ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Σύστημα Αναζήτησης Πτήσεων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΖΟΥΚΑ ΚΩΝΣΤΑΝΤΙΝΟΥ

Εξεταστική επιτροπή

Δεληγιαννάκης Αντώνιος, Αναπληρωτής Καθηγητής Π.Κ. (Επιβλέπων)

Μανιά Αικατερίνη, Αναπληρώτρια Καθηγήτρια Π.Κ.

Λαγουδάκης Μιχαήλ, Αναπληρωτής Καθηγητής Π.Κ.

Η σελίδα αυτή είναι σκόπιμα κενή.

Περίληψη

Η εργασία βασίστηκε στην ιδέα της ανάπτυξης ενός λογισμικού συστήματος για την αποτελεσματική και ευφυή αναζήτηση οικονομικών εισιτηρίων για αεροπορικές διαδρομές. Η πρωτοτυπία της εφαρμογής έγκειται στη χρήση τεχνικών διαδικτυακής εξόρυξης (web scraping) για την ανάλυση των πληροφοριακών σελίδων υφιστάμενων πλατφορμών κρατήσεων αεροπορικών εισιτηρίων και την εξόρυξη δεδομένων από αυτές. Παράλληλα, η εφαρμογή επεκτείνει την επερώτηση ενός χρήστη, προτείνοντας όχι μόνο εισιτήρια για απευθείας πτήσεις, αλλά και για εναλλακτικές πτήσεις με ενδιάμεσους σταθμούς. Οι ενδιάμεσοι σταθμοί προσδιορίζονται από τον χρήστη με βάση το μέγεθος των αεροδρομίων τους και την απόστασή τους από τον αρχικό και τον τελικό προορισμό.

Η επαύξηση της λειτουργικότητας οδηγεί στην αυτοματοποιημένη ανάκτηση επιπλέον ιστοσελίδων από πλατφόρμες κρατήσεων, ώστε ο μηχανισμός web scraping να ανακτά τιμές εισιτηρίων και για τα επιμέρους σκέλη της διαδρομής.

Για την υλοποίηση της εφαρμογής χρησιμοποιήθηκαν προγραμματιστικές τεχνικές και τεχνολογίες αιχμής, ώστε οι αναζητήσεις να διεκπεραιώνονται στον ελάχιστο δυνατό χρόνο και να μην καταναλώνονται αυξημένοι υπολογιστικοί πόροι.

Χρησιμοποιήθηκαν τεχνολογίες ασύγχρονου προγραμματισμού στο διακομιστή, RESTful διαδικτυακές υπηρεσίες για πρόσβαση στις λειτουργίες από το πρόγραμμα – πελάτη, καθώς και NoSQL βάση δεδομένων για την αποθήκευση και ανάλυση των δεδομένων των πτήσεων.

Τέλος, έλαβαν χώρα επιτυχημένες δοκιμές όσον αφορά τη λειτουργικότητα της πλατφόρμας με την χρήση δεδομένων από εμπορικά διαθέσιμο ιστότοπο εύρεσης αεροπορικών εισιτηρίων.

Abstract

This senior thesis proposes the idea of developing a software system for the efficient and intelligent search of affordable airline tickets. The originality of the application lies in the use of web scraping techniques which analyze the information extracted from pages of existing ticket booking platforms. Moreover, the application extends the user's queries, by proposing not only direct flights but also alternate flights with various stopovers. The intermediate connection options are customized in terms of their number and their maximum distance from the midpoint of the total trip.

The enrichment of the search functionality is achieved via automated retrieval of additional information pages from other booking platforms. In this way, the web scraping mechanism can retrieve ticket prices for each of the individual parts of the route.

For the implementation of the application, cutting-edge programming techniques and technologies were used in order to ensure that searches are carried out in the shortest possible time, without the need of large amount of computing resources.

Furthermore, asynchronous scheduling technologies were used on the server, RESTful web services for access to the functionality from the client application as well as NoSQL database technologies for storing and analyzing the flight data.

Finally, the functionality of the platform was successfully tested with data extracted by a commercially available flight ticket website.

Πίνακας περιεχομένων

Πίνακας Εικόνων

1 Εισαγωγή

Οι ανάγκες για αεροπορικές μετακινήσεις και προγραμματισμό δρομολογίων και εισιτηρίων διεκπεραιώνονται πλέον σε μεγάλο βαθμό διαδικτυακά, μέσα από καθιερωμένες πλατφόρμες αναζήτησης εισιτηρίων και κρατήσεων. Στην πλειονότητά τους οι πλατφόρμες αυτές συγκροτούνται από συνεταιριζόμενες αεροπορικές εταιρείες, οι οποίες προσφέρουν το προϊόν τους (δηλαδή το αεροπορικό εισιτήριο) κυρίως βάσει συμφωνιών και περιορισμών που υποβάλλουν οι ίδιες.

Συνεπώς, δημιουργείται η ανάγκη ανάπτυξης ενός μηχανισμού ελεύθερης αναζήτησης αεροπορικών δρομολογίων, που ως στόχο έχει την προσφορά της πιο οικονομικής πρότασης προς τον επιβάτη. Ως εκ τούτου, στη συγκεκριμένη διπλωματική εργασία προτείνεται ένα σύστημα το οποίο βασίζεται σε τεχνολογίες εξόρυξης δεδομένων από ιστοσελίδες (web scraping), το οποίο δίνει τη δυνατότητα εξαγωγής πληροφοριών ελεύθερα, από ιστοσελίδες διάθεσης αεροπορικών εισιτηρίων. Προκειμένου οι υπηρεσίες του προτεινόμενου συστήματος να διαφοροποιούνται σε σχέση με τις υπάρχουσες πλατφόρμες κρατήσεων εισιτηρίων, ενσωματώνει σχετικό αλγόριθμο για την πρόταση διαδρομών έως δύο ενδιάμεσων σταθμών. Η δυνατότητα μετεπιβίβασης συχνά έχει ως αποτέλεσμα την απόκτηση εισιτηρίων σε χαμηλότερη τιμή σε σύγκριση με πτήσεις χωρίς ενδιάμεσους σταθμούς.

Επιπροσθέτως, σημαντική καινοτομία για το προτεινόμενο σύστημα αποτελεί το γεγονός ότι δίνει την επιλογή στο δυνητικό επιβάτη να επιλέξει το μέγεθος των αεροδρομίων στα οποία θα πραγματοποιούνται οι ενδιάμεσες στάσεις. Οι περισσότερες πλατφόρμες αναζητούν σταθμούς μετεπιβίβασης μόνο σε αεροδρόμια τα οποία χαρακτηρίζονται «μεγάλα» από άποψη επιβατικής κίνησης. Στην περίπτωση του προτεινόμενου συστήματος, ο χρήστης έχει τη δυνατότητα να επιλέξει το μέγεθος των πιθανών ενδιάμεσων αεροδρομίων στα οποία το σύστημα θα αναζητήσει εισιτήρια για τον τελικό προορισμό. Με αυτό τον τρόπο εντοπίζονται πιθανοί ενδιάμεσοι σταθμοί σε μικρά αεροδρόμια, στα οποία οι φόροι αεροδρομίων ενδέχεται να είναι μικρότεροι και κατ' επέκταση το εισιτήριο να είναι οικονομικότερο. Γίνεται λοιπόν αντιληπτό ότι, το σύστημα έχει αυξημένες πιθανότητες να προτείνει σημαντικά οικονομικότερα εισιτήρια.

Εν συνεχεία, το προτεινόμενο σύστημα ενσωματώνει ακόμη μία παράμετρο αναζήτησης αεροπορικών δρομολογίων, επιτρέποντας τον ορισμό ακτίνας για εύρεση ενδιάμεσων αεροδρομίων. Ο χρήστης έχει τη δυνατότητα επιλογής της ακτίνας στην οποία το σύστημα

θα ψάξει για ενδιάμεσους προορισμούς, αυξάνοντας έτσι την πιθανότητα ανεύρεσης κάποιου οικονομικότερου εισιτηρίου , ενώ παράλληλα, δίνεται η δυνατότητα πειραματισμού σε διαφορετικές ακτίνες, οι οποίες καθορίζουν την απόσταση των ενδιάμεσων σταθμών από το αεροδρόμιο αφετηρίας και το αεροδρόμιο προορισμού.

Με σκοπό την ανεξαρτητοποίηση από τις προγραμματιστικές διεπαφές (Application Programming Interfaces) που προσφέρουν οι εμπορικές πλατφόρμες κρατήσεων, η προσέγγιση που ακολουθήθηκε είναι αυτή της εξόρυξης πληροφοριών άμεσα από το HTML περιεχόμενο των ιστοσελίδων, τις οποίες επιστρέφει κάθε πλατφόρμα στον τελικό χρήστη της. Η τεχνική αυτή είναι ευρύτερα γνωστή ως web scraping και χρησιμοποιείται σήμερα σε εφαρμογές τόσο εμπορικού χαρακτήρα όσο και σε διάφορα επιστημονικά πεδία για την υποστήριξη της έρευνας. Η υλοποίηση συμπεριέλαβε ένα συνδυασμό προγραμματιστικών πλαισίων και βιβλιοθηκών λογισμικού, ενώ υιοθέτησε τη λύση του ασύγχρονου προγραμματισμού για τη διαχείριση των HTML σελίδων που προκύπτουν από τις επερωτήσεις του χρήστη.

Το σύστημα υποβλήθηκε σε δοκιμές σε πραγματικές συνθήκες λειτουργίας και απέδειξε ότι σε πολλές περιπτώσεις οδηγεί στην εξεύρεση πιο οικονομικών εισιτηρίων λόγω των παραμέτρων που χρησιμοποιεί στις αναζητήσεις, αλλά και στην υπέρβαση των περιορισμών που τίθενται από τις πολιτικές των αεροπορικών εταιρειών, οι οποίες προωθούνται μέσα από τις εμπορικές πλατφόρμες κράτησης εισιτηρίων.

2 Επισκόπηση Σχετικής Έρευνας

Όπως προαναφέρθηκε, στη συγκεκριμένη διπλωματική εργασία προτείνεται ένα σύστημα το οποίο βασίζεται σε τεχνολογίες εξόρυξης δεδομένων από ιστοσελίδες (web scraping). Οι υπάρχουσες προσεγγίσεις για την υλοποίηση web scraping μπορούν να διακριθούν σε τρεις κύριες κατηγορίες: (i) βιβλιοθήκες οι οποίες επεκτείνουν γενικού σκοπού γλώσσες προγραμματισμού, (ii) προγραμματιστικά πλαίσια και (iii) αυτόνομα περιβάλλοντα. Στη συνέχεια γίνεται επεξήγηση των διαφορετικών κατηγοριών υλοποιήσεων web scraping και επισημαίνονται οι διαφορές τους.

Βιβλιοθήκες

Πρόκειται για την πιο συνήθη προσέγγιση των ερευνητών οι οποίοι αναπτύσσουν web scrapers στο πλαίσιο κάποιου ερευνητικού έργου χρησιμοποιώντας τη γλώσσα προγραμματισμού με την οποία είναι περισσότερο εξοικειωμένοι. Στην περίπτωση αυτή, η λογική του πράκτορα λογισμικού και το τελικό αποτέλεσμα υλοποιούνται ως συμβατικά προγράμματα μιας γλώσσας προγραμματισμού (Java, Python, κ.ο.κ.), χρησιμοποιώντας τις δομές δεδομένων και ελέγχου της εκάστοτε γλώσσας. Συνηθέστερα, βιβλιοθήκες τρίτων διαχειρίζονται την πρόσβαση στον ιστότοπο υλοποιώντας την πλευρά του πελάτη στο πρωτόκολλο HTTP, ενώ τα ανακτημένα περιεχόμενα από τις ιστοσελίδες αναλύονται με τις ενσωματωμένες συναρτήσεις διαχείρισης συμβολοσειρών, όπως κανονικές εκφράσεις (regular expressions), διάκριση λέξεων (tokenization) και καθαρισμό (trimming). Επιπρόσθετα, βιβλιοθήκες τρίτων υλοποιούν ευφυή ανάλυση περιεχομένου, όπως δημιουργία της δενδρικής HTML μορφής του εγγράφου και υλοποίηση αναζήτησης δεδομένων μέσω HTML ετικετών (XPath). (Glez-Peña, et al., 2014)

Ενδεικτικά αναφέρεται η libcurl η οποία αποτελεί μια διαδεδομένη βιβλιοθήκη για πρόσβαση σε ιστότοπους. Υποστηρίζει τα περισσότερα χαρακτηριστικά του πρωτοκόλλου HTTP, όπως πιστοποιητικά SSL, HTTP POST, HTTP POST, HTTP PUT, μεταφόρτωση FTP, cookies και HTTP authentication. Η ίδια βιβλιοθήκη προσφέρει συνδέσεις με πολλαπλές γλώσσες προγραμματισμού.

Επιπλέον, η Perl, γλώσσα η οποία χρησιμοποιείται ευρύτατα για ανάλυση δεδομένων ιδίως στις βιοεπιστήμες προσφέρει τη λειτουργική μονάδα WWW::Mechanize για την υλοποίηση λειτουργιών στον παγκόσμιο ιστό. Επιτρέπει τη διεπαφή με διευθύνσεις πόρων του ιστού, με

υποστήριξη HTTP, cookies, HTTP authentication και διαχείριση ιστορικού. Η Perl διαχειρίζεται επερωτήσεις περιεχομένου με γλώσσα XPath με τη χρήση πρόσθετων λειτουργικών ενοτήτων.

Με τη σειρά της η Java χρησιμοποιεί το πακέτο HttpClient, το οποίο προσομοιώνει τις κεντρικές ιδιότητες του πρωτοκόλλου HTTP (αιτήσεις, cookies, SSL και HTTP authentication) (Jacarta Commons, 2008). Για την ανάλυση του περιεχομένου και την ανάκτηση δεδομένων χρησιμοποιεί βιβλιοθήκες ανάλυσης όπως η jsoup. επίσης, η Java υποστηρίζει αναζητήσεις XPath και συνεργάζεται με πολλαπλές βιβλιοθήκες ανάλυσης HTML περιεχομένου, όπως η htmlcleaner.

Η γλώσσα Python χρησιμοποιεί τη βιβλιοθήκη BeautifulSoup για την ανάλυση HTML περιεχομένου. Η Python διαθέτει ενσωματωμένες εντολές για τη διαχείριση συνδέσεων πρωτοκόλλου HTTP. (Crummy, 2004)

Σε περιβάλλοντα Unix μπορούν να δημιουργηθούν web scrapers μέσω διοχέτευσης (piping) εντολών του λειτουργικού συστήματος. Προγράμματα όπως το curl (libcurl) και wget υλοποιούν το επίπεδο πελάτη του πρωτοκόλλου HTTP, ενώ βοηθητικά προγράμματα τύπου grep, awk, sed μπορούν να αναλύσουν με αποδοτικό τρόπο περιεχόμενο το οποίο διοχετεύεται σε αυτά. (Glez-Peña, et al., 2014)

Στην περίπτωση που υλοποιούνται πράκτορες λογισμικού οι οποίοι εκτελούνται στο διακομιστή ως διαδικτυακές εφαρμογές, τότε η υλοποίησή τους γίνεται με κάποια γλώσσα server-side, όπως PHP, Perl ή Java.

Προγραμματιστικά πλαίσια

Καταρχάς, είναι εύλογο να αναφερθεί ότι η χρήση μιας γενικού σκοπού γλώσσας προγραμματισμού για τη δημιουργία πρακτόρων λογισμικού εμπεριέχει κάποια μειονεκτήματα. Συνήθως πρέπει να συνδυαστούν πολλαπλές βιβλιοθήκες, τόσο για την πρόσβαση σε διαδικτυακούς πόρους όσο και για την ανάλυση και ανάκτηση στοιχείων του περιεχομένου εγγράφων HTML. Επιπλέον, οι πράκτορες οι οποίοι αναπτύσσονται για web scraping είναι ευπαθείς σε τροποποιήσεις των HTML σελίδων που προσπελαύνουν και απαιτούν διαρκή συντήρηση. Σε γλώσσες οι οποίες διαθέτουν μεταφραστή (compiler) όπως η Java, οποιαδήποτε τροποποίηση του κώδικα του πράκτορα απαιτεί εκ νέου μετάφραση και πολλές φορές επανεγκατάσταση του συνόλου της εφαρμογής στο διακομιστή.

Τα προγραμματιστικά πλαίσια προσφέρουν πιο ολοκληρωμένη λύση για την ανάπτυξη μηχανισμών web scraping. Ενδεικτικά, το πλαίσιο Scrapy για τη γλώσσα Python προσφέρει όλα τα προγραμματιστικά εργαλεία για την ανάπτυξη εφαρμογών web scraping. Με τη χρήση του πλαισίου Scrapy οι πράκτορες δημιουργούνται ως απόγονοι της κλάσης BaseSpider, η οποία ορίζει ένα σύνολο URLs εκκίνησης και μία συνάρτηση “parse” η οποία

καλείται σε κάθε επαναληπτική εκτέλεση του αλγόριθμου. Οι ιστοσελίδες αναλύονται αυτόματα και το περιεχόμενο ανακτάται με τη χρήση εκφράσεων XPath.

Άλλα προγραμματιστικά πλαίσια χρησιμοποιούν εξειδικευμένες γλώσσες προγραμματισμού για το συγκεκριμένο πεδίο. Τέτοιου είδους προγραμματιστικό πλαίσιο είναι το Web-Harvest, το οποίο συνεργάζεται με τη γλώσσα Java. Οι διαδικασίες ανάκτησης περιεχομένου από διαδικτυακούς πόρους εκφράζονται σε γλώσσα XML και απαρτίζονται από αγωγούς διοχέτευσης, οι οποίοι διαβιβάζουν τα αποτελέσματα ενός βήματος στο επόμενο. Παράλληλα, το πρόγραμμα διαθέτει κλάσεις για τη διεκπεραίωση των απαραίτητων λειτουργιών: "http" για την πρόσβαση σε διαδικτυακούς πόρους, "html-to-xml" για τον καθαρισμό του HTML περιεχομένου και "xpath" για την ανάκτηση περιεχομένου βάσει κριτηρίων.

Αντίστοιχο πλαίσιο της γλώσσας Java είναι το jARVEST, το οποίο επίσης διαθέτει μια εξειδικευμένη γλώσσα ειδικού σκοπού και χρησιμοποιεί το πλαίσιο JRuby για πιο συμπαγή υλοποίηση των πρακτόρων.

Ο Πίνακας 1 περιέχει σύνοψη και σύγκριση διαδεδομένων βιβλιοθηκών και πλαισίων web scraping

Πίνακας 1. Βιβλιοθήκες και πλαίσια ανοικτού κώδικα για web scraping

	Type				**Extraction facilities**
	C: HTTP client				**R: Regular expressions**
	P: Parsing				**H: HTML parsed tree**
					X: XPath
	F: Framework	**Domain-specific language**	**API/stand alone (SA)**	**Language**	**C: CSS selectors**
UNIX shell (curl/wget, grep,	**CP**	**OXI**	**SA**	**bash**	**R**

	Type	Domain-specific language	API/standalone (SA)	Language	Extraction facilities
	C: HTTP client				R: Regular expressions
	P: Parsing				H: HTML parsed tree
	F: Framework				X: XPath
					C: CSS selectors
sed, cut, paste, awk)					
Curl/libcurl	C	OXI	API/SA	C + bindings	
Web-Harvest	F	OXI	API/SA	Java	RX
Jsoup	CP	OXI	API	Java	HC
HttpClient	C	OXI	API	Java	
jARVEST	F	NAI	API/SA	JRuby/Java	RXC
WWW::Mechanize	CP	OXI	API	Perl	RX
Scrapy	F	OXI	API/SA	Python	RX
BeautifulSoup	P	OXI	OXI	Python	H

Πηγή : Web scraping technologies in an API world , Daniel Glez-Peña Anália Lourenço Hugo López-Fernández Miguel Reboiro-Jato Florentino Fdez-Riverola, Briefings in Bioinformatics, Volume 15, Issue 5, 1 September 2014, Pages 788–797

Στον πίνακα περιλαμβάνονται έξι βιβλιοθήκες, οι οποίες υλοποιούν έναν πελάτη HTTP (C) και ανάλυση εγγράφων HTML. Επίσης περιλαμβάνονται τρία προγραμματιστικά πλαίσια (σημασμένα με (F)).

Εφαρμογές web scraping

Οι εφαρμογές web scraping απευθύνονται σε χρήστες, οι οποίοι ενδιαφέρονται να εξορύξουν πληροφορίες από ιστότοπους, αλλά δεν έχουν τις προγραμματιστικές γνώσεις να αναπτύξουν εφαρμογές σε κάποια γλώσσα προγραμματισμού. Τέτοιου είδους προγράμματα προσφέρουν γραφική διεπαφή η οποία διευκολύνει τη -δημιουργία και συντήρηση των πρακτόρων. Συνήθως διαθέτουν έναν ενσωματωμένο φυλλομετρητή τον οποίο ο χρήστης χρησιμοποιεί για να φορτώσει την επιθυμητή σελίδα και να ορίσει διαδραστικά τα στοιχεία τα οποία θα ανακτηθούν, αποφεύγοντας με αυτόν τον τρόπο την αναγκαιότητα ορισμού των κανονικών εκφράσεων ταιριάσματος των όρων (regular expressions) ή εκφράσεων XPath. Επιπροσθέτως, διαθέτουν την απαραίτητη λειτουργικότητα για τη δημιουργία διαφόρων μορφότυπων εξόδου, όπως αρχεία CSV, XML ή ακόμη και Excel, αλλά και για την εισαγωγή των ανακτημένων δεδομένων σε βάσεις δεδομένων.

