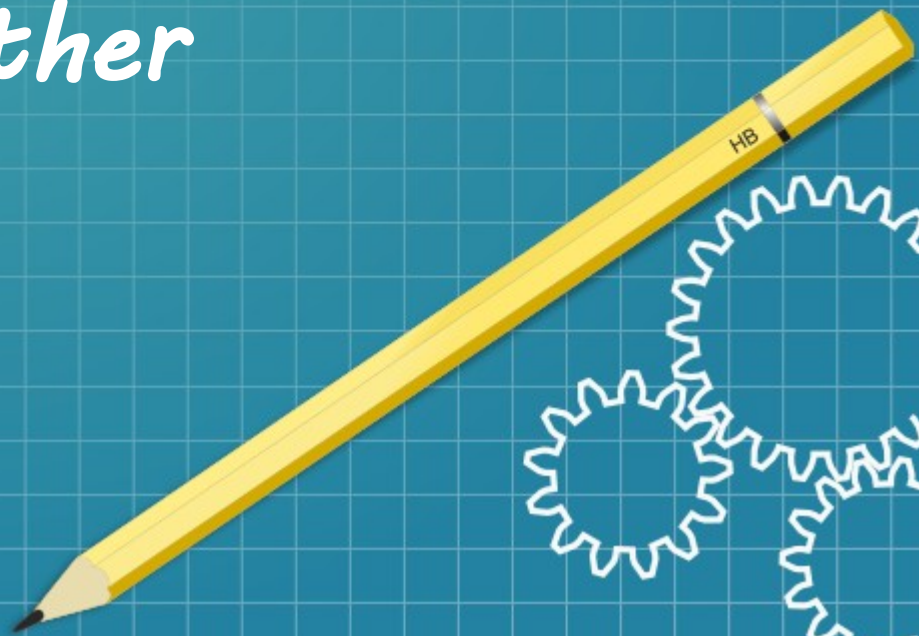
Σταματελόπουλος Νικόλαος
03116138

Internet and Applications
Project 2020

IP2Weather



Overview

Όπως περιγράφεται και στο README.md αρχείο στο github, αναπτύχθηκε μία εφαρμογή που επιτρέπει στο χρήστη την ανακάλυψη της τοποθεσίας μιας συσκευής γνωρίζοντας την IP της καθώς και τις καιρικές συνθήκες που επικρατούν.

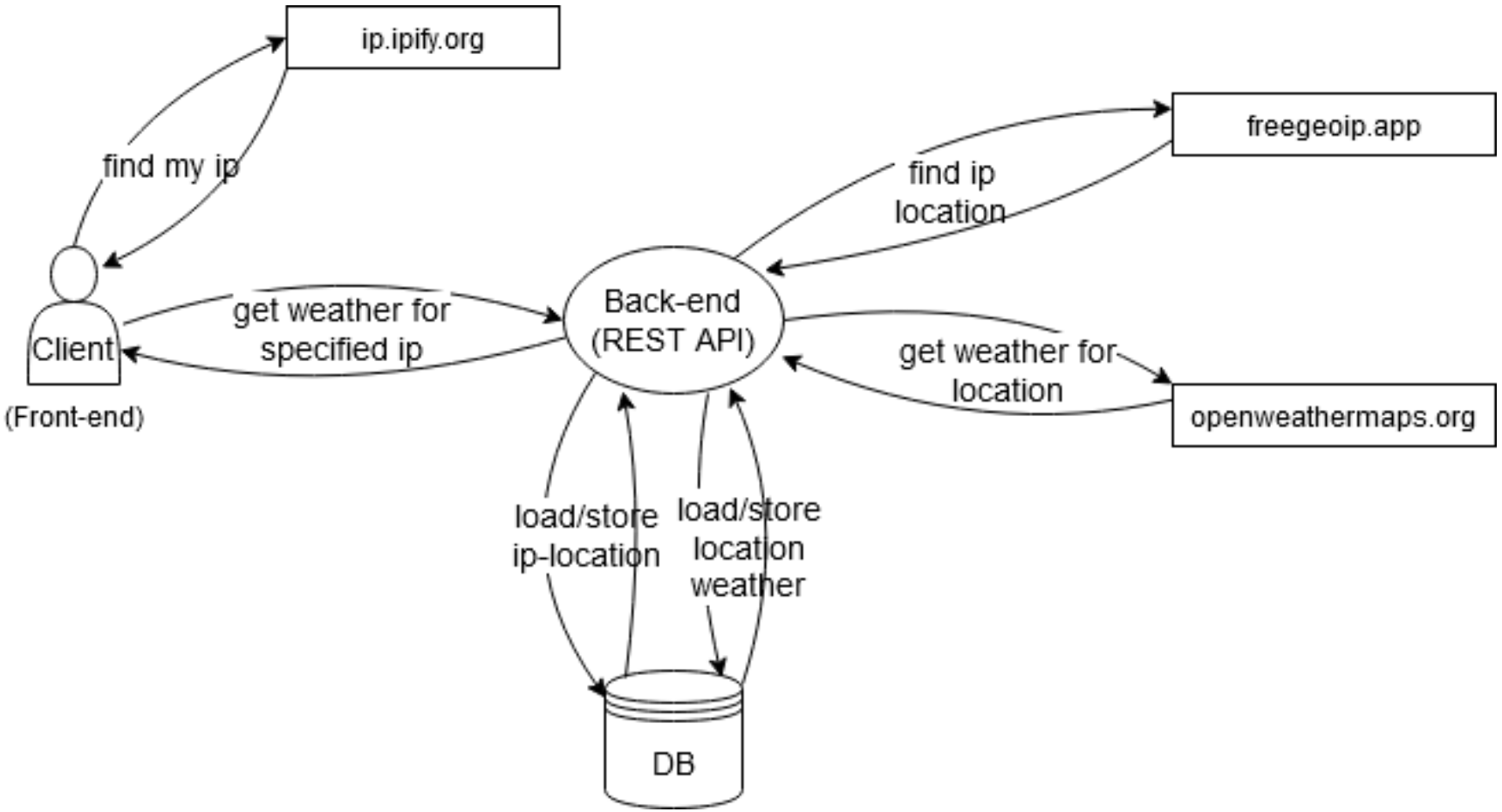
Η εφαρμογή αποτελείται από τρία μέρη:

- Front-end (html, css, JS)
- Back-end (JAVA)
- Database (MongoDB)

Ενώ γίνεται χρήση τριών (το πολύ) web APIs:

- https://api.ipify.org/?format=json
- https://freegeoip.app/json/
- http://api.openweathermap.org/data/2.5/onecall

General Control Flow (1)



General Control Flow (2)

Η γενική ροή ελέγχου στην εφαρμογή ακολουθεί τα εξής βήματα.

- Ανάκτηση της IP του χρήστη αν δεν έχει δοθεί ως είσοδος.
- HTTP get request στο REST API περνόντας την IP ως μεταβλητή στο τέλος του URL (επιπλέον ορίζεται η παράμετρος mediaType=xml για παραλαβεί του response σε xml format).
- Το REST API αναζητά αντιστοίχηση της ip σε τοποθεσία στη βάση. Αν δεν υπάρχει ή έχει “λήξει” (π.χ. 5 ώρες από τελευταία τροποίηση εγγραφής) τότε απευθύνεται στο αντίστοιχο web service.
- Γνωρίζοντας την τοποθεσία αναζητή στη βάση για καιρικές συνθήκες σε αυτή ακολουθώντας ίδια διαδικασία με πριν.
- Ανανεώνει τη βάση με την αντίστοιχη πληροφορία αν χρειάστηκε αναφορά στα web services.
- Αποστολή του response στο χρήστη με όλην την επιθυμητή πληροφορία που συλλέχθηκε.

(Ακολουθεί λεπτομερέστερη περιγραφή των σχεδιαστηκών αποφάσεων)

Database (MongoDB) (1)

Η βάση δεδομένων που δημιουργείται (ip2weather) περιέχει δύο collections:

- ip2location -> περιέχει objects με id την ip διεύθυνση επιπλέον πεδία για latitude, longitude και city της τοποθεσίας αυτής. Προφανώς η αναζήτηση γίνεται κάθε φορά βάση της IP address.
- city_weather -> περιέχει objects με id μια πόλη, βάση της οποίας γίνεται κάθε αναζήτηση, με επιπλέον πεδία weather (συνοπτικό description των παρόντων καιρικών συνθηκών), icon (όνομα icon.png που χρησιμοποιείται στο front end), temp (θερμοκρασία), pressure (βαρομετρική πίεση), humidity, wind_speed και wind_deg (κατεύθυνση ανέμου σε μοίρες).

Και στις δύο παραπάνω περιπτώσεις παρατίθεται και ένα πεδίο lastModifiedDate που φέρει προφανώς την ημερομηνία και ώρα τελευταίας τροποίησης για έλεγχο επικαιροποίησης των δεδομένων από το back-end που τα διαχειρίζεται (συνοδεύεται από ένα validtyPeriod για το πλήθος ωρών που παραμένουν έγκυρα τα δεδομένα).