Στα βασικά μειονεκτήματα των εφαρμογών web scraping συμπεριλαμβάνεται το υψηλό κόστος απόκτησης, δεδομένου ότι αποτελούν εμπορικά εκμεταλλεύσιμα προϊόντα. Ακόμη, σε πολλές περιπτώσεις προσφέρουν περιορισμένες προγραμματιστικές διεπαφές, γεγονός το οποίο δυσχεραίνει την ενσωμάτωση αυτών των scrapers σε άλλες εφαρμογές. Ο Πίνακας 2 παρουσιάζει συγκριτικά επτά εμπορικά προϊόντα web scraping.

Πίνακας 2. Εμπορικά διαθέσιμα πακέτα web scraping

Πηγή : Web scraping technologies in an API world , Daniel Glez-Peña Anália Lourenço Hugo López-Fernández Miguel Reboiro-Jato Florentino Fdez-Riverola, Briefings in Bioinformatics, Volume 15, Issue 5, 1 September 2014, Pages 788–797

	IrobotSoft	Visual Web Ripper	Newbie	Mozenda	Screen-scraper	WebSundew	FMiner
Software type							
License	Free-ware	Com-mer-cial	Commer-cial	Com-mer-cial	Free-ware (Basic edi-tion)	Com-mer-cial	Com-mer-cial
Open source	No	No	No	No	No	No	No
Plat-forms	Win	Win	Win	Win	Win Linux Mac	Win	Win
Site ac-cess							
Form POST	Yes	Yes	Yes	Yes	Yes	No	Yes
Session	Yes	Yes	Yes	Yes	Yes	No	Yes

	IrobotSoft	Visual Web Ripper	Newbie	Mozenda	Screen-scraper	WebSundew	FMiner
Conf. user agent	IE	IE and 2 in-ternal UAs	IE Firefox	IE	No	Fire-fox and 1 inter-nal UA	No
Iteration over pages	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Anony-mizer Proxies	Yes	Yes	N/A	No	No	N/A	Yes
Formats							
Input formats	.irb	.rip	Webpages URL list	.xml	.sss	.zws	.sep
Output formats	Text	CSV	Text	CSV	Text	CSV	CSV
	CSV	XML	Excel	TSV	CSV	XML	Excel
	XML	DB	DB	XML	DB	Excel	DB
	DB	Excel		Excel			

	IrobotSoft	Visual Web Ripper	Newbie	Mozenda	Screen-scraper	WebSundew	FMiner
Robot file format	**.irb**	**.rip**	**.nbs scripts**	**.xml**	**.sss**	**.zws**	**.sep**
Runtime							
Multi-threading	**Yes**	**Yes**	**Yes**	**No**	**Yes**	**Yes**	**Yes**
Progressive results	**Yes**	**No**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Design environment							
GUI-based designer	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
API access	**No**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**No**
Scriptable	**Yes**	**Limited**	**Yes**	**No**	**Yes**	**N/A**	**No**

3 *Θεωρητικό Υπόβαθρο*

Data Scraping

Η σύγχρονη εποχή διακρίνεται από ραγδαία τεχνολογική ανάπτυξη, αιχμιακό σημείο της οποίας είναι η εξέλιξη του διαδικτύου και η δραστικότατη διάχυση της πληροφορίας. Καθημερινά, διακινούνται και δημοσιεύονται στο διαδίκτυο (Castrillo-Fernández, 2015) εκατομμύρια bytes πληροφορίας σε διάφορες μορφές. Οι πιο συχνές από αυτές είναι οι HTML, PDF, XML, κ.α. Την τελευταία δεκαετία ο ρυθμός με τον οποίο παράγονται δεδομένα έχει πολλαπλασιαστεί εκθετικά χρίζοντας σκόπιμο να δημιουργηθούν εργαλεία για την επισκόπηση, την επεξεργασία, την εξαγωγή και τη μετατροπή τους σε άλλες μορφές, ώστε να χρησιμοποιούνται αποτελεσματικά και έξυπνα από διάφορους μελετητές, αναλυτές, προγραμματιστές κ.α. με στόχο να καλύψουν πολλαπλές ανάγκες.

Το data scraping είναι μια τεχνική με την οποία αποσπάμε δεδομένα από μία ορισμένη πηγή της οποίας η έξοδος προορίζεται για προβολή σε χρήστη. Είναι εύλογο, ότι τα εν λόγω δεδομένα δεν είναι εύκολα αναλύσιμα, ούτε έχουν σωστή τεκμηρίωση. Παρ' όλα αυτά , αναγκαζόμαστε να τα εξάγουμε με scraping για λόγους που θα αναλυθούν περαιτέρω σε επόμενα κεφάλαια. Καταληκτικά , ορισμένες από της μεθόδους απόσπασης δεδομένων είναι το screen scraping, web scraping, report mining κ.α. Το παρόν πόνημα αποπειράται να επεξηγήσει τη λειτουργία του web scraping.

Web Scraping

Το web scraping (εξόρυξη δεδομένων ιστού) εμπεριέχει ένα εύρος τεχνικών για την ανάκτηση και εξόρυξη δεδομένων από ιστοσελίδες, για την πραγματοποίηση των οποίων χρησιμοποιούνται τεχνικές εντοπισμού των ιστοσελίδων, διευθυνσιοδότησής τους, επεξεργασίας του μη δομημένου περιεχομένου τους και τέλος, εξαγωγής των δομημένων δεδομένων που ενδιαφέρουν την εκάστοτε εφαρμογή. Από τα παραπάνω προκύπτει, ότι μια πηγή δεδομένων μπορεί να είναι είτε ένα αποθετήριο αρχείων στο διαδίκτυο, είτε μια στατική HTML ιστοσελίδα, είτε σύνθετα έγγραφα τα οποία παράγονται δυναμικά από ένα διακομιστή διαδικτύου (όπως για παράδειγμα μια ενότητα σχολίων σε ένα δημοσιευμένο άρθρο μιας διαδικτυακής εφημερίδας) (Castrillo-Fernández, 2015).

Σε κάθε περίπτωση, υπάρχουν διαθέσιμα διαφορετικά εργαλεία, τα οποία χρησιμοποιούνται κατά περίπτωση από διαφορετικά έργα web scraping. Ωστόσο, λόγω της ποικιλότητας των πλατφορμών διαχείρισης και διάθεσης περιεχομένου, αλλά και της συχνότητας εναλλαγής της δομής των σελίδων, δεν είναι δυνατή η εύρεση ενός εργαλείου για την ολοκληρωτική διεκπεραίωση ενός έργου web scraping. (Black, 2016)

Ειδικότερα σε περιπτώσεις στις οποίες τα δεδομένα των ιστοσελίδων έχουν αυστηρή και καλά δομημένη παρουσίαση (όπως οι σελίδες των ηλεκτρονικών καταστημάτων), οι τεχνικές web scraping αποδεικνύονται ιδιαίτερα αποδοτικές για τη δεικτοδότηση περιεχομένου από τις μηχανές αναζήτησης, τις πλατφόρμες σύγκρισης τιμών και τις πλατφόρμες διαφήμισης. Μεγάλες επιχειρήσεις του διαδικτύου (όπως η Google) αναπτύσσουν και εξελίσσουν δικές τους τεχνολογίες web scraping. Μάλιστα, πλέον υπάρχουν διαθέσιμα εργαλεία ανοικτού κώδικα τα οποία διατίθενται ελεύθερα στην προγραμματιστική κοινότητα για τη δημιουργία υπηρεσιών εξόρυξης πληροφοριών για συγκεκριμένες εφαρμογές (οικονομική ανάλυση, διευκόλυνση του καταναλωτή στη σύγκριση προϊόντων και τιμών, κ.ο.κ.) (Black, 2016).

Συνεχίζοντας, οι τεχνικές web scraping εστιάζουν στην εξόρυξη συγκεκριμένων τύπων δομημένων δεδομένων από μη δομημένα έγγραφα του διαδικτύου. Στο σημείο αυτό θα πρέπει να γίνει η αντιδιαστολή μεταξύ κακόβουλου και μη-κακόβουλου web scraping. Η χρήση τεχνολογίας για την υποκλοπή δεδομένων με σκοπό την εμπορική τους χρήση προσκρούει σε νομικά ζητήματα και δεν θεωρείται αποδεκτή. Αντιθέτως, η χρήση τεχνικών εξόρυξης δεδομένων για επιστημονικούς και ερευνητικούς σκοπούς μπορεί να ωφελήσει πολλαπλά τους φορείς που διεξάγουν την αντίστοιχη έρευνα. (Boyd, et al., 2016)

Η μη κακόβουλη χρήση web scraping μπορεί να διακριθεί σε δύο βασικές κατηγορίες:

1. Τυπικά υποστηριζόμενη εξόρυξη δεδομένων
2. Τυπικά επιτρεπόμενη εξόρυξη δεδομένων

Η τυπικά υποστηριζόμενη εξόρυξη δεδομένων διεξάγεται μέσω μιας προγραμματιστικής διεπαφής (Application Programming Interface - API) ή ενός προγραμματιστικού πλαισίου (SOAP / REST web services) το οποίο χρησιμοποιείται αποκλειστικά για τη διεκπεραίωση και την απόκριση σε αιτήσεις παροχής δεδομένων σε ένα πρόγραμμα – πελάτη (συνήθως ένα φυλλομετρητή ο οποίος υπέβαλλε το αίτημα). Σε πρακτικό επίπεδο τέτοιου είδους διεπαφές λειτουργούν μέσω της υποβολής αιτημάτων στο διακομιστή στη μορφή εντολών και παραμέτρων σε μια συμβολοσειρά URL. Ένα τέτοιο αίτημα επιστρέφει ένα δομημένο σύνολο δεδομένων με συγκεκριμένη γραμμογράφηση και όχι ένα μη δομημένο έγγραφο σε μορφή HTML. Οι ιστότοποι οι οποίοι υποστηρίζουν τέτοιου είδους διεπαφές παρέχουν και την αντίστοιχη τεκμηρίωση στους προγραμματιστές, ώστε αυτοί να γνωρίζουν επακριβώς τις εντολές και τις παραμέτρους που θα πρέπει να διαβιβάσουν στο διακομιστή για να λάβουν το επιθυμητό αποτέλεσμα. Επιπλέον, οι συγκεκριμένοι ιστότοποι ενδέχεται να απαιτούν μία

διαδικασία εγγραφής των προγραμματιστών, ώστε να διατηρούν τα ίδια επίπεδα ασφάλειας αλλά και να περιορίζουν ενδεχομένως τον όγκο τον δεδομένων που μπορεί να λάβει κάθε εφαρμογή. (Black, 2016)

Αντίθετα με την τυπικά υποστηριζόμενη εξόρυξη δεδομένων, το web scraping στοχεύει σε πηγές δεδομένων οι οποίες άτυπα επιτρέπουν την εξόρυξη των δεδομένων τους, υπό την έννοια ότι ο ιστότοπος ούτε ρητά επιτρέπει ούτε ρητά απαγορεύει την αυτοματοποιημένη ανάκτηση των δεδομένων που δημοσιεύει στο διαδίκτυο. Ο όρος "web scraping" συναντάται αρκετές φορές και ως "screen scaping", δεδομένου ότι οι περισσότερες τεχνικές πρώτα ανακτούν τις HTML σελίδες στην οθόνη (σε μορφή που να μπορεί να αναγνωσθεί από έναν άνθρωπο θα τη διαβάζει ένα άνθρωπος) και στη συνέχεια εφαρμόζουν σε αυτές φίλτρα στον HTML κώδικα, ώστε να εξαχθούν συγκεκριμένες και επιθυμητές πληροφορίες. Ως εκ τούτου, το web scraping απαιτεί την εξέταση του πρωτογενούς εγγράφου (το οποίο αποτελεί μια μονάδα σε μια ευρεία συλλογή ομοειδών εγγράφων) και την ανάπτυξη ενός ευρεστικού αλγόριθμου ο οποίος απομονώνει από το συνολικό περιεχόμενο μόνο εκείνα τα δεδομένα τα οποία ενδιαφέρουν τον ερευνητή. Από το παραπάνω συνάγεται, ότι ο κάθε αλγόριθμος web scraping συνδέεται στενά με συγκεκριμένα σύνολα ιστοσελίδων. Προφανώς, ακόμη και αν δύο ομοειδείς ιστότοποι χρησιμοποιούν την ίδια πλατφόρμα διαχείρισης περιεχομένου, η σχεδίαση των ιστοσελίδων τους θα είναι διαφορετική, γεγονός το οποίο σημαίνει ότι χρησιμοποιούν διαφορετικές HTML ετικέτες και δομές. Αυτό έχει ως αποτέλεσμα ένα φίλτρο (ή αλγόριθμος) εξόρυξης το οποίο λειτουργεί στον έναν ιστότοπο να μη λειτουργεί στον άλλο. (Black, 2016)

Συνεπώς, οι τεχνικές web scraping ανοίγουν τεράστιους ορίζοντες στην εξόρυξη πληροφοριών από ιστότοπους, ακόμη και αν αυτοί δεν προσφέρουν προγραμματιστικές διεπαφές για την ανάκτηση των δομημένων δεδομένων τους. Σε κάθε περίπτωση, υπάρχουν σημαντικά τεχνικά αλλά και ηθικά / νομικά ζητήματα στη χρήση web scraping για ευρείας κλίμακας εφαρμογές.

3.2.1. Ορισμοί

Στη βιβλιογραφία έχουν προταθεί διάφοροι ορισμοί για το web scraping. Οι ορισμοί που παρατίθενται εδώ αναφέρονται στην εξόρυξη δεδομένων από πολλαπλές πηγές. Η διαφορά τους εντοπίζεται στο μορφότυπο της πηγής των δεδομένων που εξορύσσονται.

Ορισμός 1

"Υπάρχουν περιπτώσεις στις οποίες είναι απαραίτητη η συλλογή πληροφοριών από ιστότοπους, οι οποίες είναι έτσι δομημένες ώστε να απευθύνονται σε ανθρώπους-αναγνώστες και όχι σε πράκτορες λογισμικού. Η διαδικασία αυτή είναι γνωστή ως "web scraping". (Apress, 2009)

αυτός ο παραπάνω ορισμός αναφέρεται σε πηγές δεδομένων, οι οποίες αρχικά προορίζονται για ανάγνωση από τον άνθρωπο. Ο συγκεκριμένος ορισμός θεωρείται ξεπερασμένος, καθώς η έλευση αυτοματοποιημένων τεχνικών δημιούργησε τις προϋποθέσεις για εξόρυξη δεδομένων και από πηγές αναγνώσιμες από λογισμικό. Πρέπει όμως να ληφθεί υπόψη ότι το 2009 – όταν και προτάθηκε η διάδοση των προγραμματιστικών διεπαφών ήταν πολύ περιορισμένη. (Berlind, 2015)

Ορισμός 2

Το web scraping είναι μία τεχνική για την εξόρυξη δεδομένων από τον παγκόσμιο ιστό (WWW) και την αποθήκευσή τους σε ένα σύστημα αρχείων ή μια βάση δεδομένων για περαιτέρω διαχείριση ή ανάλυση. Τα διαδικτυακά δεδομένα εξορύσσονται με τη χρήση του πρωτοκόλλου Hypertext Transfer Protocol (HTTP) ή μέσω ενός φυλλομετρητή. Αυτό επιτυγχάνεται είτε χειροκίνητα είτε αυτόματα μέσω μιας μηχανής λογισμικού. Λαμβάνοντας υπόψιν ότι καθημερινά ένας τεράστιος όγκος ετερογενών δεδομένων δημιουργείται στον παγκόσμιο ιστό, το web scraping αναγνωρίζεται ευρέως ως μια αποδοτική και ισχυρή τεχνική για τη συλλογή μεγάλου όγκου δεδομένων. (Califf & Mooney, 1999).

Η τρέχουσα κατάσταση αποτυπώνεται με μεγαλύτερη ακρίβεια στον ορισμό αυτό, αφού το web scraping αναφέρεται ως πηγή για τη συλλογή μεγάλου όγκου δεδομένων (big data). Επίσης είναι αρκετά ακριβής όσον αφορά την τεχνολογία η οποία χρησιμοποιείται για την εξόρυξη δεδομένων μέσα από ιστοσελίδες με τη χρήση του πρωτοκόλλου HTTP.

Ορισμός 3

Ο τελευταίος ορισμός δεν είναι ιδιαίτερα λεπτομερής αλλά καταγράφει συνοπτικά τις λειτουργίες του web scraping με ακρίβεια.

“Το web scraping περιλαμβάνει τις διαδικασίες επερώτησης μιας πηγής πληροφορίας, ανάκτησης της σελίδας των αποτελεσμάτων και ανάλυσης της σελίδας με σκοπό την εξόρυξης των αποτελεσμάτων”. (Salerno & Boulware, 2003).

Προσέγγιση

Οι τεχνικές web scraping βασίζονται στη δομή και τις ιδιότητες της γλώσσας περιγραφής εγγράφων HTML. Ο εντοπισμός των επιθυμητών ετικετών και μορφότυπων στο σώμα ενός HTML εγγράφου, διευκολύνει τα λογισμικά web scraping στην ανάκτηση των αντίστοιχων δεδομένων. Τα εργαλεία web scraping αποθηκεύουν συνακόλουθα τα ανακτημένα δεδομένα σε μορφότυπους κατάλληλους για περαιτέρω επεξεργασία και ανάλυση όπως: JSON, XML, CSV, XLS ή RSS (Castrillo-Fernández, 2015).

Επιπλέον, η διαδικασία web scraping επιτρέπει την αυτοματοποίηση της εξόρυξης δεδομένων από ομοιόμορφες ιστοσελίδες. Η αυτοματοποίηση έγκειται στην παροχή μιας προγραμματιστικής διεπαφής (API) η οποία επιτρέπει τον ορισμό των δομικών στοιχείων της

HTML τα οποία σηματοδοτούν τα δεδομένα που πρέπει να ανακτηθούν από την ιστοσελίδα. Ο ορισμός αυτός χρησιμοποιείται από τη μηχανή web scraping για την ανάκτηση δεδομένων από περισσότερες από μία ιστοσελίδες της ίδιας δομής. Τα εργαλεία web scraping χρησιμοποιούν το URL της ιστοσελίδας, το οποίο δίνει πρόσβαση στο HTML έγγραφο από όπου πρόκειται να εξορυχθούν τα δεδομένα. Ο εν λόγω μηχανισμός είναι ιδιαίτερα αποδοτικός για ολοκληρωμένες Business-To-Business (B2B) διαδικασίες, καθώς τροφοδοτεί με δεδομένα άλλες επιχειρησιακές εφαρμογές και υπηρεσίες, οι οποίες βασίζονται στα δεδομένα που προκύπτουν από την εξόρυξη. Προκύπτει συνεπώς η αναγκαιότητα για την ύπαρξη μιας απλής προγραμματιστικής διεπαφής, η οποία θα επιτρέπει την προσαρμογή σε διαφορετικές κλάσεις ιστοσελίδων.

Διαδικασία εξόρυξης

Η διαδικασία web scraping απαιτεί τη χρήση ενός πράκτορα λογισμικού, ο οποίος προσιδιάζει στη χρήση ενός διαδικτυακού διακομιστή από έναν άνθρωπο, ο οποίος περιηγείται στις σελίδες του. Ο πράκτορας λογισμικού αιτείται από το διαδικτυακό διακομιστή όλες τις σελίδες οι οποίες είναι απαραίτητες για τη συλλογή των δεδομένων που του έχει ανατεθεί. Οι σελίδες αυτές διαβιβάζονται στη μορφή HTML εγγράφων, τα οποία ο πράκτορας αναλύει, ώστε να εντοπίσει και να εξάγει τα προκαθορισμένα δεδομένα. Τα δεδομένα που ανακτώνται αποθηκεύονται κατά περίπτωση σε κατάλληλες δομές (XML, JSON, σχεσιακά συστήματα). (Glez-Peña, et al., 2014) Οι προγραμματιστικές διεπαφές web scraping διευκολύνουν τις επαναλαμβανόμενες εργασίες, ώστε να διεκπεραιώνονται οι εκάστοτε στόχοι ανάκτησης δεδομένων, υλοποιώντας τα ακόλουθα βήματα:

Πρόσβαση στον ιστότοπο: η μηχανή web scraping εγκαθιδρύει επικοινωνία με τον ιστότοπο ο οποίος διαθέτει τις υπό εξέταση ιστοσελίδες χρησιμοποιώντας το πρωτόκολλο HTTP το οποίο και συντονίζει τη διαδικασία αιτήματος – απόκρισης μεταξύ του πελάτη (τυπικά ενός φυλλομετρητή) και του διαδικτυακού διακομιστή. Το πρωτόκολλο HTTP προβλέπει δύο μεθόδους υποβολής αιτημάτων: τη μέθοδο GET η οποία εφαρμόζεται για τα αιτήματα πόρων και τη μέθοδο POST η οποία χρησιμοποιείται για την αποστολή δεδομένων ηλεκτρονικών φορμών και τη μεταφόρτωση αρχείων. Το πρωτόκολλο HTTP προβλέπει μια σημαντική κεφαλίδα στα πακέτα αιτημάτων που διαβιβάζει, δεδομένου ότι ο διακομιστής εξετάζει τη συγκεκριμένη κεφαλίδα για να διαπιστώσει το είδος του προγράμματος το οποίο αιτείται πρόσβαση στα περιεχόμενα που διαχειρίζεται (φυλλομετρητής ή πράκτορας). Ανάλογα με το είδος του προγράμματος, ο διακομιστής τροποποιεί και τις απαντήσεις του προς το χρήστη που υπέβαλλε το αίτημα. Επιπροσθέτως, όπως όλοι οι πράκτορες λογισμικού στο web, έτσι και οι web data scrapers πρέπει να συμμορφώνονται στους «όρους χρήσης» του εκάστοτε ιστότοπου από όπου ανακτούν δεδομένα, όπως αυτοί δηλώνονται στο αρχείο "robots.txt". Στο αρχείο αυτό δηλώνονται οι πόροι οι οποίοι δεν πρέπει να προσπελαύνονται από αυτοματοποιημένες διαδικασίες. Επίσης, οι web scrapers πρέπει να ενεργοποιούνται κατά τρόπο τέτοιο

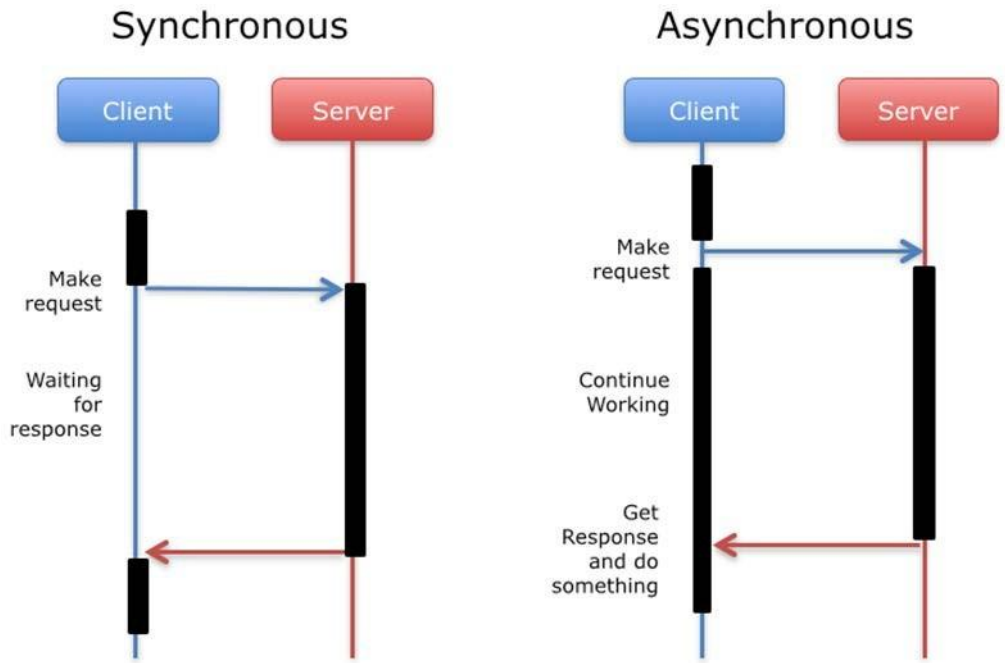
και σε χρόνο κατά τον οποίο θα αποφεύγεται υπερφόρτωση του διακομιστή. (Glez-Peña, et al., 2014)

Ανάλυση HTML περιεχομένου και εξαγωγή δεδομένων: αφού ανακληθεί το έγγραφο μορφής HTML, η μηχανή web scraping είναι σε θέση να εντοπίσει και να ανακτήσει τα δεδομένα που ενδιαφέρουν την εκάστοτε εφαρμογή. Συνήθως αυτό επιτυγχάνεται με την τεχνική ταιριάσματος "κανονικών εκφράσεων (regular expressions)", αλλά και με την επιπρόσθετη αλγοριθμική λογική. Εναλλακτικές μέθοδοι είναι η ανάλυση του HTML εγγράφου με χρήση του Document Object Model (DOM) με τη χρήση γλωσσών επερωτήσεων ετικετών HTML (όπως η XPath) (W3C, 1999). Μια γενική κατεύθυνση που ακολουθείται από τους web scrapers είναι η χρήση όσο το δυνατό γενικότερων εκφράσεων για το ταίριασμα της δομής του HTML εγγράφου, ώστε οι μηχανές web scraping να είναι λιγότερο ευπαθείς σε τροποποιήσεις του HTML εγγράφου. Οι τεχνικές εξόρυξης είναι πιο ανθεκτικές σε τροποποιήσεις, όταν ο ιστότοπος χρησιμοποιεί σημασιολογική σήμανση του περιεχομένου του (semantic markup). (Glez-Peña, et al., 2014)

Δεδομένα εξόδου: ο κεντρικός σκοπός του web scraping είναι η μετατροπή των ανακτημένων δεδομένων από ιστοσελίδες σε δομημένη αναπαράστασή τους, η οποία είναι κατάλληλη για περαιτέρω ανάλυση και αποθήκευση. Τα εργαλεία web scraping διαθέτουν μηχανισμούς για την παραγωγή δομών δεδομένων στη μνήμη ή σε αρχεία κειμένου (XML, JSON, CSV).

Ασύγχρονος προγραμματισμός

Ο ασύγχρονος προγραμματισμός πρωτοεμφανίστηκε στις αρχές του 21ου αιώνα και ήρθε να δώσει λύσεις στην απόκριση και στην ταχύτητα ενός συστήματος. Πιο αναλυτικά, με τον ασύγχρονο προγραμματισμό εξαλείφονται πολλές προβληματικές του σύγχρονου προγραμματισμού και της φιλοσοφίας του, χωρίς αυτό να σημαίνει ότι ο ασύγχρονος προγραμματισμός θα αντικαταστήσει το συμβατικό σύγχρονο τρόπο. Στο σύγχρονο προγραμματισμό ο κώδικας εκτελείται διαδοχικά, γραμμή προς γραμμή. Αυτό σημαίνει, ότι όταν καλείται μια διαδικασία το πρόγραμμα πρέπει να περιμένει μια απάντηση πριν ξεκινήσει την εκτέλεση της επόμενης εντολής. Αντίθετα, όταν εκτελείται μια διαδικασία ασύγχρονα, δεν υπάρχει αναμονή για απάντηση και εκτελείται κανονικά η επόμενη εντολή ή κάποια άλλη εργασία (task). Όταν είναι έτοιμη η απάντηση εμφανίζεται στον χρήστη, διαφορετικά η επιλογή του κώδικα που θα χρησιμοποιηθεί είναι στην ευχέρεια του προγραμματιστή. Στην εικόνα 3.1 παρουσιάζεται γραφικά η διαφορά στη λειτουργία της ασύγχρονης και σύγχρονης λογικής. Έτσι, ο ασύγχρονος προγραμματισμός παράγει αισθητά πιο γρήγορη απόκριση ενός προγράμματος και παρέχει πολύ καλή απόκριση σε front-end εφαρμογές. Στα πλεονεκτήματα του επίσης συγκαταλέγεται η καλύτερη κλιμάκωση (scalability) σε εφαρμογές που τρέχουν σε διακομιστές.



Εικόνα 3.1. *Σύγχρονος προγραμματισμός σε σχέση με το Ασύγχρονο*
Πηγή: https://medium.com/@santhoshhari/efficient-web-scraping-with-pythons-asynchronous-programming-6b9e730f1ff7

Είναι εύλογο, ότι διάφοροι παράγοντες οι οποίοι βασίζονται στην τεχνολογική εξέλιξη των υπολογιστικών συστημάτων, οδήγησαν στη χρήση αλλά και στην αναγκαιότητα ύπαρξης του σύγχρονου προγραμματισμού. Η τεχνολογική ανάπτυξη των κεντρικών μονάδων επεξεργασίας (processors), δημιουργεί το υπόβαθρο για αποτελεσματική χρήση του ασύγχρονου προγραμματισμού (Richard Blewett, 2013). Οι πολλαπλοί πυρήνες προσφέρουν μεγάλη υπολογιστική ισχύ. Χρησιμοποιώντας ένα ασύγχρονο προγραμματιστικό μοντέλο η πυροδότηση του κώδικα σε κάθε πυρήνα σε μη προκαθορισμένες χρονικά στιγμές, οδηγεί στην πλήρη αξιοποίηση όλων των διαθέσιμων πόρων.

Επιπλέον, οι απαιτήσεις των χρηστών για λογισμικό το οποίο αποκρίνεται ταχύτητα στις αιτήσεις τους έχουν αυξηθεί. Πιο αναλυτικά, οι χρήστες και οι επιχειρήσεις απαιτούν γρήγορη και χωρίς καθυστερήσεις διαχείριση πληροφορίας και περιήγηση στο διαδίκτυο καθώς και αυξημένη ανταποκρισιμότητα (responsiveness) στις ιστοσελίδες και το επιχειρησιακό λογισμικό. Για το λόγο αυτό οι προγραμματιστές οδεύουν στην υιοθέτηση μεθόδων ασύγχρονου προγραμματισμού (Rocha, et al., 2015).

Μηχανισμοί ασύγχρονης εκτέλεσης

Έχουν προταθεί τρία μοντέλα ασύγχρονου προγραμματισμού, τα οποία υλοποιούνται σε διαφορετικές γλώσσες προγραμματισμού και τεχνολογίες, ώστε να επιτύχουν ασύγχρονη εκτέλεση ενός προγράμματος (Richard Blewett, 2013).

- Πολλαπλοί κόμβοι: Σε αυτό το μοντέλο ένα αίτημα εξυπηρέτησης υποβάλλεται από έναν υπολογιστικό κόμβο. Στη συνέχεια το αίτημα αυτό διαβιβάζεται στη μορφή μηνύματος σε μια ουρά αναμονής. Το σύστημα περιλαμβάνει έναν κόμβο - εργάτη, ο οποίος είναι επιφορτισμένος με τη λήψη του μηνύματος και τη διαχείριση του μεταφερόμενου σε αυτό αιτήματος από το έργο. Όταν ολοκληρωθεί η εξυπηρέτηση, η διαδικασία επιστρέφει την απάντηση στον κόμβο που υπέβαλλε το αίτημα, επανεισάγοντας ένα αντίστοιχο μήνυμα σε μια ουρά αναμονής. Το μοντέλο προτείνει την υλοποίηση πολλαπλών κόμβων – εργατών, οι οποίοι εκτελούνται ανεξάρτητα ο ένας από τον άλλον. Η υλοποίηση αυτή επιτυγχάνει την ταχύτερη εξυπηρέτηση της ουράς αναμονής των αιτημάτων και συνεπώς την ταχύτερη εξυπηρέτηση του κόμβου που υπέβαλε ένα αίτημα.
- Πολλαπλές διεργασίες: το μοντέλο αυτό επιτυγχάνει ταυτόχρονη πρόσβαση στους πυρήνες επεξεργασίας μέσω πολλαπλών διεργασιών, οι οποίες δε μοιράζονται τον ίδιο χώρο μνήμης. Είναι τυπικά ανεξάρτητες και έχουν ξεχωριστούς χώρους διευθυνσιοδότησης. Επί της ουσίας χρησιμοποιείται η ίδια τεχνική ουράς αναμονής με το μοντέλο των πολλαπλών κόμβων, με τη μόνη διαφορά ότι εδώυπάρχει διαθέσιμη μόνο μία υπολογιστική μηχανή. Έτσι, επιτυγχάνεται καλύτερος συγχρονισμός διεργασιών και αντιμετώπισης σφαλμάτων.
- Πολλαπλά νήματα: Το νήμα είναι μια ακολουθία εντολών που εμπεριέχεται σε μια διεργασία και της οποίας η διαχείριση είναι ανεξάρτητη από το λειτουργικό σύστημα. Η πολυνημάτωση είναι ένα μοντέλο εκτέλεσης διεργασιών κύριο χαρακτηριστικό του οποίου είναι η ύπαρξη πολλαπλών νημάτων μέσα σε μία μόνο διεργασία και η ταυτόχρονη εκτέλεσή τους. Σε αντίθεση με τις διεργασίες, τα νήματα μπορούν να μοιράζονται τους ίδιους πόρους και την ίδια μνήμη. Τέλος, μοιράζονται τον ίδιο χώρο διευθυνσιοδότησης που τους παραχωρείται από την διεργασία και η εναλλαγή ανάμεσά τους γίνεται πολύ γρηγορότερα απ’ ότι αυτή ανάμεσα σε διαφορετικές διεργασίες.

4 Λειτουργικές Προδιαγραφές και UI Prototyping

Για την ανάπτυξη και τελική ολοκλήρωση μιας διαδικτυακής εφαρμογής απαιτείται κατάτμηση του λογισμικού στις δομικές του ενότητες καθώς και συνεργασία και συντονισμός μεταξύ των ομάδων που αναλαμβάνουν να το πραγματώσουν. Αρχικά, αναλύονται οι ανάγκες του χρήστη, έτσι ώστε να υπάρχει πλήρης εικόνα των απαιτήσεων της εφαρμογής και του τελικού της στόχου. Μέσα από ομαδική δουλειά σχεδιάζονται και οργανώνονται οι λειτουργικές και μη λειτουργικές απαιτήσεις, τίθενται οι περιορισμοί και το χρονοδιάγραμμα του έργου. Στη συνέχεια, με βάση όλα τα παραπάνω, οι σχεδιαστές αναλαμβάνουν να σχεδιάσουν την διεπαφή χρήστη (user interface-UI) κάνοντας συνεχώς αξιολόγησή της. Μόλις ολοκληρωθεί, οι προγραμματιστές ξεκινούν να χτίζουν την εφαρμογή, πάλι σύμφωνα με τις απαιτήσεις και ανάγκες του χρήστη.

Στο συγκεκριμένο κεφάλαιο περιγράφονται οι λειτουργικές προδιαγραφές της διαδικτυακής εφαρμογής εξόρυξης δεδομένων για αεροπορικά εισιτήρια και τα εργαλεία που θα χρησιμοποιηθούν για την υλοποίηση της διεπαφής χρήστη, της λειτουργικότητας του πελάτη και των διακομιστών της.

Λειτουργικές Απαιτήσεις

Η εφαρμογή που αναπτύχθηκε στο πλαίσιο της εργασίας στοχεύει στην γρήγορη και αποτελεσματική εύρεση και παρουσίαση φθηνών αεροπορικών εισιτηρίων, τα οποία να έχουν τη δυνατότητα να ικανοποιούν την απαίτηση του χρήστη για μετακίνηση από ένα αεροδρόμιο σε ένα άλλο σε συγκεκριμένες χρονικές ζώνες. Απευθύνεται σε χρήστες όλων των ηλικιών με στοιχειώδη -ακόμη και πολύ μικρή εξοικείωση με τους ηλεκτρονικούς υπολογιστές και το διαδίκτυο. Για το λόγο αυτό απαιτείται προσεκτική σχεδίαση της διεπαφής χρήστη, ώστε να είναι πλήρως κατανοητή και εργονομική, δεδομένου ότι υποστηρίζει τόσο υπολογιστές γραφείου και φορητούς όσο και κινητές μονάδες (tablets και smartphones). Οι απαιτήσεις από το περιβάλλον εργασίας συνοψίζονται στα εξής:

- Πλήρως ανταποκρίσιμη διεπαφή χρήστη (user interface).
- Ομαλή πλοήγηση της εφαρμογής.

- Ευπαρουσίαστη και εύχρηστη διεπαφή χρήστη.
- Απλουστευτικός και ορθός σχεδιασμός της διεπαφής, ώστε να γίνεται εύκολη χρήση της εφαρμογής όχι μόνο από επαγγελματίες αλλά και από χρήστες που δεν είναι εξοικειωμένοι με τέτοιου είδους εφαρμογές.
- Συμβατότητα με όλα τα σύγχρονα κινητά τηλέφωνα, tablets και φυλλομετρητές.

Personas

Οι εικονικές προσωπικότητες χρησιμοποιούνται στην εργασία για να διευκολύνουν τους σχεδιαστές και τους προγραμματιστές να καθορίσουν τις απαιτήσεις και τις ανάγκες του συστήματος και να προχωρήσουν στο σχεδιασμό της εφαρμογής. Ουσιαστικά, πρόκειται για εικονικούς χρήστες με ιδιότητες και ρόλους που ανταποκρίνονται σε πραγματικούς χρήστες που εμφανίζουν ενδιαφέρον για την εφαρμογή.

Jay , 32, Γιατρός

Ο Jay είναι γιατρός στο επάγγελμα και κατάγεται από τη Γερμανία. Λόγω της φύσης της δουλειάς του ταξιδεύει τουλάχιστον 2 φορές το μήνα. Γι' αυτό το λόγο θα ήθελε να ψάχνει για φθηνές αεροπορικές πτήσεις από το κινητό του τηλέφωνο άμεσα και γρήγορα.

Κώστας , 20 , Φοιτητής

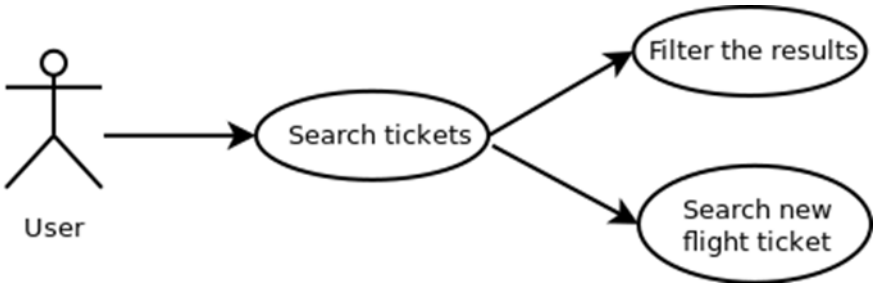
Ο Κώστας είναι φοιτητής στη σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών του Πολυτεχνείου Κρήτης. Του αρέσει να κανονίζει ταξίδια – εξορμήσεις με γνώμονα πάντα τα οικονομικά του. Δεν τον απασχολεί αν το ταξίδι μέχρι τον προορισμό θα διαρκέσει πολλές ώρες, αφού έχει ελεύθερο χρόνο, ούτε σε ποιές χώρες και αεροδρόμια υπάρχει περίπτωση να κάνει στάση, αρκεί να τον καλύπτει οικονομικά.

Μαρία, 60, συνταξιούχος

Η Μαρία είναι συνταξιούχος και περνά τον ελεύθερο χρόνο της κάνοντας ταξίδια με γίλους. Έχει μια πολύ βασική εξοικείωση με την τεχνολογία και την πλοήγηση στο διαδίκτυο. Το παραπάνω έχει ως αποτέλεσμα να χρειάζεται μία φιλική και κατανοητή διεπαφή χρήστη, ώστε να διεκπεραιώσει την αναζήτηση χωρίς προβλήματα, γρήγορα και αποτελεσματικά.

Περιπτώσεις χρήσης (Use cases)

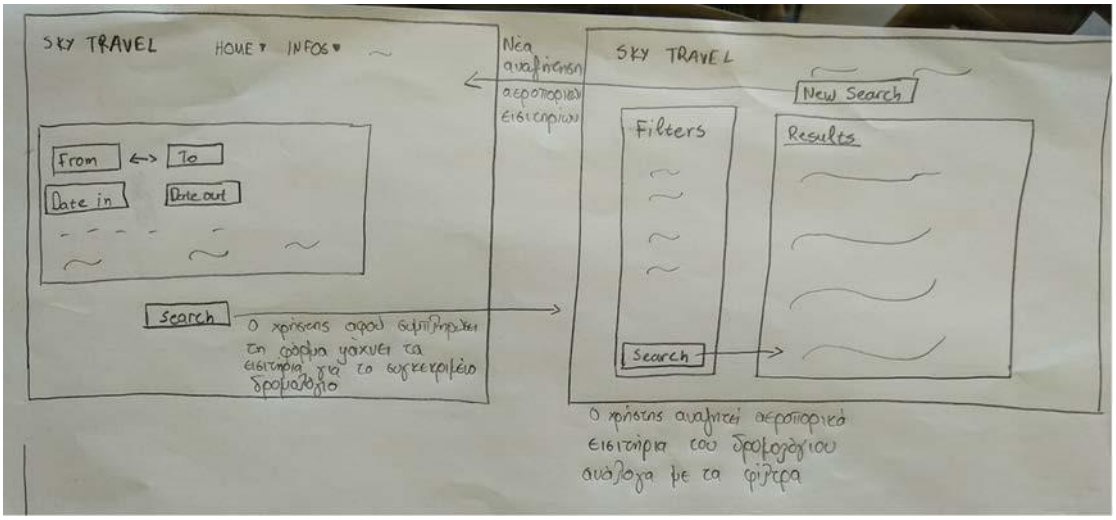
Ως περιπτώσεις χρήσης νοούνται όλες εκείνες οι ενέργειες και τα γεγονότα που είναι απαραίτητο να ακολουθηθούν και εξηγούν τη σχέση των χρηστών με το σύστημα. Η Εικόνα 4.1 παρουσιάζει την μοναδική περίπτωση χρήσης του συστήματος.