Database (MongoDB) (2)

Ένα object του ip2location collection

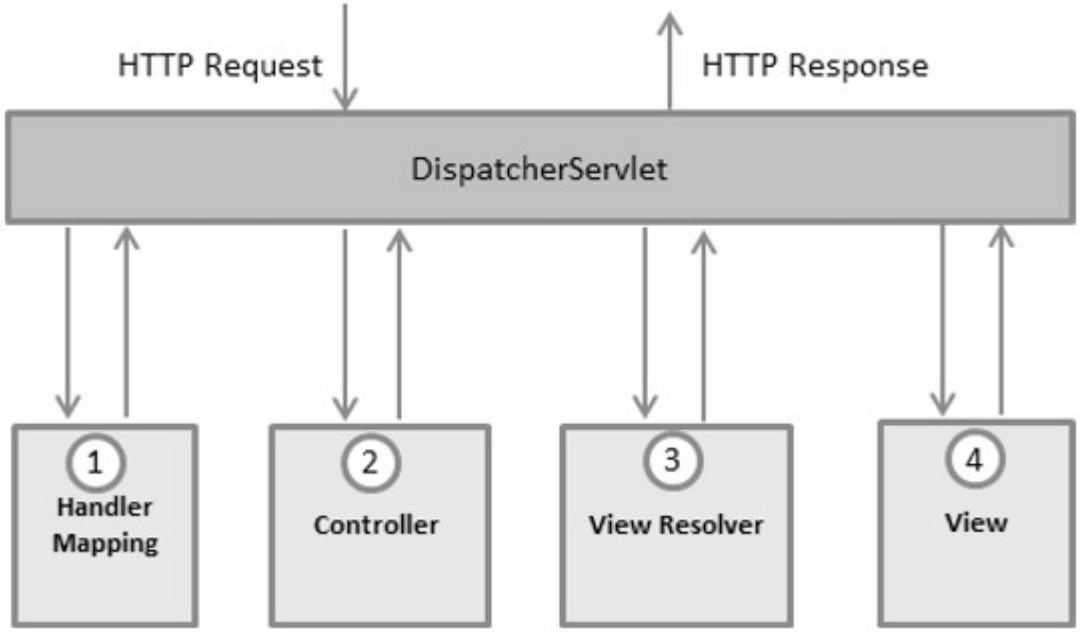

```
_id: "2.2.2.2"
latitude: "48.8582"
longitude: "2.3387"
city: "Paris 01 Louvre"
lastModifieDate: 2020-08-06T16:53:52.822+00:00
validityPeriod: 5
```


Ένα object του city_weather collection


```
_id: "Paris 01 Louvre"
weather: "clear sky"
icon: "01d"
temp: "306.99"
pressure: "1017"
humidity: "21"
wind_speed: "2.1"
wind_deg: "190"
lastModifieDate: 2020-08-06T16:53:53.096+00:00
validityPeriod: 1
```


Θεωρώ περίοδο ισχύως μιας εγγραφής στην πρώτη περίπτωση 1 ώρα (αν και δεν ήατν απαραίτητο καθώς αύξηση ακρίβειας geopisitioning ή τυχόν μεταβολές αυτού δε θα είχαν κάποιο σημαντικό αντίκτυπο στην εφαρμογή), ενώ στη δεύτερη περίπτωση θεωρήθηκε περίοδος μίας ώρας προσεγγίζοντας real-time προγνώσεις.

Back-end (1)



Εκμεταλευόμενοι το Spring web MVC framework (παραπάνω γραφική αναπαράσταση) αποφεύγουμε το boilerplate coding χειρισμού των HTTP requests. Αρκεί ο ορισμός μίας class που ορίζεται ως Controller στο μοντέλο ενώ η κατάλληλη μέθοδος που θα χειρίζεται τα εισερχόμενα http get requests στο url http://localhost:8080/ip2weather . Η ίδια μέθοδος φροντίζει για την άντληση των δεδομένων από τα web APIs και τη βάση (δεδομένου ύπαρξης μόνο ενός endpoint στο REST API είναι πρακτικά η “καρδιά” της εφαρμογής).