Εικόνα 4.1. *Περιπτώσεις χρήσης του συστήματος εύρεσης φθηνών αεροπορικών εισιτηρίων*

Storyboards

Ο σχεδιασμός των διεπαφών του συστήματος με το χρήστη βασίστηκε στην τεχνική των storyboards. Πρόκειται για εικονικές αναπαραστάσεις των γραφικών των σελίδων της εφαρμογής και αναπαριστούν τη λειτουργικότητα του συστήματος καθώς και τις σχέσεις ανάμεσα στις σελίδες. Πιο αναλυτικά, δείχνουν την πλοήγηση πάνω στις σελίδες της εφαρμογής, υποστηρίζουν το σχεδιαστή, ώστε να αξιολογήσει τη διεπαφή και, ώστε να προσθέσει, να τροποποιήσει ή να αφαιρέσει λειτουργικότητα εύκολα και γρήγορα. Στην Εικόνα 4.2 παρουσιάζεται το storyboard που σχεδιάστηκε για τις ανάγκες της εφαρμογής εύρεσης αεροπορικών εισιτηρίων.



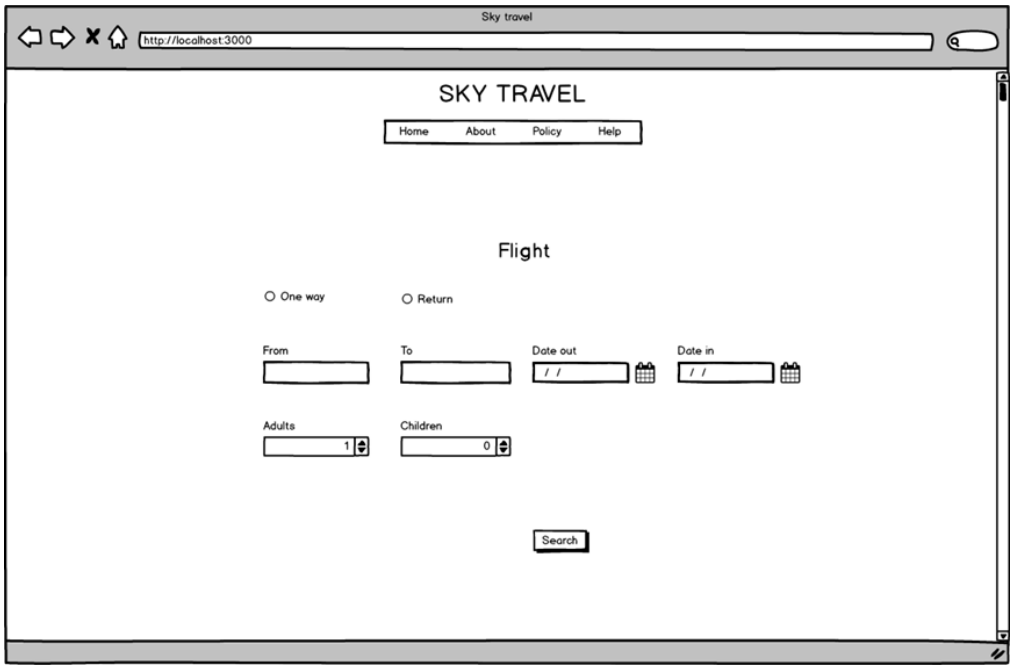
Εικόνα 4.2. *Το βασικό storyboard της εφαρμογής.Μετάβαση από την αρχική σελίδα, στη σελίδα προσθήκης φίλτρων αναζήτησης και εμφάνισης αποτελεσμάτων*

Paper prototypes

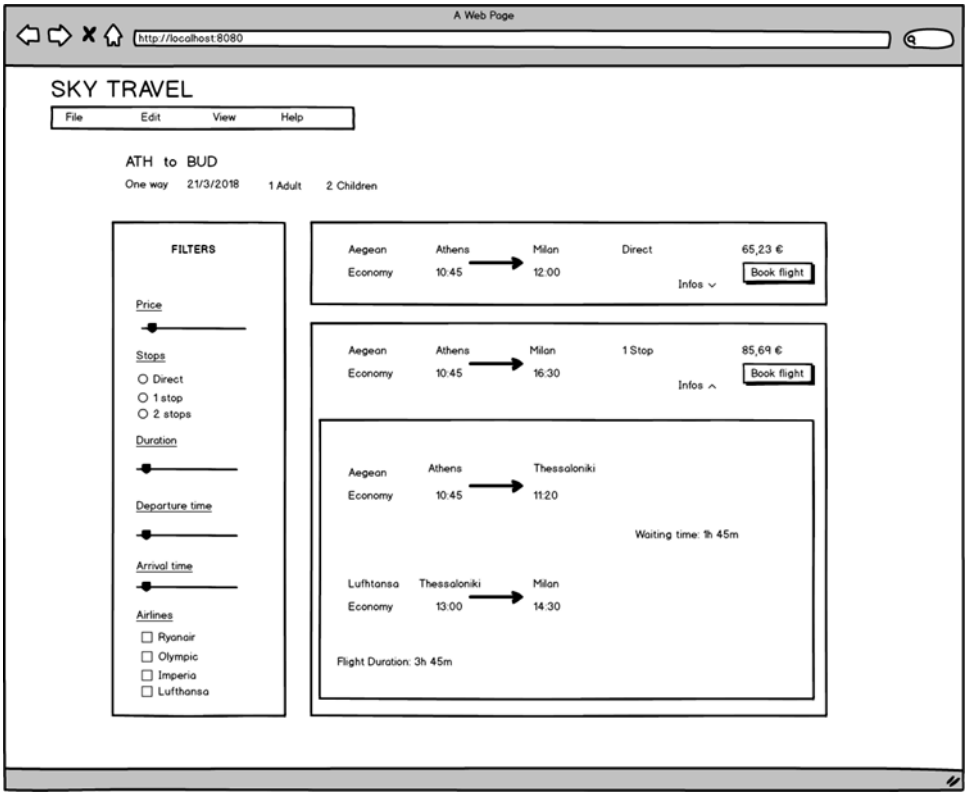
Το paper prototyping (Christodoulakis, 2012) είναι μια μέθοδος σχεδίασης διεργασίας την οποία χρησιμοποιούν οι προγραμματιστές για να σχεδιάσουν και να δοκιμάσουν διεπαφές χρήστη. Μετά την υλοποίηση των storyboards και την αξιολόγηση που πραγματοποιείται σε

αυτά, οι σχεδιαστές προχωρούν με τα paper prototypes. Ένα paper prototype αποτελεί ένα σχέδιο των διεπαφών του χρήστη της εφαρμογής με όλες τις λειτουργικότητες και τα στοιχεία που αυτές περιέχουν, με το τελικό αποτέλεσμα να είναι οι οθόνες της εφαρμογής.

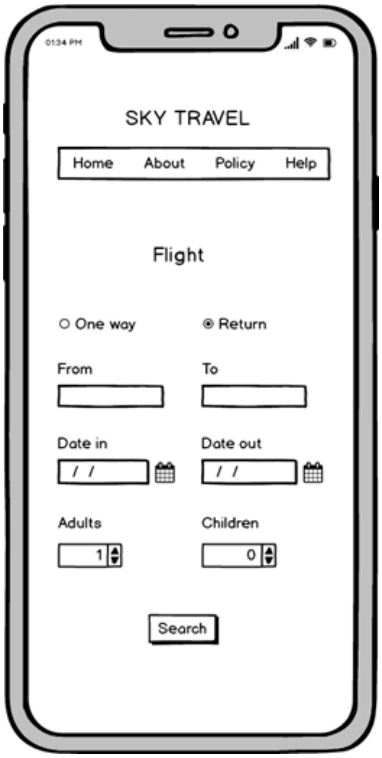
Με τη συνδρομή της συγκεκριμένης μεθόδου οι προγραμματιστές είναι σε θέση να δοκιμάσουν τις διεπαφές σε αρχικό στάδιο και έτσι να εξοικονομήσουν χρόνο ανάπτυξης. Τούτο συμβαίνει, διότι η επανασχεδίαση κάποιας διεπαφής λόγω λάθους ή εξαιτίας κάποιας καινούριας λειτουργικότητας που προκύπτει είναι εύκολη και γρήγορη. Όπως γίνεται κατανοητό, το πρώιμο στάδιο της εφαρμογής και της φύσης των paper prototypes τα οποία σχεδιάζονται συχνά ακόμη και σε χαρτί, ενθαρρύνει την κριτική των διεπαφών από τους δυνητικούς χρήστες, οι οποίοι μελετούν τις οθόνες που σχεδιάστηκαν ως πρωτότυπα. Η εν λόγω μέθοδος βοηθά στην αξιολόγηση και στον καθορισμό απαιτήσεων πριν ακόμη ξεκινήσει η ανάπτυξη του κώδικα της εφαρμογής (Sanjay Kumar Malik, 2011). Στις Εικόνες 4.3 έως 4.6 παρατίθενται τα paper prototypes τα οποία δημιουργήθηκαν πριν ξεκινήσει η ανάπτυξη της εφαρμογής εύρεσης αεροπορικών εισιτηρίων.



Εικόνα 4.3. *Εύρεση πτήσης – Desktop εφαρμογή.*



Εικόνα 4.4. *Εμφάνιση αποτελεσμάτων αεροπορικών εισιτηρίων - Desktop εφαρμογή.*

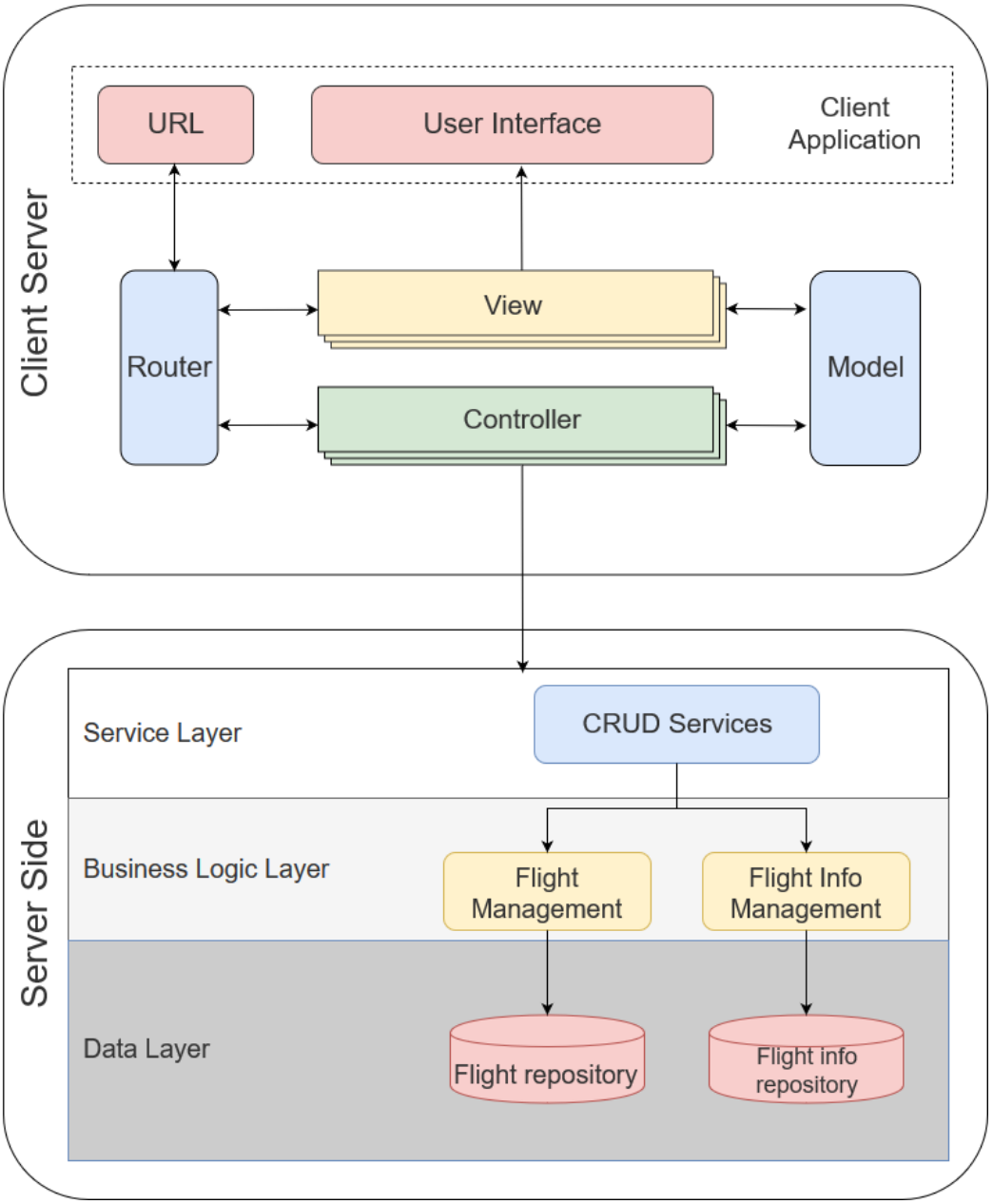


Εικόνα 4. 5. *Εύρεση πτήσης – Mobile εφαρμογή*

Εικόνα 4. 6. *Εμφάνιση αποτελεσμάτων αεροπορικών εισιτηρίων*

5 *Σχεδιασμός Εφαρμογής*

Ο σχεδιασμός της εφαρμογής βασίστηκε στην πρωταρχική ιδέα ότι τα δεδομένα που θα προκύψουν σε μία αναζήτηση του πελάτη (browser) θα προέλθουν από τις πραγματικές ιστοσελίδες, οι οποίες θα φορτωθούν σε έναν ή περισσότερους φυλλομετρητές, θα αναλυθούν βάσει του HTML περιεχομένου τους, θα αποθηκευτούν σε μία NoSQL βάση δεδομένων και θα αναλυθούν με γνώμονα τη λογική που διέπει τη σύνθεση ενός αεροπορικού ταξιδιού. Το κεντρικό σχεδιαστικό μοντέλο της εφαρμογής παρουσιάζεται στην Εικόνα 5.1.



Εικόνα 5.1. *Βασικό σχεδιαστικό μοντέλο της εφαρμογής*

Η συνολική εφαρμογή σχεδιάστηκε, ώστε να αξιοποιεί τις αρχές και τα πλεονεκτήματα του μοντέλου Model – View – Controller (MVC). Αυτό σημαίνει, ότι θα δημιουργηθούν διακριτές ενότητες οι οποίες θα ελέγχουν την παρουσίαση των δεδομένων (View), το φιλτράρισμά τους σύμφωνα με τα κριτήρια του χρήστη (Controller) και τη διαχείρισή τους σε ένα μοντέλο δεδομένων (Model).

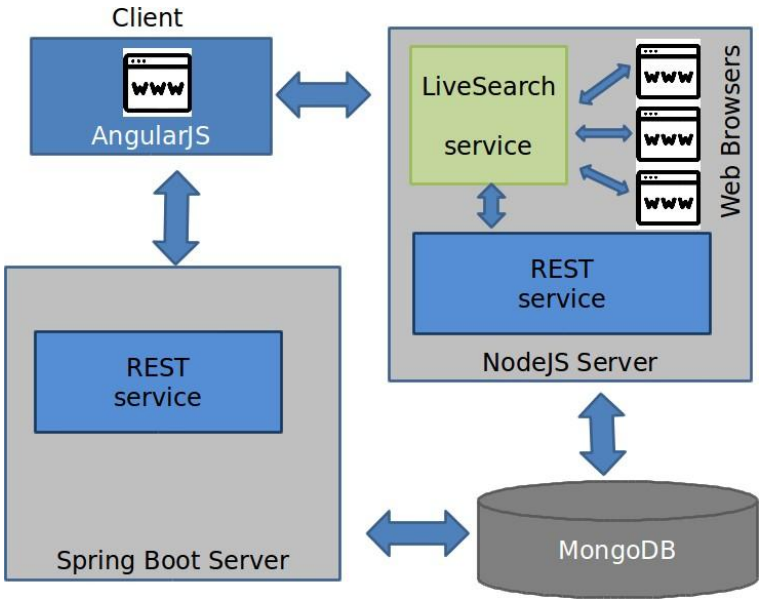
Από την Εικόνα 5.2, είναι προφανές ότι η υπηρεσία ανάκτησης των ιστοσελίδων των αποτελεσμάτων αναζήτησης εισιτηρίων για αεροπορικές διαδρομές θα πρέπει να διεκπεραιώνεται σε έναν διακομιστή, ο οποίος θα λειτουργεί ως πράκτορας, που αναζητά για λογαριασμό

του πελάτη (browser) σελίδες με αεροπορικά εισιτήρια τα οποία εξυπηρετούν την επιθυμητή διαδρομή. Προκειμένου η διαδικασία αυτή να διεκπεραιώνεται με ταχύτητα και αποδοτική εκμετάλλευση των υπολογιστικών πόρων, επιλέχθηκε η τεχνική της ασύγχρονης εκτέλεσης της διαδικασίας φόρτωσης των ιστοσελίδων σε έναν διακομιστή.

Τα δεδομένα διαβιβάζονται μέσω REST υπηρεσίας στο επίπεδο Model του συστήματος, όπου αποθηκεύονται με σκοπό τη χρήση τους στις περαιτέρω επερωτήσεις, οι οποίες θα έχουν ως αποτέλεσμα την εξαγωγή των δεδομένων που ικανοποιούν τα κριτήρια του πελάτη.

Επιπλέον, τα δεδομένα τα οποία αποθηκεύονται στο επίπεδο Model, βάσει του Controller ο οποίος καθορίζει τη λογική της αναζήτησης φιλτράρονται ώστε να εξαχθούν εν προκειμένω οι διαδρομές και τα αντίστοιχα εισιτήρια που εξυπηρετούν τα κριτήρια που έθεσε ο πελάτης (αφετηρία, προορισμό και επιθυμητοί χρόνοι αναχώρησης και άφιξης).

Σε ένα δεύτερο επίπεδο, το μοντέλο εξειδικεύεται, ώστε σε κάθε ενότητα να αντιστοιχηθούν οι τεχνολογίες οι οποίες ανταπεξέρχονται στις αντίστοιχες απαιτήσεις που αναφέρθηκαν παραπάνω (Εικόνα 5.2).



Εικόνα 5.2. *Σχεδιασμός των διακριτών επιπέδων του συστήματος*

Επίπεδο πελάτη

Στο επίπεδο πελάτη (φυλλομετρητή) δημιουργείται το περιβάλλον εισαγωγής των παραμέτρων αναζήτησης εισιτηρίων για το επιθυμητό δρομολόγιο.

Επίσης, ο φυλλομετρητής εμφανίζει τα αποτελέσματα της αναζήτησης εισιτηρίων για την επιθυμητή διαδρομή.

Επίπεδο διακομιστών

Στο επίπεδο διακομιστών το σύστημα υλοποιεί τις εξής υπηρεσίες:

1. Ανάκτηση των ιστοσελίδων με ημερομηνίες, αφετηρία και προορισμό εισιτηρίων, όπως προκύπτουν από την εξόρυξη δεδομένων σε διαδικτυακό ιστότοπο.
3. Αποθήκευση δεδομένων πτήσεων και τιμών εισιτηρίων στη βάση δεδομένων.
4. Φιλτράρισμα των δεδομένων βάσει των κριτηρίων του χρήστη.

Στη βάση της διάκρισης των υπηρεσιών στο επίπεδο του πελάτη και το επίπεδο του διακομιστή, η σχεδίαση στα δύο αυτά επίπεδα περιγράφεται όπως στις παραγράφους που ακολουθούν.

Το λογισμικό του πελάτη βασίζεται στο μοντέλο Model – View – Controller, ενώ ταυτόχρονα το μοντέλο του διακομιστή ακολουθεί το ίδιο μοντέλο. Το μοντέλο MVC του διακομιστή υλοποιείται μέσω των RESTful υπηρεσιών, οι οποίες αναλαμβάνουν τα εξής καθήκοντα:

1. Προετοιμασία και διαβίβαση επερωτήσεων προς το μοντέλο δεδομένων, ώστε να προκύψουν οι πτήσεις που ικανοποιούν τα εκάστοτε κριτήρια.
2. Αποστολή δεδομένων που προέκυψαν από τη διαδικασία εξόρυξης για αποθήκευση στο μοντέλο δεδομένων.

Η ταυτόχρονη χρήση μοντέλου MVC τόσο στον πελάτη όσο και στο διακομιστή, προσφέρει συγκεκριμένα πλεονεκτήματα τα οποία είναι κρίσιμης σημασίας σε μια εφαρμογή στην οποία προκύπτουν σημαντικοί όγκοι δεδομένων (σε περιβάλλον παραγωγικής λειτουργίας). Τέτοιου είδους πλεονεκτήματα είναι (Hyun & Soo , 2010):

Η πιο αραιή επικοινωνία και ανταλλαγή δεδομένων μεταξύ πελάτη και διακομιστή, δεδομένου ότι λειτουργίες επί των δεδομένων που έχουν διαβιβαστεί στον πελάτη μπορούν να διεκπεραιώνονται σε τοπικό επίπεδο, ελαχιστοποιώντας έτσι τις καθυστερήσεις στην απόκριση της εφαρμογής.

Η επαύξηση της ανταπόκρισης της εφαρμογής, αφού η επεξεργασία εστιάζεται σε πιο μικρό σύνολο δεδομένων σχετικών με την εφαρμογή στην πλευρά του πελάτη.

Η κατανομή του φόρτου από ένα μοναδικό διακομιστή, ο οποίος υλοποιεί τον controller του μοντέλου σε πολλαπλούς φυλλομετρητές.

Μοντέλο MVC στο επίπεδο πελάτη

Ο πελάτης της εφαρμογής έχει βασιστεί στο μοντέλο Model – View – Controller (MVC). Η υλοποίηση αυτή εξασφαλίζει ότι ο διακομιστής διαβιβάζει στην εφαρμογή δεδομένα και μόνο. Τα δεδομένα επεξεργάζονται από τον πελάτη σύμφωνα με το τυπικό μοντέλο MVC, ώστε να προκύψει τελικά η μορφή παρουσίασής τους στο φυλλομετρητή - πελάτη. Σημαντικό ρόλο στην υλοποίηση αυτή έπαιξε η ύπαρξη αρκετών προγραμματιστικών πλαισίων, τα οποία με τη χρήση της γλώσσας JavaScript το υλοποιούν με σχετικά απλό τρόπο.

Η πρόσβαση του πελάτη σε ένα προγραμματιστικό πλαίσιο MVC διευκολύνει την ανάπτυξη εφαρμογών που διαχειρίζονται δυναμικά δεδομένα. Αναλυτικότερα, λόγω του ότι η υλοποίηση MVC στον πελάτη ελαχιστοποιεί την ανάγκη ανταλλαγής μορφοποιημένων δεδομένων με το διακομιστή, προσφέρεται για την ανάπτυξη βασικών CRUD υπηρεσιών, οι οποίες διαχειρίζονται πρωτογενή, μη μορφοποιημένα δεδομένα. Οι CRUD υπηρεσίες (create, read, update, delete είναι συναρτήσεις που στόχο έχουν την αποθήκευση, ανανέωση, εύρεση και διαγραφή δεδομένων από μια βάση δεδομένων. Η θεμελιώδης αρχή στην ο- ποία βασίζεται η υλοποίηση MVC στον πελάτη, είναι η εμφύτευση ετικετών στον κώδικα HTML, οι οποίες ενεργοποιούν αντίστοιχο κώδικα γλώσσας JavaScript. Στο πλαίσιο αυτό, ο φυλλομετρητής φορτώνει την HTML σελίδα και δημιουργεί το Document Object Model για αυτήν. Στη συνέχεια, το μοντέλο DOM μεταφράζεται και συγκεντρώνονται όλες οι ετικέτες του προγραμματιστικού πλαισίου. Τέλος, μέσω των ετικετών αυτών ενεργοποιούνται οι συν- δεδεμένες συναρτήσεις της JavaScript και η σελίδα επαναφορτώνεται στην τελική της μορφή.

Server Side

Σχεδιαστικά στο διακομιστή του συστήματος εκτελούνται τρεις υπηρεσίες:

1. Η υπηρεσία ασύγχρονης ανάκτησης ιστοσελίδων οι οποίες περιέχουν πληροφορίες. Πρόκειται για την υπηρεσία υλοποίησης του μηχανισμού web scraping (LiveSearch service).
2. Η υπηρεσία διαχείρισης της επιχειρησιακής λογικής (REST service).
3. Η υπηρεσία αποθήκευσης και διαχείρισης των δεδομένων της εφαρμογής (Database Service).

LiveSearch service

Η υπηρεσία LiveSearch παραλαμβάνει το αίτημα του πελάτη για την αναζήτηση των εκάστοτε εισιτηρίων καθώς και των τιμών τους. Με δεδομένο ότι θα εφαρμοστεί τεχνική web scraping, η υπηρεσία θα δημιουργήσει το κατάλληλο URI, το οποίο και θα υποβάλλει στον ιστότοπο διάθεσης εισιτηρίων.

Προκειμένου το λογισμικό να προσφέρει διαφοροποιημένες υπηρεσίες σε σχέση με τις υπάρχουσες πλατφόρμες κρατήσεως εισιτηρίων, προσφέρει την αυτοματοποιημένη

πρόταση διαδρομών με έως δύο ενδιάμεσους σταθμούς, ώστε να επιτυγχάνεται η καλύτερη δυνατή τιμή για την συγκεκριμένη μετακίνηση.

Ως εκ τούτου , η LiveSearch service επαυξάνει το ερώτημα του πελάτη, ώστε να συμπεριλάβει και ενδιάμεσα αεροδρόμια. Αυτό επιτυγχάνεται μέσω της αναζήτησης στη βάση δεδομένων άλλων αεροδρομίων, που γεωγραφικά γειτνιάζουν με το αεροδρόμιο αφετηρίας και το αεροδρόμιο προορισμού, βάσει των γεωγραφικών τους συντεταγμένων. Συνεπώς, η αρχική επερώτηση του πελάτη παράγει επερωτήσεις οι οποίες αφορούν τα εισιτήρια με ενδιάμεση στάση. Οι επερωτήσεις αυτές θα παράξουν αντίστοιχα URI, τα οποία θα χρησιμοποιηθούν για την ανάκτηση των αντίστοιχων σελίδων και την ανάλυσή τους μέσω web scraping.

Επιπροσθέτως, λόγω της φύσης της εφαρμογής, απαιτείται η ανάκτηση όλων των σελίδων οι οποίες περιέχουν πληροφορίες για τις επιθυμητές διαδρομές (είτε απευθείας είτε με ενδιάμεση στάση). Για την αποδοτικότερη χρήση των πόρων, υιοθετείται η χρήση ασύγχρονου προγραμματισμού, ώστε η ανάκτηση των σελίδων να γίνεται ταυτόχρονα και η ολοκλήρωση της φόρτωσης της σελίδας να σηματοδοτεί την έναρξη της ανάλυσής της, της εξόρυξης των δεδομένων και της αποθήκευσής τους στη βάση δεδομένων.

Ο ασύγχρονος προγραμματισμός επιτρέπει με τη χρήση ενός και μόνο νήματος την ταυτόχρονη εκτέλεση πολλών διεργασιών για τις οποίες ο χρόνος ολοκλήρωσης δεν είναι προκαθορισμένος. Με την ολοκλήρωση μιας διεργασίας, η επόμενη διεργασία η οποία αναμένει εξυπηρέτηση εισέρχεται στην ουρά εξυπηρέτησης και η διαδικασία ολοκληρώνεται με την εισαγωγή στην ουρά εξυπηρέτησης και της τελευταίας διεργασίας (Rocha, et al., 2015). Στην περίπτωση της LiveSearch service, η φόρτωση κάθε URI αποτελεί μια διεργασία, με μη προκαθορισμένο χρόνο ολοκλήρωσης. Προγραμματίζοντας ασύγχρονα, η ολοκλήρωση μιας διεργασίας φόρτωσης μιας ιστοσελίδας προκαλεί την εκκίνηση της διεργασίας φόρτωσης της επόμενης.

Σχεδιαστικό παράγοντα αποτελεί και η παράλληλη φόρτωση των ιστοσελίδων, ώστε η υπηρεσία να εκμεταλλεύεται τους υπολογιστικούς πόρους αλλά και τη δυνατότητα των browsers να φορτώνουν πολλαπλές σελίδες ταυτόχρονα, χρησιμοποιώντας μια καρτέλα για την καθεμία (Jones, et al., 2009).

REST service

Η αρχιτεκτονική Representational State Transfer (REST) δημιουργεί μια διεπαφή μεταξύ συστημάτων, χρησιμοποιώντας το πρωτόκολλο HTTP για τη διακίνηση δεδομένων αλλά και την ενεργοποίηση λειτουργιών επί των δεδομένων αυτών με τη χρήση πρότυπων μορφότυπων, όπως XML και JSON. Τα κυριότερα χαρακτηριστικά της αρχιτεκτονικής REST είναι:

1. Πρόκειται για πρωτόκολλο πελάτη / διακομιστή, το οποίο δε διατηρεί την κατάσταση του καναλιού επικοινωνίας. Αυτό σημαίνει ότι ούτε ο πελάτης ούτε ο διακομιστής είναι

απαραίτητο να «θυμούνται» οποιαδήποτε προηγούμενη κατάσταση του καναλιού επικοινωνίας προκειμένου να επικοινωνήσουν εκ νέου.

Τα αντικείμενα στην αρχιτεκτονική REST διαχειρίζονται μέσω του URI. Οποιοσδήποτε πόρος σε ένα σύστημα REST διευθυνσιοδοτείται από το URI του. Το URI επιτρέπει την πρόσβαση στις πληροφορίες, με σκοπό την τροποποίηση ή τη διαγραφή τους.

2. Ομοιόμορφη διεπαφή. Προκειμένου να διαβιβαστούν δεδομένα μέσω REST, το σύστημα εφαρμόζει συγκεκριμένες πράξεις στους διαθέσιμους πόρους (POST, GET, PUT και DELETE) υπό την προϋπόθεση ότι αυτοί διευθυνσιοδοτούνται με ένα URI. Ο μηχανισμός αυτός δημιουργεί ομοιομορφία ως προς τη διαχείριση των πληροφοριών που διαχειρίζεται ένας διακομιστής.
3. Διαστρωματωμένη αρχιτεκτονική: προσφέρεται η δημιουργία ιεραρχικών στρωμάτων μεταξύ των ενοτήτων μιας εφαρμογής. Στο πλαίσιο αυτό η προγραμματιστική διεπαφή μπορεί να διατίθεται από το διακομιστή Α, η αποθήκευση των δεδομένων να γίνεται σε έναν διακομιστή Β και οι αιτήσεις για έλεγχο ταυτότητας να διεκπεραιώνονται από έναν τρίτο διακομιστή Γ.

Database service

Για την υλοποίηση των λειτουργιών αποθήκευσης και ανάλυσης των δεδομένων που προέρχονται από τις διαδικασίες web scraping επιλέχθηκε η τεχνολογία NoSQL. Με τον όρο NoSQL εννοούνται εκείνες οι βάσεις δεδομένων που διαθέτουν έναν μηχανισμό για αποθήκευση και ανάκτηση δεδομένων διαφορετικό από εκείνο των σχεσιακών βάσεων (SQL), ο οποίος βασίζεται στην αποθήκευση ημιδομημένων αντικειμένων. Οι τύποι δεδομένων που χρησιμοποιούν οι NoSQL βάσεις (κλειδί-τιμή, έγγραφα, κ.α.) είναι διαφορετικοί από εκείνους των σχεσιακών βάσεων γεγονός που τις καθιστά πιο αποδοτικές σε διάφορες λειτουργίες, κυρίως ανάλυσης δεδομένων (Madison, et al., 2015).

Στην περίπτωση της εφαρμογής εύρεσης εισιτηρίων για αεροπορικές διαδρομές, ο κρίσιμος παράγοντας είναι η ευελιξία στην αναζήτηση και στο συνδυασμό τιμών παρά η διεκπεραίωση των λειτουργιών ενημέρωσης με τήρηση των κανόνων ACID (Atomicity, Consistency, Isolation, Durability) που διασφαλίζουν τα σχεσιακά συστήματα διαχείρισης βάσεων δεδομένων.

5.3. Μοντέλο MVC στους διακομιστές

Οι τρεις υπηρεσίες που περιγράφηκαν παραπάνω και υλοποιούνται στην πλευρά του διακομιστή, συνεργάζονται στη βάση του μοντέλου MVC. Το μοντέλο MVC του διακομιστή υλοποιεί τρεις δομικές ενότητες.

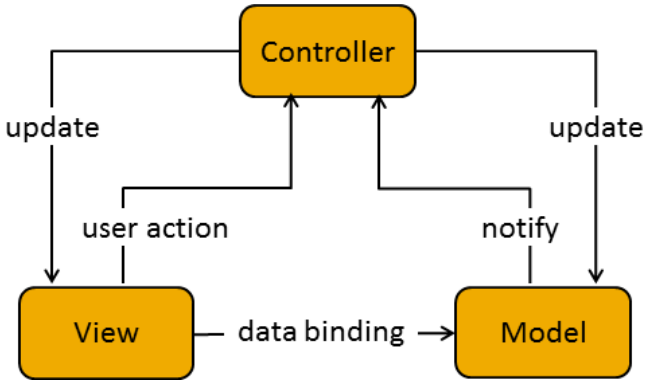
1. Η ενότητα View (Εμφάνιση) εξυπηρετεί την εμφάνιση των σελίδων οι οποίες περιέχουν τα στοιχεία για τις πτήσεις και τα εισιτήρια τα οποία ενδιαφέρουν το εκάστοτε ερώτημα του πελάτη.

2. Η ενότητα Controller (Ελεγκτής) συντονίζει τη ροή των δεδομένων από και προς την ενότητα Model και την ενότητα View. Ο Ελεγκτής αντιδρά στις ενέργειες του χρήστη και ανακαλεί πληροφορίες από το Μοντέλο, ώστε να τις διοχετεύσει στην ενότητα View.

   Ο Ελεγκτής αποτελείται από τις υπηρεσίες REST, οι οποίες καθορίζουν τους κανόνες αναζήτησης πτήσεων και εισιτηρίων προς τον διαδικτυακό ιστότοπο που εξορίζονται τα δεδομένα και δέχονται τα αποτελέσματα από τις διεργασίες web scraping.

3. Η ενότητα Model (Μοντέλο) αναπαριστά τα δεδομένα που αποθηκεύονται στη βάση δεδομένων. Η υλοποίηση αυτής της ενότητας αντιστοιχίζει τα δεδομένα της βάσης σε προγραμματιστικά αντικείμενα (objects). Τα προς αποθήκευση αντικείμενα διαβιβάζονται στην Database service του συστήματος για εξυπηρέτηση.

Συνοπτικά το μοντέλο MVC απεικονίζεται όπως στην Εικόνα 5.3.



Εικόνα* 5.*3. *Το μοντέλο Model – View – Controller*
Πηγή: SAP Help Portal, https://help.sap.com/erp_hcm_ias2_2014_03/helpdata/en/91/f233476f4d1014b6dd926db0e91070/content.htm?no_cache=true

6 *Υλοποίηση Εφαρμογής*

Client Side

Για την υλοποίηση των λειτουργιών της εφαρμογής στην πλευρά του πελάτη χρησιμοποιήθηκαν αποκλειστικά τεχνολογίες οι οποίες μπορούν να εκτελεστούν σε έναν τυπικό φυλλομετρητή, χωρίς να είναι απαραίτητη η εγκατάσταση κάποιου πρόσθετου λογισμικού (όπως κάποιο runtime). Με δεδομένο ότι ένας πελάτης της εφαρμογής, ο οποίος θα καταναλώσει τις υπηρεσίες του διακομιστή είναι ένας φυλλομετρητής και μόνο, η σχεδιαστική απόφαση για την υλοποίησή του οδήγησε στην υιοθέτηση των εξής τεχνολογιών και προγραμματιστικών εργαλείων:

- HTML5
- CSS3
- AngularJS

Η επιλογή του προγραμματιστικού πλαισίου AngularJS έγινε με γνώμονα την απλουστευμένη υλοποίηση του μοντέλου MVC στην πλευρά του πελάτη, μέσω της εμφύτευσης στον κώδικα HTML πρόσθετων ετικετών οι οποίες και πυροδοτούν την εκτέλεση λειτουργιών μέσω του πλαισίου. Το πλεονέκτημα της AngularJS έγκειται στο ότι η HTML παραμένει η γλώσσα περιγραφής της σελίδας, ενώ ταυτόχρονα συγχρονίζει τα δεδομένα από τη διεπαφή του χρήστη (User Interface) με τα JavaScript αντικείμενα που έχουν δημιουργηθεί από τη σελίδα, χρησιμοποιώντας τεχνική 2-way binding.

Η προσέγγιση του AngularJS είναι η υλοποίηση «διαχειριστών γεγονότων» (event handlers) και η ενεργοποίησή τους, όταν το προγραμματιστικό πλαίσιο λάβει την απόφαση ότι πρέπει να ενεργοποιήσει κάποιον απ' αυτούς.

Με τη χρήση του προγραμματιστικού πλαισίου AngularJS επιτυγχάνεται η πλήρης διάκριση μεταξύ του προτύπου μιας ιστοσελίδας (DOM), τα μοντέλα και τη λειτουργικότητα (όπως αυτή έχει αναπτυχθεί στους controllers).

MVC Pattern

Το σχεδιαστικό μοντέλο στην πλευρά του πελάτη είναι το Model-View-Controller, το οποίο υλοποιήθηκε με το AngularJS. Στη βάση του συγκεκριμένου μοντέλου

δημιουργήθηκαν δομικές μονάδες (components), οι οποίες μπορούν να επαναχρησιμοποιηθούν στην ίδια ή διαφορετικές σελίδες. Οι μονάδες αυτές αντλούν δεδομένα από το μοντέλο δεδομένων και τροφοδοτούν το επίπεδο view.

Η χρήση του AngularJS ως προγραμματιστικού πλαισίου στην πλευρά του πελάτη, υλοποιεί το μοντέλο MVC ως εξής:

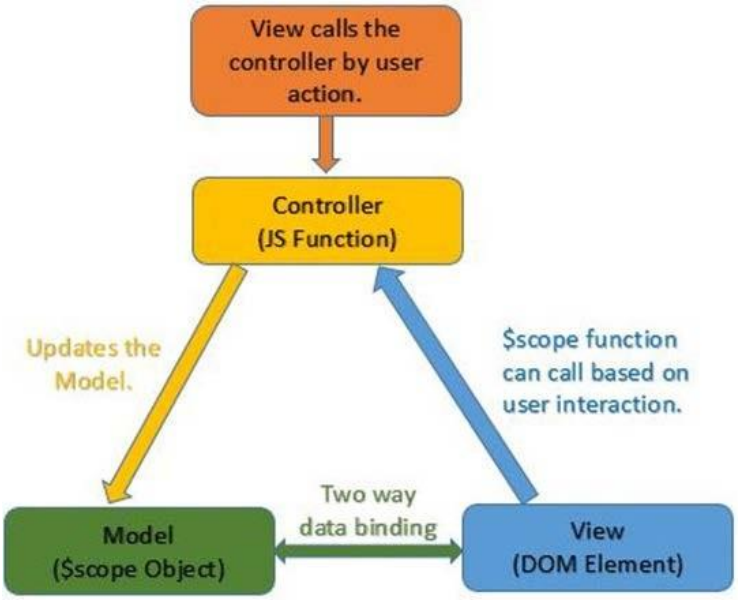
View Υλοποιείται με τη γλώσσα περιγραφής HTML και τις πρόσθετες ετικέτες του AngularJS, ώστε να είναι δυνατός ο έλεγχος του Document Object Model που δημιουργεί η εκάστοτε σελίδα που φορτώνεται στο φυλλομετρητή.

Controller Ο Controller αποστέλλει αιτήματα προς το REST Service και αναθέτει τις πληροφορίες οι οποίες είναι απαραίτητες για το επίπεδο View στο αντικείμενο $scope του προγραμματιστικού πλαισίου.

Υλοποιεί επίσης τις callback συναρτήσεις, οι οποίες ενεργοποιούνται ως απόκριση σε συγκεκριμένα γεγονότα.