Back-end (2)

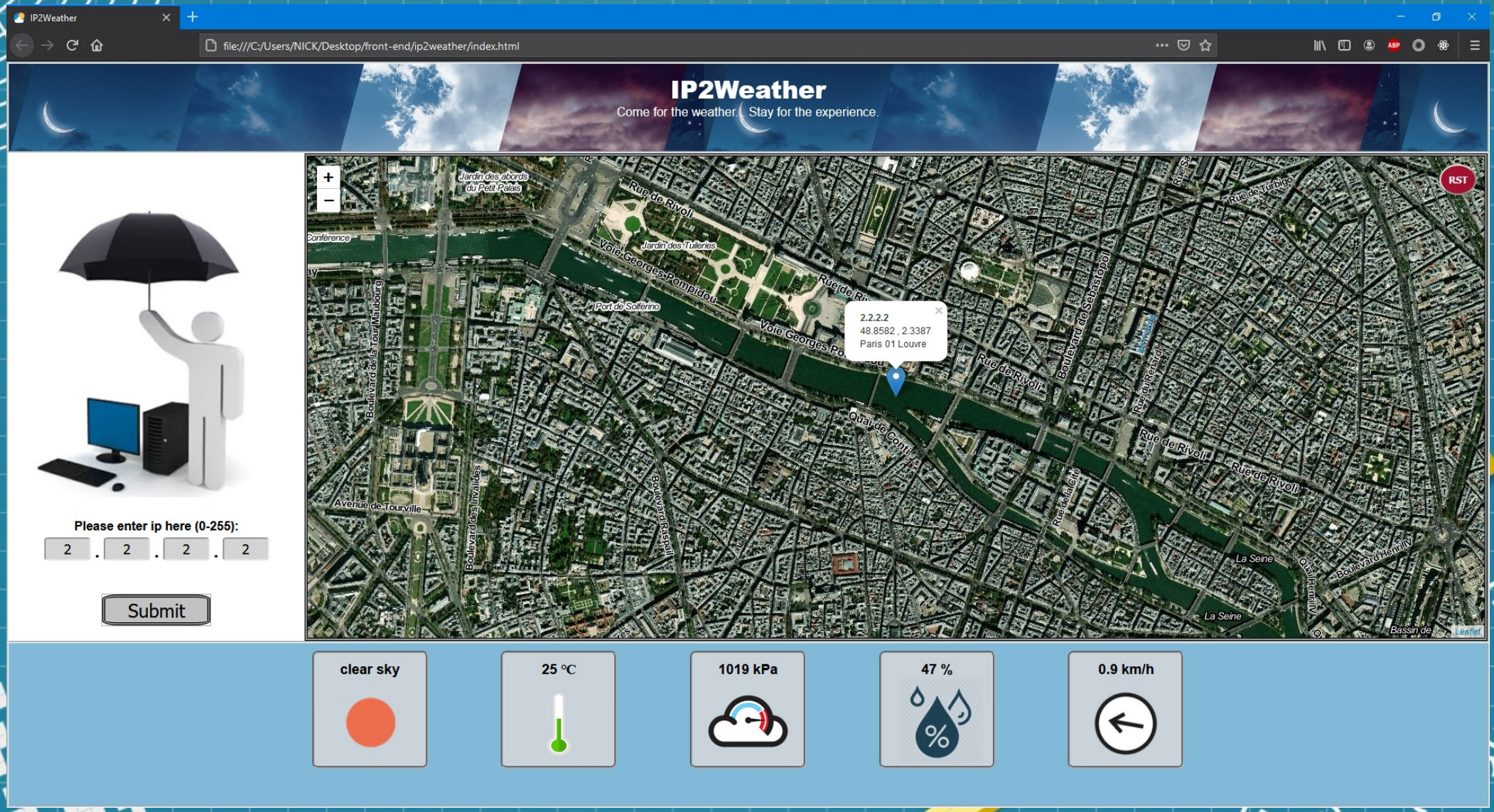
Λόγω χρήσης 2 διαφορετικών web APIs με κοινό στοιχείο δισύνδεσης των πληροφοριών το όνομα της πόλης, παρατηρήθηκαν σημαντικές ασυμφωνίες στην ονοματοδοσία. Αυτό λύθηκε αλλάζοντας το πεδίο city μιας από των δύο εγγραφών στη βάση ώστε να εξασφαλίζεται η συνοχή. Δίνεται λοιπόν προτεραιότητα στο όνομα που παίρνουμε από το ip-geolocation ενώ σε περίπτωση επιστροφής invalid ονόματος (πατηρήθηκε επιστροφή κενής συμβολοσειράς), λαμβάνουμε ως σημείο αναφοράς το όνομα από το openweathermaps.org (στην περίπτωση αυτή επαναλαμβάνεται η αναζήτηση βάση των συντεταγμένων αυτή τη φορά).

Το back-end χρησιμοποιεί ένα interface (spring.data.mongodb) για τη γρήγορη και εύκολη διαχείριση της βάσης δεδομένων (αποφυγή boilerplate του JDBC). Η επικοινωνία με τη βάση επιτυγχάνεται μέσω του ορισμού και της χρήσης δύο MongoRepository interfaces (ένα για κάθε collection).