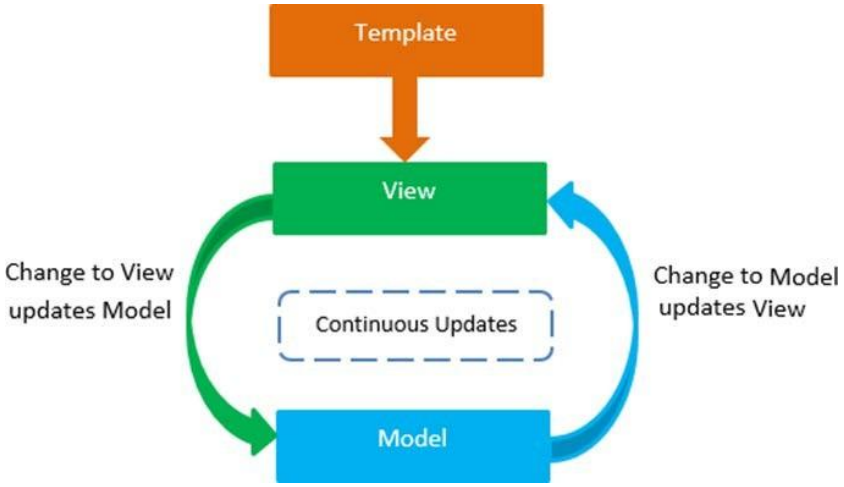
Επιφορτίζεται με λειτουργίες επικύρωσης δεδομένων, ώστε αυτή να διεκπεραιώνεται από τον πελάτη και να μην επιβαρύνεται ο διακομιστής με τέτοιου είδους καθήκοντα.

Η Εικόνα 6.1 και 6.2 παρουσιάζει τα στοιχεία του MVC μοντέλου και τη σύνδεση μεταξύ αυτών.



Εικόνα 6.1. *AngularJS MVC Framework Workflow 2-way data binding*

Πηγή : http://javaelegance.blogspot.com/2015/12/angularjs-mvc-framework-tutorial.html



Εικόνα 6.2. *2-Way Data Binding*

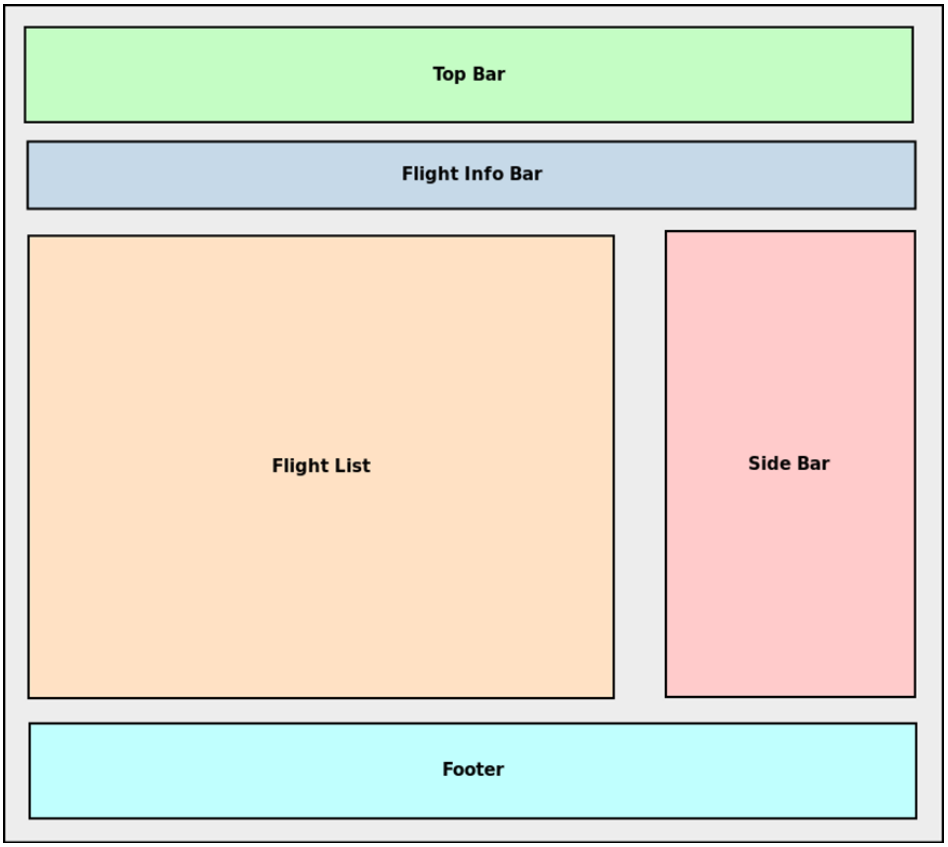
Πηγή: https://codesjava.com/angularjs-two-way-data-binding-program-code

Components

Βάσει της αρχιτεκτονικής του AngularJS, ο πελάτης απαρτίζεται από διαφορετικές δομικές μονάδες (components). Κάθε δομική μονάδα υλοποιείται σύμφωνα με το μοντέλο MVC και διαθέτει δικό της Controller, View και Service. Το AngularJS διευκολύνει την επαναχρησιμοποίηση μιας δομικής μονάδας με τη χρήση της ετικέτας:

```
<component_name>.
```

Η Εικόνα 6.3 περιγράφει τη δόμηση μιας σελίδας του πελάτη με χρήση δομικών μονάδων.



Εικόνα 6.3. *Δόμηση σελίδας του πελάτη με χρήση δομικών μονάδων (components) του AngularJS.*

Server Side

Ο διακομιστής του συστήματος υλοποιεί τρεις υπηρεσίες, όπως αυτές έχουν περιγραφεί στο σχεδιασμό της εφαρμογής. Οι υπηρεσίες αυτές αναπτύχθηκαν με αντίστοιχες τεχνολογίες, ως εξής:

1. LiveSearch service

   Βασίστηκε στο NodeJS και το πλαίσιο Puppeteer για τον έλεγχο των φυλλομετρητών οι οποίοι φορτώνουν τις προς εξόρυξη σελίδες, τον ασύγχρονο προγραμματισμό της φόρτωσης των σελίδων και την εξόρυξη των δεδομένων που αφορούν στην εφαρμογή.

2. REST service

   Οι υπηρεσίες REST υλοποιούνται στο πλαίσιο Spring Boot. Οι υπηρεσίες REST δέχονται αιτήματα εξυπηρέτησης και από το NodeJS και από τον πελάτη, κάθε φορά που αυτός τροποποιεί τα φίλτρα αναζήτησης και ολοκληρώνεται η εξόρυξη των αντίστοιχων δεδομένων.

3. Database service

   Υλοποιείται στη MongoDB, ώστε να απλουστεύεται η αποθήκευση των δεδομένων που προκύπτουν από τη διαδικασία εξόρυξης αλλά και η επερώτησή τους για τον εντοπισμό των διαδρομών που ενδιαφέρουν τον πελάτη.

LiveSearch service

Η υπηρεσία LiveSearch αρχικά ψάχνει στη βάση δεδομένων, ώστε να εντοπίζει τα ενδιάμεσα αεροδρόμια μεταξύ του αρχικού και του τελικού προορισμού. Η αναζήτηση γίνεται με βάση δύο κριτήρια τα οποία συμπλήρωσε ο χρήστης στη φόρμα αναζήτησης, αυτά είναι: το μέγεθος των ενδιάμεσων αεροδρομίων στα οποία επιθυμεί ο χρήστης να ψάξει το σύστημα για πιθανόν στάσεις και το μήκος της ακτίνας στην οποία επιθυμεί να βρίσκονται τα συγκεκριμένα αεροδρόμια. Φυσικά, μεγαλύτερη ακτίνα σημαίνει περισσότερα αεροδρόμια και πιθανώς μεγαλύτερο ταξίδι. Η αναζήτηση στη βάση δεδομένων έγινε με τη βοήθεια του γεωχωρικού δείκτη (2dsphere index) που παρέχει η MongoDB. Πιο συγκεκριμένα, υπολογίζεται η μέση απόσταση των αεροδρομίων αναχώρησης και προορισμού και στη συνέχεια με τη βοήθεια του δείκτη υπολογίζεται η ακτίνα αναζήτησης ενδιάμεσων σταθμών λαμβάνοντας ως παράμετρο και το μέγεθος της επιθυμητής ακτίνας που δίνει ο χρήστης. Στη συνέχεια, με τη χρήση του δείκτη βρίσκουμε όλες τις συντεταγμένες αεροδρομίων που βρίσκονται σε αυτή την ακτίνα. Ο αλγόριθμος αναζήτησης παρατίθεται στην Εικόνα 6.4.

```
db.<collection>.find( { <location field> :
                        { $geoWithin :
                          { $centerSphere :
                             [ [ <x>, <y> ] , <radius> ] }
                      } } )
```

Εικόνα 6.4 *Ερώτηση αναζήτησης τοποθεσίας με βάση την ακτίνα της οποίας το κέντρο δλινεται από το γεωγραφικό πλάτος και μήκος (<x> και <y>)*

Επιπλέον, το προγραμματιστικό πλαισίο Puppeteer ενεργοποιεί ασύγχρονα έναν φυλλομετρητή στο διακομιστή στον οποίο φορτώνονται οι σελίδες του ιστότοπου που θα διεξαχθεί το web scraping. Το Puppeteer ασύγχρονα φορτώνει τις HTML σελίδες σε διαδοχικές καρτέλες. Αν μια διαδρομή μπορεί να εξυπηρετηθεί από ενδιάμεσα αεροδρόμια, τότε για κάθε ενδιάμεσο αεροδρόμιο δημιουργεί μια καρτέλα στο φυλλομετρητή, ώστε να προκύψουν τα αντίστοιχα δεδομένα.

Εν συνεχεία, για κάθε ένα από τα ενδιάμεσα αεροδρόμια (ανοίγοντας έναν καινούριο φυλλομετρητή), το Puppeteer δημιουργεί μία καρτέλα ανά διαδρομή (από το ενδιάμεσο αεροδρόμιο προς τον τελικό προορισμό).

Η παραπάνω υλοποίηση παρουσιάζεται με τη μορφή ψευδοκώδικα στην Εικόνα 6.5.

Η Εικόνα 6.6 παρουσιάζει ένα παράδειγμα δημιουργίας φυλλομετρητή και αναζήτησης στην ιστοσελίδα με τις απαραίτητες παραμέτρους.

```
Ασύγχρονη εκτέλεση
{
    Ανοιγμα φυλλομετρητή
        Φόρτωση της ιστοσελίδας με την απευθείας πτήση στον ιστότοπο που κάνουμε εξόρυξη
        δεδομένων
            Εξόρυξη δεδομένων από την ιστοσελίδα και αποθήκευσή τους στη βάση δεδομένων
        Κλείσιμο σελίδας
    Κλείσιμο φυλλομετρητή

    Για κάθε ένα ενδιάμεσο αεροδρόμιο που βρέθηκε
        Ανοιγμα φυλλομετρητή
            Ασύγχρονη φόρτωση των ιστοσελίδων από τον αρχικό προορισμό στα ενδιάμεσα
            αεροδρόμια στον ιστότοπο που κάνουμε εξόρυξη δεδομένων
                Εξόρυξη δεδομένων από την ιστοσελίδα και αποθήκευσή τους στη βάση
                δεδομένων
            Κλείσιμο σελίδων
        Κλείσιμο φυλλομετρητή
}
Ασύγχρονη εκτέλεση
{
    Αναζήτηση στη βάση δεδομένων όλων των ενδιάμεσων αεροδρομίων στα οποία βρέθηκαν
    πτήσεις σε αυτά από τον αρχικό προορισμό.

    Για κάθε ένα ενδιάμεσο αεροδρόμιο που βρέθηκαν πτήσεις
        Ανοιγμα φυλλομετρητή
            Ασύγχρονη φόρτωση των ιστοσελίδων από τον ενδιάμεσο προορισμό στον τελικό
            προορισμό στον ιστότοπο που κάνουμε εξόρυξη δεδομένων
                Εξόρυξη δεδομένων από την ιστοσελίδα και αποθήκευσή τους στη βάση
                δεδομένων
            Κλείσιμο σελίδων
Κλείσιμο φυλλομετρητή
}
```

Εικόνα 6.5 *Ψευδοκώδικας που αναλύει την ασύγχρονη φόρτωση των απαραίτητων ιστοσελίδων για εξόρυξη δεδομένων.*

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto(`http://flights.lonelyplanet.com/en-GB/flights#/result?
    originplace=${from}&destinationplace=${to}&outbounddate=${check_in}
    &inbounddate=&adults=${adult_num}&children=${child_num}
    &infants=0&cabinclass=Economy`)
);
  const divElement = await page.evaluate(() =>
document.querySelector('div').textContent.trim();

  browser.close();
})();
```

Εικόνα 6.6 *Δημιουργία φυλλομετρητή και αναζήτηση σε ιστοσελίδα με τις απαραίτητες παραμέτρους.*

REST service

Οι υπηρεσίες REST υλοποιούν δύο λειτουργικές απαιτήσεις:

1. Την αποθήκευση των δεδομένων που παράγονται από το LiveSearch service (NodeJS & Puppeteer). Πρόκειται για τα δεδομένα που προέρχονται από web scraping και αφορούν το σύνολο των εγγραφών που βρέθηκαν στις σελίδες τις οποίες επισκέφθηκε το LiveSearch Service.
2. Την ανάκτηση δεδομένων από τη βάση δεδομένων, με γνώμονα τις επερωτήσεις που υπέβαλλε ο πελάτης. Οι επερωτήσεις εξυπηρετούνται από το Database Service, όπου διαβιβάζονται μέσω της αντίστοιχης REST υπηρεσίας.

Database service

Η database service του συστήματος υλοποιήθηκε σε σύστημα βάσης δεδομένων της οικογένειας NoSQL. Επιλέχθηκε η βάση δεδομένων MongoDB, ανοικτού κώδικα. Η αποθηκευτική μονάδα της MongoDB είναι το Έγγραφο. Μια εγγραφή στη MongoDB ισούται με ένα Έγγραφο (Document). Έγγραφο είναι μια δομή δεδομένων αποτελούμενη από ζεύγη πεδίων και των αντίστοιχων τιμών. Είναι παρόμοια με ένα JSON αντικείμενο, ενώ οι τιμές των πεδίων τους μπορούν να περιέχουν: όλους τους υποστηριζόμενους JSON τύπους, άλλα Έγγραφα, πίνακες ή πίνακες Εγγράφων καθώς και κάποιους επιπλέον τύπους όπως Date, Timestamps, ObjectId.

Τα πλεονεκτήματα ενός εγγράφου είναι τα παρακάτω:

- Τα Έγγραφα μπορούν να αναπαρασταθούν εύκολα σε κάθε γλώσσα προγραμματισμού με τους τύπους δεδομένων που διαθέτει η κάθε μία.
- Ενσωματωμένα Έγγραφα και πίνακες μειώνουν την ανάγκη για περίπλοκες συνενώσεις (expensive joins).
- Τα Έγγραφα δεν έχουν στατική μορφή και μπορούν να αλλάξουν οποιαδήποτε στιγμή δυναμικά, υποστηρίζοντας έτσι ευέλικτα τον πολυμορφισμό και δίνοντας ευελιξία στο προγραμματιστή όσον αφορά τη δημιουργία της δομής της βάσης δεδομένων.

Δομή της βάσης δεδομένων

Η βάση δεδομένων MongoDB χρησιμοποιείται για την αποθήκευση τριών συλλογών αντικειμένων:

1. Όλα τα αεροδρόμια του κόσμου. Για κάθε αεροδρόμιο αποθηκεύονται και τα γεωχωρικά του χαρακτηριστικά, ώστε η αναζήτηση των ενδιάμεσων σταθμών να γίνεται βάσει της γειτνίασης των αεροδρομίων.

4. Οι πρωτογενείς πληροφορίες πτήσεων, οι οποίες προέκυψαν από την εξόρυξη δεδομένων από τη διαδικασία web scraping. Πρόκειται για τις πληροφορίες αφετηρίας και άφιξης, τις αντίστοιχες ημερομηνίες και τις τιμές των εισιτηρίων.

5. Οι πληροφορίες πτήσης όπως προέκυψαν από τη φόρμα συμπλήρωσης.

Η ευελιξία της MongoDB επιτρέπει τη δημιουργία πολλαπλών εγγραφών στο ίδιο έγγραφο, καθεμία από τις οποίες αντιστοιχεί σε ένα τμήμα της πτήσης, σε περίπτωση που αυτή απαρτίζεται από περισσότερα του ενός σκέλη. Η μόνη διαφοροποίηση είναι ότι το έγγραφο που αντιστοιχεί σε μία πτήση με δύο σκέλη (έναν ενδιάμεσο σταθμό) περιλαμβάνει δύο εγγραφές, καθεμιά από τις οποίες αντιστοιχεί σε ένα σκέλος. Η δυνατότητα αυτή είναι πολύ σημαντική υπό το πρίσμα της επέκτασης του συστήματος για την εύρεση πτήσεων με πολλαπλές ενδιάμεσες στάσεις.

Ακόμη, με τη χρήση του 2dsphere geospatial index της MongoDB, διεκπεραιώνεται η αναζήτηση των αεροδρομίων τα οποία γειτνιάζουν με τα αεροδρόμια αφετηρίας και προορισμού, ώστε η αναζήτηση να επεκταθεί και να αναζητούνται εισιτήρια τόσο για την απευθείας διαδρομή όσο και για τις επιμέρους διαδρομές, οι οποίες και αποτελούν τα δύο σκέλη της συνολικής.

Τεχνολογίες

NodeJS

Το NodeJS είναι μια πλατφόρμα ανάπτυξης λογισμικού (κυρίως διακομιστών) χτισμένη σε περιβάλλον JavaScript. Στόχος της είναι να παρέχει ένα εύκολο τρόπο δημιουργίας κλιμακωτών διαδικτυακών εφαρμογών. Σε αντίθεση με τα περισσότερα σύγχρονα περιβάλλοντα ανάπτυξης εφαρμογών δικτύων μία διεργασία node δεν στηρίζεται στην πολυνηματικότητα (multithreading) αλλά σε ένα μοντέλο ασύγχρονης επικοινωνίας εισόδου/εξόδου (Tilkov & Vinoski, 2010). Αυτό του είδους το μοντέλο λειτουργίας στοχεύει στη βελτίωση της διεκπεραιωτικής ικανότητας των διαδικτυακών εφαρμογών με πολλές λειτουργίες εισόδου/εξόδου, όπως και εφαρμογών ιστού πραγματικού χρόνου (προγράμματα επικοινωνίας πραγματικού χρόνου, παιχνίδια φυλλομετρητών) (Dmello, 2016).

Η αρχιτεκτονική της πλατφόρμας φέρνει τον προγραμματισμό οδηγούμενο από γεγονότα (event – driven programming) στους εξυπηρετητές, επιτρέποντας την ανάπτυξη γρήγορων διακομιστών σε JavaScript (Teixeira, 2012). Ο οδηγούμενος από γεγονότα προγραμματισμός είναι ένα μοντέλο στο οποίο η ροή του προγράμματος καθορίζεται από γεγονότα όπως δράσεις του χρήστη (κλικ ποντικιού, πάτημα κουμπιού), εξόδους αισθητήρων ή μηνύματα από άλλα προγράμματα/νήματα. Είναι το κυρίαρχο μοντέλο προγραμματισμού που χρησιμοποιείται στις γραφικές διεπαφές χρηστών και σε εφαρμογές επικεντρωμένες στην εκτέλεση συγκεκριμένων ενεργειών ως απόκριση σε είσοδο του χρήστη. Σε οδηγούμενες από γεγονότα εφαρμογές υπάρχει συνήθως ένας βασικός βρόχος που αναμένει την εμφάνιση γεγονότων και στη συνέχεια πυροδοτεί μια συνάρτηση επανάκλησης (callback function), όταν εμφανιστεί κάποιο γεγονός (Schaumont, 2013).

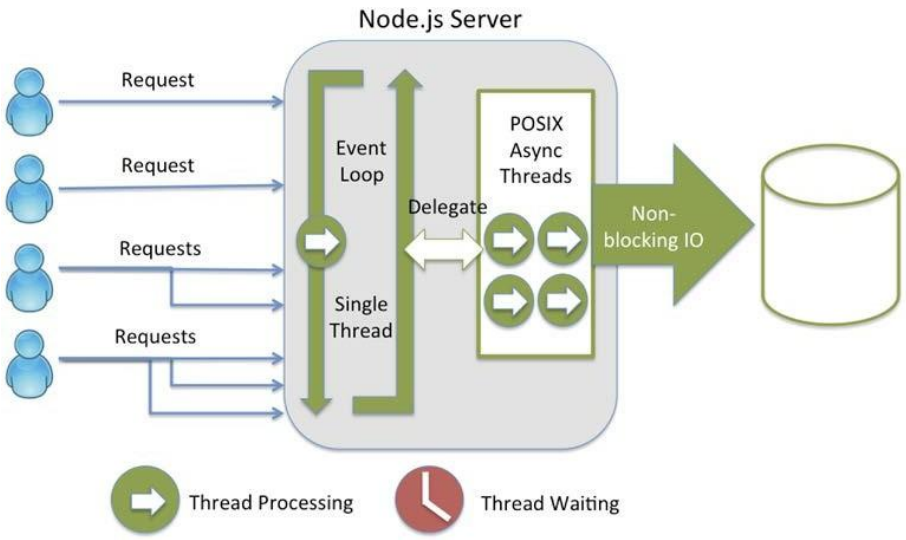
Με τη βοήθεια του μοντέλου που περιγράφεται παραπάνω, ο προγραμματιστής μπορεί να δημιουργήσει μεγάλης έκτασης εξυπηρετητές χωρίς τη χρήση της πολυνηματικότητας, αλλά με την εκμετάλλευση ενός απλοποιημένου μοντέλου. Το απλοποιημένο μοντέλο του οδηγούμενο από γεγονότα προγραμματισμού χρησιμοποιεί επανακλήσεις για να σηματοδοτήσει την ολοκλήρωση μίας διεργασίας (Teixeira, 2012). Η πλατφόρμα της Node δημιουργήθηκε, επειδή η παραλληλία είναι δύσκολα υλοποιήσιμη σε πολλές γλώσσες πραγματισμού εξυπηρετητών και συχνά οδηγεί σε μειωμένη απόδοση. Η ανάπτυξη της Node είναι βασισμένη στην ανοιχτού κώδικα μηχανή V8 JavaScript της Google, διαθέτει εξαιρετική ταχύτητα και παρουσιάζει ευχέρεια στα βασικά διαδικτυακά πρωτόκολλα HTTP, DNS, TCP (Ornbo, 2012). Τέλος η βάση της πλατφόρμας, η γλώσσα JavaScript, είναι τόσο διαδεδομένη, ώστε την κάνει άμεσα προσβάσιμη στο σύνολο της κοινότητας των προγραμματιστών διαδικτύου.

Πλεονεκτήματα της NodeJS είναι:

- Η αυξημένη ταχύτητα. Όπως προαναφέρθηκε, η Node είναι μία πλατφόρμα ανάπτυξης λογισμικού που χρησιμοποιεί την μηχανή V8, η οποία κατασκευάστηκε από την Google για την ενσωμάτωση στον περιηγητή της, τον Chrome. Η συγκεκριμένη μηχανή μεταγλωττίζει και εκτελεί κώδικα JavaScript σε εξαιρετικές ταχύτητες κυρίως λόγω του γεγονότος ότι ο μεταγλωττιστής της μετατρέπει απευθείας τη JavaScript σε κώδικα μηχανής.
- Η προσφορά του βρόχου γεγονότων. Ο βρόχος γεγονότων είναι ένα μονό νήμα που εκτελεί ασύγχρονα όλες τις λειτουργίες εισόδου/εξόδου. Παραδοσιακά οι λειτουργίες εισόδου/εξόδου εκτελούνται σύγχρονα, μπλοκάροντας η μία την άλλη, ή ασύγχρονα αξιοποιώντας παράλληλα νήματα. Αυτή η προσέγγιση τείνει να ξεπεραστεί λόγω της αυξημένης μνήμης που απαιτεί και της φήμης της στη δυσκολία προγραμματισμού. Αντίθετα, όταν μια Node εφαρμογή απαιτεί την εκτέλεση μιας λειτουργίας εισόδου/εξόδου αποστέλλει μια ασύγχρονη εργασία στο βρόχο γεγονότων, μαζί με μία συνάρτηση επανάκλησης και συνεχίζει την κανονική ροή του προγράμματος. Τέλος, όταν ολοκληρωθεί η ασύγχρονη διεργασία, ο βρόχος γεγονότων επιστρέφει στη διαδικασία και εκτελεί την ανάκλησή της.
- Το γεγονός ότι βασίστηκε σε μία ήδη διαδεδομένη γλώσσα, τη JavaScript. Τα πιο διαδεδομένα πλαίσια ανάπτυξης εφαρμογών στην πλευρά του πελάτη (client – side) στηρίζονται κατά κόρον, αν όχι αποκλειστικά, στη λογική της JavaScript. Με τη Node δε χρειάζεται πλέον η μετάφραση της λογικής της πλευράς πελάτη σε αυτή της πλευράς του εξυπηρετητή, αφού είναι κοινή. Επίσης δεν απαιτείται η μετάφραση των HTTP δεδομένων που στέλνονται σε διαφορετικά αντικείμενα στην πλευρά του εξυπηρετητή.

Ασύγχρονη εκτέλεση

Το NodeJS αποτελεί ένα πολύ ισχυρό εργαλείο λόγω του ασύγχρονου προγραμματισμού που υποστηρίζει. Σύμφωνα με την ασύγχρονη λογική μια διεργασία δεν είναι απαραίτητο να περιμένει για μια ενέργεια που βρίσκεται σε έναν βρόγχο γεγονότων να ολοκληρωθεί έτσι ώστε να αρχίσει να εκτελείται η επόμενη διεργασία του βρόγχου. Αυτό επιτυγχάνεται με το να δηλώσουμε το αποτέλεσμα μιας διεργασίας ως callback, και όταν αυτή ολοκληρωθεί, η συνάρτηση callback επιστρέφει το αποτέλεσμα στο περιβάλλον της εφαρμογής για περαιτέρω επεξεργασία. Η λειτουργικότητα αυτή καθιστά το NodeJS ένα αποτελεσματικό εργαλείο για server-side προγραμματισμό χαρακτηριστικό του οποίου είναι ότι υποστηρίζει εγγενώς την αποτελεσματική χρήση των διαθέσιμων πόρων. Στην Εικόνα 6.7 παρουσιάζεται η αρχιτεκτονική του NodeJS.

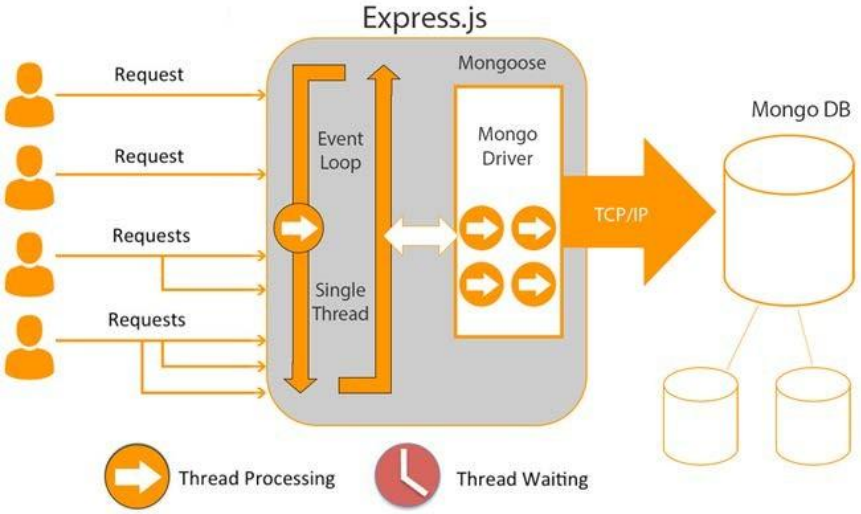


Εικόνα 6.7. *Αρχιτεκτονική NodeJS*
Πηγή: *https://codeburst.io/all-about-node-js-you-wanted-to-know-25f3374e0be7*

Express

Το Express (Hahn, 2016) (Mithun Atheesh, 2015) είναι ένα προγραμματιστικό πλαίσιο (framework) για το NodeJS, το οποίο βοηθά στην οργάνωση της λειτουργικότητας της εφαρμογής. Το πλαίσιο αυτό προσθέτει πολλά χαρακτηριστικά στο NodeJS, τα οποία απλοποιούν σε μεγάλο βαθμό την ανάπτυξη δυναμικών HTML σελίδων. Χρησιμοποιεί το MVC μοντέλο για την ανάπτυξη μιας εφαρμογής και παρέχει προγραμματιστικά εργαλεία, ώστε να επιτυγχάνεται εύκολη επικοινωνία με ένα διακομιστή. Πιο συγκεκριμένα, αυτό που στην πράξη προσφέρει το Express είναι η ευελιξία στην επικοινωνία με τον διακομιστή HTTP, χρησιμοποιώντας περιορισμένο αριθμό γραμμών κώδικα, ενώ παράλληλα προσφέρει στην εφαρμογή αυξημένη λειτουργικότητα. Χαρακτηριστικό παράδειγμα λοιπόν είναι η μεταφόρτωση ανέβασμα μιας φωτογραφίας στο διακομιστή. Για τη διεκπεραίωση του έργου με χρήση NodeJS API και μόνο απαιτούνται περίπου 40 γραμμές κώδικα ενώ η αντίστοιχη λειτουργία με τη χρήση του πλαισίου Express διεκπεραιώνεται με μόνο μία γραμμή κώδικα.

Επιπρόσθετα, ένα άλλο χαρακτηριστικό που διαθέτει το πλαίσιο Express και το οποίο βοηθά στη συγγραφή βαθμωτού κώδικα που μπορεί να παραμετροποιηθεί και να συντηρηθεί εύκολα, είναι η χρήση μικρών συναρτήσεων για τη διαχείριση των αιτημάτων. Το NodeJS API χρησιμοποιεί μια JavaScript συνάρτηση που ονομάζεται «διαχειριστής αιτημάτων», η οποία δέχεται ως ορίσματα το αίτημα και την απάντηση. Ο διαχειριστής αιτημάτων διαβιβάζει τα αιτήματα στην εφαρμογή. Με τη χρήση μικρών και αρκετών τέτοιων διαχειριστών δημιουργείται απλοποιημένος και πιο συμπαγής κώδικας (Teixeira, 2012). Η Εικόνα 6.8 παρουσιάζει την αρχιτεκτονική του προγραμματιστικού πλαισίου Express.



Εικόνα 6.8. *Αρχιτεκτονική του προγραμματιστικού πλαισίου Express*
Πηγή: https://apiko.com/blog/express-mobile-app-development/

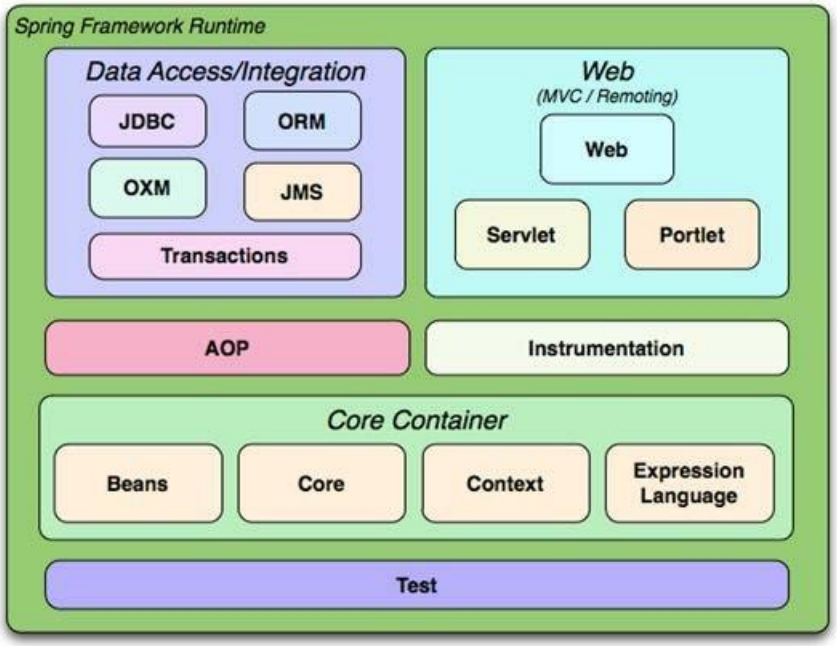
Spring Framework

Το Spring Framework είναι περιβάλλον εργασίας ανοιχτού κώδικα (open source) βασισμένο στον κώδικα που δημοσιεύεται στο «Expert One-on-One J2EE Design and Development» από τον Rob Johnson και σκοπό έχει την διευκόλυνση της ανάπτυξης εφαρμογών Java. Το Spring ακολουθεί αρθρωτή αρχιτεκτονική και έχει διαιρεθεί σε ανεξάρτητα πακέτα τα οποία, όμως, μπορούν να λειτουργήσουν ανεξάρτητα με άλλα frameworks προσφέροντας την λειτουργικότητά τους με ελάχιστες απαιτούμενες ρυθμίσεις. Μειώνει την προσπάθεια για την ανάπτυξη μίας εφαρμογής καθώς και το κόστος, ενώ προσφέρει διευκολύνσεις στην αποτελεσματικότητα των δοκιμών.

Το πλαίσιο Spring έχει περίπου 20 ενότητες για την οργάνωση των χαρακτηριστικών του μερικές από τις οποίες φαίνονται στην Εικόνα 6.9. Οι βασικότερες υπηρεσίες που προσφέρει είναι:

- Inversion of Control container (IoC)
- Aspect-oriented Programming (AOP)
- Data Access
- Transaction Management
- Model-and-View Controller
- Remote Access Framework
- Batch Processing
- Authentication και Authorization

- Remote Management
- Messaging
- Testing



Εικόνα 6.9. *Συνοπτική αρχιτεκτονική του προγραμματιστικού πλαισίου Spring.*
Πηγή:http://techmyguru.com/spring/index.php?section=2/Architecture_of_Spring_Framework

Οι πιο σημαντικές λειτουργίες του Spring Framework είναι η Inversion-of Control (IoC) και η AOP (Aspect Oriented Programming). Στην πρώτη παρέχεται ένα σύνολο από τεχνικές στις οποίες η ροή ελέγχου αντιστρέφεται σε σχέση με το συμβατικό τρόπο που διεξάγεται χωρίς τη χρήση του πλαισίου. Το πλαίσιο καλεί τις διαδικασίες/μεθόδους που έχουν δημιουργηθεί από τον προγραμματιστή και όχι ο προγραμματιστής τις διαδικασίες/μεθόδους του πλαισίου. Η δεύτερη προσφέρει μία ευέλικτη λύση στην εφαρμογή κρίσιμων λειτουργιών, όπως η διαχείριση δοσοληψιών (transaction management).

Inversion of Control Container and Dependency Injection

Ένα από τα πιο σημαντικά χαρακτηριστικά του Spring είναι το IoC (Inversion of Control) ή αλλιώς Dependency Injection (DI). Κάθε μεγάλη εφαρμογή (Walls, 2015) αποτελείται από κλάσεις οι οποίες συνεργάζονται μεταξύ τους. Τα αντικείμενα λοιπόν αναλαμβάνουν μόνα τους να αποκτήσουν τις δικές τους συνδέσεις με τα αντικείμενα με τα οποία συνεργάζονται (δηλαδή να προσδιορίσουν την εξάρτησή τους με αυτά). Αυτό αποτελεί ένα είδος

“pull” σύνθεσης (pull configuration) (Rod Johnson, 2005) δηλαδή το αντικείμενο προσπαθεί μόνο του να αποσπάσει τις εξαρτήσεις από το περιβάλλον του.

Αντίθετα με το DI, οι εξαρτήσεις χορηγούνται (inject) στα αντικείμενα που τις χρειάζονται κατά τη δημιουργία τους. Αυτό το καθήκον ανατίθεται στον container, ο οποίος διαβιβάζει τα ονόματα των αντικειμένων σε άλλα αντικείμενα μέσω του κατασκευαστή (constructor), μεθόδων, ή “factory” μεθόδων. Σε αυτή την προσέγγιση δημιουργείται ένα είδος “push” σύνθεσης (push configuration). Με αυτό τον τρόπο οι εξαρτήσεις είναι σαφείς και οι κλάσεις τεκμηριώνονται από μόνες τους. Δεν είναι συνεπώς απαραίτητο ο προγραμματιστής να αναπτύξει τον κώδικα για τη σύνθεση μιας κλάσης, δεδομένου ότι αυτό διεκπεραιώνεται από το πλαίσιο εύκολα και γρήγορα.

Spring Boot

Το προγραμματιστικό πλαίσιο Spring Boot είναι ένα ανοιχτού κώδικα σύνολο βιβλιοθηκών στη γλώσσα Java, το οποίο παρέχει μια ιδιαίτερα περιεκτική από άποψη υποδομών υποστήριξη για ανάπτυξη εύρωστων εφαρμογών υλοποιημένων σε Java EE (Enterprise), με εύκολο και γρήγορο τρόπο. Το Spring Boot Framework γράφτηκε αρχικά από τον Rod Johnson, ενώ κυκλοφόρησε υπό την άδεια χρήσης του Apache 2.0 τον Ιούνιο του 2003. Βασικά χαρακτηριστικά του Spring Boot είναι:

1. Αυτόματη σύνθεση: Η σύνθεση των διάφορων λειτουργικοτήτων της εφαρμογής γίνεται αυτόματα. Η δυνατότητα αυτόματης σύνθεσης διασφαλίζει υψηλότερα επίπεδα ασφάλειας, ευκολότερη ανάπτυξη του μοντέλου Spring MVC, Java Persistence API κ.α.
2. Εκκινητής εξαρτήσεων: Ανάλογα με τη λειτουργικότητα που ο προγραμματιστής δηλώνει ότι είναι απαραίτητη για το έργο του, το Spring Boot αναλαμβάνει να συμπεριλάβει τις κατάλληλες βιβλιοθήκες. Ενδεικτικά, για την υλοποίηση μιας διαδικτυακής εφαρμογής, πρέπει να προστεθεί ο εκκινητής “web”. Αντίστοιχα, εάν γίνεται ανάπτυξη μιας εφαρμογής Mongo persistence τότε πρέπει να προστεθεί ο εκκινητής “mongo”, κ.ο.κ.
3. Διεπαφή γραμμής εντολών (CLI): είναι ένα εργαλείο γραμμής εντολών το οποίο χρησιμοποιείται προαιρετικά για γρήγορη ανάπτυξη με το Spring framework. Επιτρέπει την εκτέλεση scripts, τα οποία είναι περίπου όμοια με τον αντίστοιχο κώδικα Java.
4. Ενεργοποιητής: Δίνει τη δυνατότητα παρακολούθησης των διεργασιών που εκτελούνται στο πλαίσιο μιας εφαρμογής καθώς αυτή λειτουργεί. Οι αντίστοιχες πληροφορίες παρουσιάζονται μέσω web endpoints ή μέσω μια διεπαφής shell.

AngularJS

Το AngularJS (Dayley, 2014) (Pawel Kozlowski, 2013) είναι ένα client-side JavaScript προγραμματιστικό πλαίσιο το οποίο αναπτύχθηκε από την Google. Ανήκει στην οικογένεια των MVC (model-view-controller) framework.

Το μοντέλο MVC υλοποιείται στο προγραμματιστικό πλαίσιο AngularJS με HTML και JavaScript. Το επίπεδο View ορίζεται σε HTML, ενώ τα επίπεδα Model και Controller σε JavaScript.

Views

Μια view στην AngularJS ορίζεται μέσω της κατάλληλης σήμανσης με την ετικέτα ng-view. Υπό μία έννοια το συνολικό HTML έγγραφο αποτελεί τη view. Δεδομένου όμως ότι οι εφαρμογές AngularJS σχεδιάζονται ως single-page applications, μόνο ένα τμήμα της σελίδας αντιπροσωπεύει κάθε φορά την τρέχουσα view. Συνεπώς, μπορεί να θεωρηθεί ότι τα περιεχόμενα της ετικέτας BODY αντιπροσωπεύουν τη view και οι ετικέτες HEAD και HTML δημιουργούν το δοχείο για τις επιμέρους views. Ένα παράδειγμα σήμανσης μιας περιοχής της HTML σελίδας ως view στο πλαίσιο AngularJS παρατίθεται στο επόμενο τμήμα κώδικα:

```
<!DOCTYPE html>
<html>
<head>
<title>Hack Hands Angular - Demos</title>
<meta charset="utf-8" />
</head>
<body>
<h1>Hack Hands Angular Demos</h1>
<div ng-view>
<div id="messageTitle"></div>
<div id="message">Hello World</div>
</div>
</body>
</html>
```

Προκειμένου να λειτουργήσει η AngularJS view πρέπει να προβλεφθούν τα εξής:

1. Να κληθεί το πλαίσιο AngularJS μέσω κατάλληλης ετικέτας <script>:

```
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.5/angular.min.js">
</script>
```

2. Να οριστεί μέσω κατάλληλης ετικέτας η εφαρμογή AngularJS, η οποία εκτελείται μέσω της συγκεκριμένης view.

Controllers

Ένας controller στην AngularJS είναι ένας κατασκευαστής σε γλώσσα JavaScript, ο οποίος χρησιμοποιείται για την επαύξηση της λειτουργικότητας της σελίδας. Ο controller δηλώνεται στο DOM μέσω της ετικέτας "ng-controller". Η AngularJS δημιουργεί ένα αντικείμενο της κλάσης Controller με τη χρήση του συγκεκριμένου κατασκευαστή. Ένας controller χρησιμοποιείται για την απόκριση σε είσοδο από το χρήστη και για αλληλεπίδραση με τις views, ώστε να πραγματοποιούνται αλλαγές στη διεπαφή του χρήστη. Επιπλέον

χρησιμοποιούνται για τη συντήρηση του model και την τροποποίησή του. Ο κώδικας που ακολουθεί παρουσιάζει έναν απλό controller σε γλώσσα JavaScript.