Τέλος, ορίζεται ένας “global” exception handler για την ανάκτηση από σφάλματα και την επιστροφή κατάλληλων διαγνωστικών μυνημάτων, ανάλογα της προέλευσης του σφάλματος.

Το τελευταίο συνοδεύεται από τον ορισμό ενός βοηθητικού exception για τον έλεγχο εισόδου από το χρήστη, αλλά και ενός custom error που υλοποιείται απο μια class με τα κατάλληλα πεδία και επιστρέφεται ως απάντηση στο χρήστη σε περίπτωση εμφάνισης σφάλματος.

Front-end (1)


IP2Weather
file:///C:/Users/NICK/Desktop/front-end/ip2weather/index.html
IP2Weather
Come for the weather. Stay for the experience.
Please enter ip here (0-255):
2 . 2 . 2 . 2
Submit
2.2.2.2
48.8582 , 2.3387
Paris 01 Louvre
RST
clear sky
25 °C
1019 kPa
47 %
0.9 km/h

Front-end (2)

Όπως φαίνεται στο προηγούμενο screenshot, η σελίδα και οι λειτουργίες της είναι αυτεξήγητες και αρκετά εύκολες στη χρήση. Ο χάρτης είναι interactive, επιτρέποντας την περιήγηση σε όλη την υδρόγειο (εισαγωγή με χρήση leaflet βιβλιοθήκης), ενώ προστέθηκε ένα κόκκινο RST button στο άνω δεξί άκρο του για εύκολη επαναφορά και εστίαση στο τρέχον σημείο ενδιαφέροντος στο χάρτη.

Για την κύρια λειτουργεία αρκεί η συμπλήρωση της φόρμας στα αριστερά που αποτελείταιο από 4 πεδία (επιτρέπεται επιπλέον να αφεθούν **όλα** κενά). Δέχεται μόνο ακεραίους αριθμούς με το πολύ 3 ψηφία (υλοποίηση με js listeners), ενώ γίνεται και έλεγχος στο τέλος της εισαγωγής ότι είναι ακέραιος σε εύρος 0-255 με επιστροφή ανάλογου μηνύματος σε ενδεχόμενο παραβίασης της συνθήκης. Η υποβολή της ip και εκτέλεση της αναζήτησης επιτυγχάνεται με το κουμπί “Submit”.

Front-end (3)

Η παρουσίαση των αποτελεσμάτων γίνεται όπως φαίνεται στα 5 κουτιά στο κάτω μέρος της οθόνης, καθώς και με την εμφάνιση ενός popupστο χάρτη για την τοποθεσία (και την εστίαση σε αυτό).

Τα 2 πρώτα και το τελευταίο εικονίδια ανανεώνονται ανάλογα με την πληροφορία που παρουσιάζεται με χρήση κατάλληλων js scripts (στο αρχείο scripts.js).

Γι την επικοινωνία με το back-end χρησιμοποιείται XMLHttpRequests (AJAX), όμως λόγω της ασύχρονης φύσης των αιτήσεων απαιτείται μια ένδειξη στο χρήστη της επιτυχούς υποβολής του αιτήματος. Αυτό επιτυγχάνεται με απενεργοποίση του submit button (αποφυγή συσσώρευσης πολλαπλών αιτημάτων ταυτόχρονα), και αντίστοιχης γραφικής ένδειξης (και ενός loading animation) όπως φαίνεται στο ακόλουθο screenshot. Τέλος ορίζεται και αντίστοιχος έλεγχος εγκυρότητας της απάντηση από το server, ενώ πριν έχουν τεθεί request timeouts σε περίπτωση αδυναμίας επικοινωνίας με το αντίστοιχο service.

Front-end (4)


IP2Weather
file:///C:/Users/NICK/Desktop/front-end/ip2weather/index.html
IP2Weather
Come for the weather... Stay for the experience.
Please enter ip here (0-255):
46 . 198 . 199 . 115
Loading...
RST
--
-- °C
-- hPa
-- %
--km/h

Conclusion

Το μεγαλύτερο τμήμα του κώδικα συνοδεύεται από επαρκή σχόλια για εύκολη ανάγνωση και κατανόησή του.

- Περισσότερες λεπτομέρειες για την υλοποίηση και τις σχεδιαστικές αποφάσεις σε βίντεο στο ακόλουθο link:

  https://youtu.be/DP6kWuXwtY0

- Οδηγίες εγκατάστασης στο αρχείο README.md:

  https://github.com/nickst74/InternetApplications/blob/master/README.md

Σας ευχαριστώ
για το χρόνο σας!