```
var indexController = hackApp.controller("indexController", function
($scope) {
// controller logic goes here
}
);
```

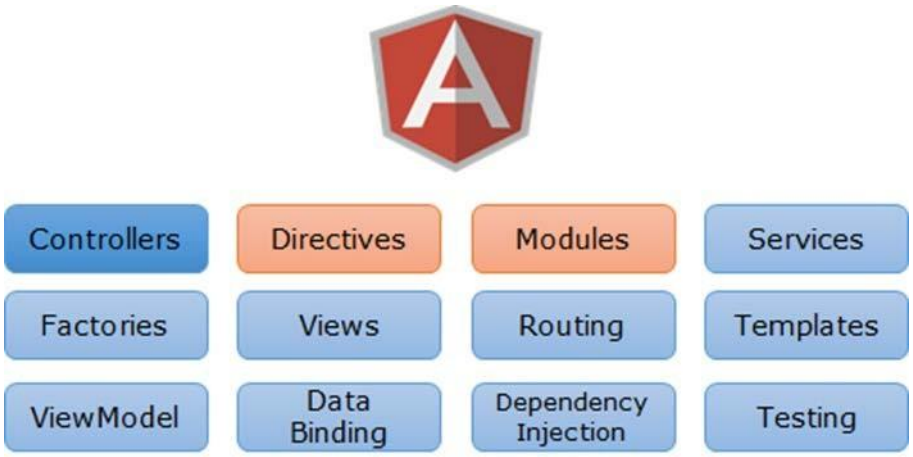
Models

Το μοντέλο που αντιστοιχεί σε έναν controller περιλαμβάνει τα δεδομένα που πρόκειται να εμφανιστούν σε μια σελίδα, όπως επίσης και τα δεδομένα τα οποία συλλέγονται μέσω φορμών. Επιπροσθέτως, τα μοντέλα μπορούν να περιλαμβάνουν συναρτήσεις οι οποίες ενεργοποιούνται από είσοδο δεδομένων του χρήστη ή άλλες δραστηριότητες, όπως την πίεση ενός κουμπιού ή από την αλλαγή των δεδομένων.

Η AngularJS διαθέτει την κλάση $scope, η οποία εάν ενσωματωθεί στον controller μπορεί να χρησιμοποιηθεί άμεσα ως μοντέλο. Βάσει των παραπάνω, ο κώδικας που ακολουθεί δηλώνει και θέτει τιμή σε ένα μοντέλο.

```
var indexController = hackApp.controller("indexController", function
($scope) {
// controller logic goes here
$scope.message = "Hello Hacking World"
}
);
```

Με βάση λοιπόν το MVC μοντέλο η AngularJS απαρτίζεται από κάποια δομικά στοιχεία (Εικόνα 6.10) τα οποία κάνουν την εφαρμογή στιβαρή και ευκολονόητη. Παρακάτω περιγράφουμε πιο αναλυτικά τη δομή και τον σκοπό που εξυπηρετούν μερικά από αυτά τα δομικά στοιχεία.



Εικόνα 6.10. *Η αρθρωτή αρχιτεκτονική της AngularJS*
Πηγή: https://codewala.net/2014/06/03/learning-angularjs-with-examples-part-2/

Modules

Το module είναι ένας container, στο εσωτερικό του οποίου υπάρχουν τα διάφορα δομικά στοιχεία της εφαρμογής όπως πχ. οι controllers. Κύριο χαρακτηριστικό του είναι ότι χαρακτηρίζεται από ένα όνομα και έτσι η αναφορά σε άλλα δομικά στοιχεία γίνεται βάσει του ονόματος αυτού. Με αυτό τον τρόπο πετυχαίνεται καλύτερη οργάνωση των διάφορων ενοτήτων της εφαρμογής καθώς και η παραμετροποίησή τους.

Data-Binding

Το Data-Binding είναι η σύνδεση των δεδομένων μεταξύ του μοντέλου και των στοιχείων που απεικονίζονται στην οθόνη (view). Αυτή η σύνδεση έχει διπλή κατεύθυνση. Πιο αναλυτικά, αν τα δεδομένα σε μια ιστοσελίδα τροποποιηθούν, τότε αυτόματα ανανεώνεται το μοντέλο και αντίστροφα.

Scope

Το Scope είναι ένα αντικείμενο (object) το οποίο με τις κατάλληλες μεθόδους και ιδιότητες πετυχαίνει τη σύνδεση μεταξύ του view και της JavaScript και πιο συγκεκριμένα του αρμόδιου Controller.

Directives

Όταν φορτώνεται μια σελίδα στο φυλλομετρητή (browser), δημιουργείται ένα αντικείμενο που ονομάζεται DOM (Document Object Model), το οποίο με χρήση JavaScript μπορεί να δημιουργήσει δυναμική HTML. Με τα directives που παρέχει η AngularJS είναι δυνατή η προσθήκη επιπλέον χαρακτηριστικών και κλάσεων στην HTML καθώς και η επαύξηση των δυνατοτήτων του DOM.

Expressions

Το AngularJS χρησιμοποιεί τα expressions σε μια σελίδα HTML, ώστε να εμφανίσει τα αποτελέσματα σε αυτή. Όταν η τιμή ενός expression αλλάξει, τότε αλλάζει δυναμικά και στην HTML σελίδα. Το ίδιο συμβαίνει και όταν αλλάξει η τιμή του στην HTML όπου μέσω του αντικειμένου scope ανανεώνεται αυτόματα και το μοντέλο.

Controllers

Ένας Controller είναι ένα JavaScript Object το οποίο ελέγχει τα δεδομένα της εφαρμογής. Αναλαμβάνει ουσιαστικά να αρχικοποιήσει την κατάσταση και τις τιμές του scope καθώς και να προσθέσει λειτουργικότητα σε αυτό.

Services

Τα services είναι μεμονωμένα αντικείμενα, τα οποία παρέχουν σημαντική λειτουργικότητα σε μια εφαρμογή. Μία από αυτές τις λειτουργικότητες και η πιο σημαντική είναι η αποστολή αιτήσεων σε ένα διαδικτυακό server με το ενσωματωμένο HTTP service. Εκτός από τα ενσωματωμένα, ένας προγραμματιστής μπορεί να δημιουργήσει και να χρησιμοποιήσει δικά του services.

Filters

Τα φίλτρα χρησιμοποιούνται στους controller, στα services και στην HTML και μορφοποιούν (format) τα δεδομένα και τις τιμές των εκφράσεων.

Puppeteer

Το Puppeteer είναι μια βιβλιοθήκη NodeJS του Google Chrome (Google, n.d.) η οποία παρέχει ένα API το οποίο ελέγχει το φυλλομετρητή Chrome που εκτελείται είτε στο προσκήνιο είτε στο παρασκήνιο. Είναι δυνατός ο πλήρης έλεγχος του φυλλομετρητή με απλές εντολές. Στο πλαίσιο αυτό καθίσταται εφικτή η έναρξη του φυλλομετρητή, η δημιουργία μιας καρτέλας και η φόρτωση μιας ιστοσελίδας, η περιήγηση σε αυτή και η εξόρυξη πληροφοριών. Ο κώδικας στην Εικόνα 6.11 που ακολουθεί αποτελεί ένα ενδεικτικό παράδειγμα κλήσης διαφορετικών λειτουργιών του φυλλομετρητή από το πρόγραμμα.

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://tuc.gr');
  const tucLogo = await page.evaluate(() =>
document.querySelector('brand').getAttribute('src'));

  browser.close();
})();
```

Εικόνα 6.11 *Περιήγηση στην ιστοσελίδα https://tuc.gr και εξόρυξη του logo του Πολυτεχνείου Κρήτης.*

MongoDB

Η MongoDB είναι ένα ανοιχτού κώδικα πρόγραμμα (Kristina Chodorow, 2010), σχεδιασμένο να αποθηκεύει και να διαχειρίζεται document-oriented πληροφορία, δηλαδή δεδομένα τα οποία είναι ημιδομημένα. Αναπτύχθηκε και συντηρείται από την MongoDB Inc. Μπορεί να τρέξει σε διαφορετικά λειτουργικά συστήματα (cross platform) και ανήκει στην οικογένεια των NoSQL βάσεων δεδομένων, είναι δηλαδή μη σχεσιακή. Σε αντίθεση με τις γνωστές σε όλους σχεσιακές SQL βάσεις δεδομένων στη MongoDB δεν υπάρχει η έννοια των γραμμών, των εξαρτήσεων, των πινάκων (tables), ούτε συναντάμε joins και foreign keys. Αυτό που συναντάμε είναι έγγραφα τύπου JSON με ευέλικτα σχήματα, με αποτέλεσμα σε πολλές εφαρμογές να έχουμε πιο γρήγορη και εύκολη συγχώνευση δεδομένων.

Έγγραφα και συλλογές

Η MongoDB όπως προαναφέραμε αποθηκεύει έγγραφα (documents) τύπου JSON, τα οποία αναπαρίστανται με κλειδιά και τιμές. Συνήθως συναντάμε έγγραφα με πολλαπλά κλειδιά και τιμές. Η Εικόνα 6.12 παρουσιάζει ένα τέτοιο έγγραφο.

Εικόνα 6.12. *Έγγραφο με πεδία που έχουν τιμή ένα υποέγγραφο.*

Πηγή: https://docs.mongodb.com/manual/core/data-modeling-introduction/

Όπως στις σχεσιακές βάσεις δεδομένων έχουμε τους πίνακες, αντίστοιχα στη MongoDB έχουμε τις συλλογές (collections). Σε αυτές αποθηκεύονται έγγραφα σχετικά μεταξύ τους (πχ. έγγραφα με κοινό index). Η Εικόνα 6.13 παρουσιάζει μια συλλογή με πολλά έγγραφα.

Εικόνα 6.13. *Συλλογή με πολλαπλά έγγραφα.*
Πηγή: ***https://www.w3resource.com/mongodb/databases-documents-collections.php***

Οι συλλογές δεν έχουν καθορισμένο σχήμα. Συνεπώς τα έγγραφα μπορούν να έχουν διαφορετική δομή και να βρίσκονται στην ίδια συλλογή.

Querying

Τα ερωτήματα (queries) είναι ουσιαστικά ερωτήσεις-αιτήσεις με συγκεκριμένα κριτήρια, που υποβάλλει ένας χρήστης στη βάση δεδομένων σε μια συγκεκριμένη συλλογή, η οποία αφού εντοπίσει τα ενδιαφερόμενα έγγραφα τα επιστρέφει στον χρήστη.

Τα ερωτήματα μπορεί να χρησιμοποιηθούν για να την εύρεση, εισαγωγή, διαγραφή ή ενημέρωση ενός ή περισσότερων εγγράφων στη βάση δεδομένων. Οι ενέργειες αυτές μπορούν να διεκπεραιώνονται με χρήση κριτηρίων σχεσιακούς τελεστές, όρια, indexes, κ.α.

Indexing

Για τη βελτίωση της απόδοσης στην εκτέλεση επερωτήσεων η MongoDB χρησιμοποιεί indexes. Απουσία indexes, η αναζήτηση σε μία συλλογή διεξάγεται με σάρωση όλων των εγγράφων μιας συλλογής ένα προς ένα.

Aggregation

Τα aggregations είναι ένα σύνολο από συναρτήσεις που επιτρέπουν το χειρισμό δεδομένων που επιστρέφει η MongoDB ως απάντηση σε μια επερώτηση. Οι λειτουργίες συνάθροισης ομαδοποιούν τις τιμές από τα πολλαπλά έγγραφα, και πραγματοποιώντας κάποιες λειτουργίες σε αυτές, επιστρέφουν ένα συμπαγές αποτέλεσμα.

Η MongoDB παρέχει τρεις μεθόδους συνάθροισης δεδομένων: τον αγωγό συνάθροισης (aggregation pipeline), την συνάρτηση map-reduce και τις απλού σκοπού λειτουργίες συνάθροισης.

7 Διεπαφές Χρήστη

Σκοπός της εργασίας είναι η αναζήτηση πτήσεων και εισιτηρίων εύκολα, γρήγορα και αποτελεσματικά. Η εφαρμογή σχεδιάστηκε από τα πρώτα της βήματα με κριτήριο ο σχεδιασμός των σελίδων να είναι βελτιστοποιημένος για την εμπειρία χρήστη και την ευχρηστία. Απευθύνεται σε χρήστες όλων των ηλικιών, με καλή ή σχετική εξοικείωση με διαδικτυακές εφαρμογές.

Επιπλέον, λαμβάνοντας υπόψη τις ανάγκες που έχουν γεννηθεί με την χρήση έξυπνων κινητών τηλεφώνων δεν θα μπορούσαμε να μην αναπτύξουμε σελίδες για αυτά που θα έχουν πλήρη απόκριση και λειτουργικότητα.

Στη συνέχεια αναλύονται μερικά από τα χαρακτηριστικά που θέτουμε, ώστε να επιτευχθούν τα παραπάνω.

Απλότητα : Η απλή, γρήγορη και ολοκληρωμένη εμπειρία χρήστη στο διαδικτυακό σχεδιασμό είναι ένα από τα πιο δύσκολα σημεία που έρχονται να αντιμετωπίσουν σχεδιαστές και προγραμματιστές. Κύριο χαρακτηριστικό των σύγχρονων εφαρμογών είναι η απλότητα. Ο χρήστης πρέπει να είναι σε θέση να περιηγείται στην εφαρμογή με αυτοπεποίθηση και άνεση. Έτσι στόχος της εφαρμογής που υλοποιήθηκε στα πλαίσια αυτής της διπλωματικής, είναι ο χρήστης να έχει μια εμπειρία που θα είναι ψυχολογικά άνετη και οπτικά ξεκούραστη. Γι' αυτό και οι σελίδες σχεδιάστηκαν με τέτοιο τρόπο ώστε να είναι εύκολα κατανοήσιμες, απλές, με ευδιάκριτο περιεχόμενο και λίγα χρώματα που δεν κουράζουν ούτε συγχέουν τον χρήστη.

Οπτική ιεραρχία : Τα στοιχεία μιας σελίδας πρέπει να είναι οργανωμένα και τοποθετημένα στα κατάλληλα σημεία έτσι ώστε μόλις οι χρήστες ανοίξουν την εφαρμογή να μπορούν να εστιάσουν πρώτα στα πιο σημαντικά από αυτά. Μεγάλο ρόλο σε αυτό παίζει και η σειρά με την οποία εμφανίζονται τα στοιχεία, το μέγεθός τους κ.α.

Πλοηγησιμότητα : Ένα ακόμα πολύ σημαντικό χαρακτηριστικό που πρέπει να έχει μια διαδικτυακή εφαρμογή είναι η εύκολη πλοήγηση. Πρέπει να είναι πλήρως κατανοητή από το χρήστη, δηλαδή ο χρήστης πρέπει, μόλις ανοίξει την εφαρμογή, να είναι σε θέση να καταλάβει γρήγορα τί πρέπει να συμπληρώσει, στη συνέχεια ποιο κουμπί θα πατήσει, πως θα γυρίσει στην προηγούμενη σελίδα, κτλ.

Διεπαφές Χρήστη

Έχοντας λοιπόν ως βασικό άξονα όλα τα παραπάνω και με τη βοήθεια και αξιολόγηση που έγινε από χρήστες διαμορφώθηκαν οι σελίδες της εφαρμογής, ώστε αυτές να συμπεριφέρονται εργονομικά και ορθά τόσο σε υπολογιστές γραφείου όσο και σε φορητές συσκευές (tablets, smartphones).

Σελίδα αναζήτησης

Η σελίδα αναζήτησης είναι η πρώτη σελίδα που συναντά ο χρήστης. Περιέχει μια φόρμα αναζήτησης με το δρομολόγια και τα χαρακτηριστικά του ταξιδιού.

Πιο αναλυτικά η φόρμα έχει ως εξής :

Επιλογή ταξιδιού **:** Εδώ ο χρήστης επιλέγει αν το ταξίδι του θα έχει επιστροφή ή θα είναι μίας κατεύθυνσης.

Αεροδρόμιο αναχώρησης **:** Εδώ ο χρήστης πληκτρολογεί το αεροδρόμιο αναχώρησης και αυτόματα εμφανίζεται μια λίστα με πιθανά αεροδρόμια που ψάχνει ο χρήστης ανάλογα με τα γράμματα που έχει πληκτρολογήσει.

Αεροδρόμιο αναχώρησης : Αντίστοιχα ο χρήστης ψάχνει το αεροδρόμιο προορισμού.

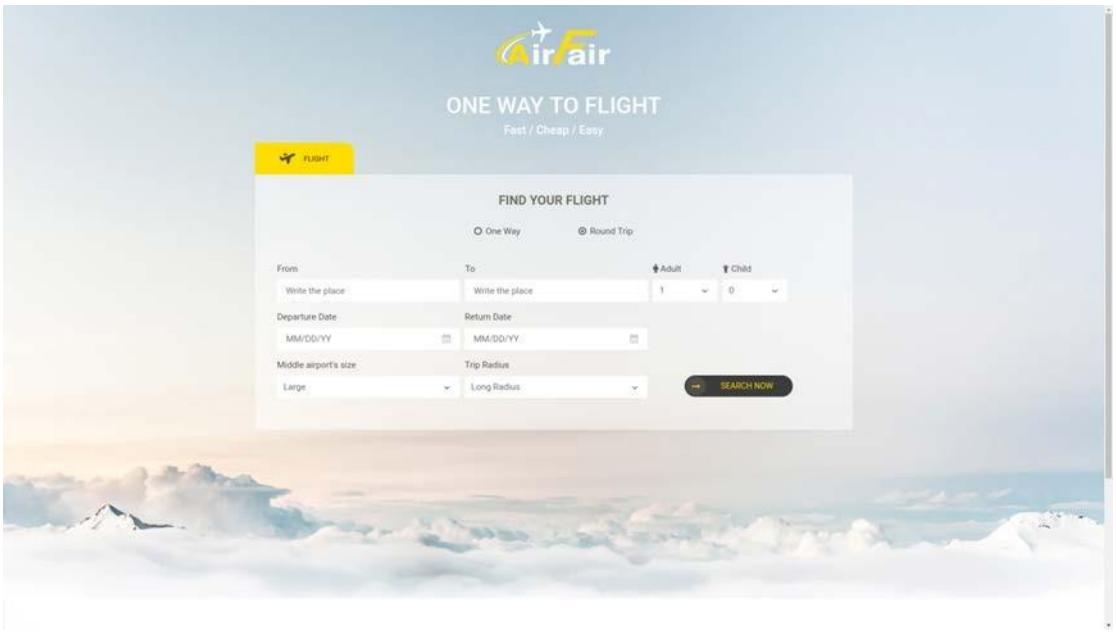
Ημερομηνία αναχώρησης : Επιλογή ημερομηνίας αναχώρησης μέσα από ένα γραφικό περιβάλλον ημερολογίου.

Ημερομηνία επιστροφής : Επιλογή ημερομηνίας επιστροφής μόνο εάν χρήστης έχει δηλώσει ότι επιθυμεί εύρεση εισιτηρίων με επιστροφή.

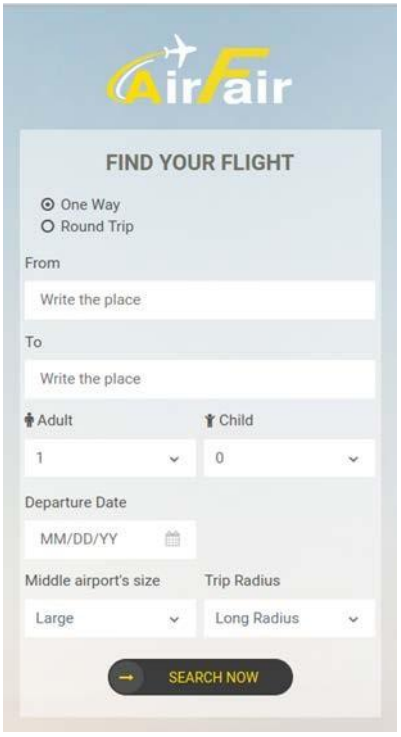
Μέγεθος ενδιάμεσων αεροδρομίων : Ο χρήστης επιλέγει το μέγεθος των ενδιάμεσων αεροδρομίων που επιθυμεί να μεταβεί (αν υπάρχουν) πριν φτάσει στον προορισμό του. Οι πιθανές επιλογές είναι μεγάλο, μεσαίο και μικρό μέγεθος.

Ακτίνα ενδιάμεσων αεροδρομίων : Η ακτίνα ενδιάμεσων αεροδρομίων ορίζεται γύρω από το κέντρο του μέσου των αεροδρομίων αναχώρησης και προορισμού. Επιπροσθέτως, δίνεται η δυνατότητα στο χρήστη, να επιλέξει ανάμεσα σε 2 επιλογές μεγέθους τη μικρή και μεγάλη, ώστε η εφαρμογή να ψάξει για πιθανούς ενδιάμεσους σταθμούς σε αεροδρόμια, μέσα σε αυτή την ακτίνα.

Αριθμός ενήλικων και παιδιών : Ο χρήστης δηλώνει τον αριθμό των ενήλικων και παιδιών που επιθυμούν να ταξιδέψουν μαζί, ώστε να βρει μία ενιαία τιμή για όλους. Στην Εικόνα 7.1 παρουσιάζεται η αρχική σελίδα της εφαρμογής που περιέχει τη φόρμα αναζήτησης. Αντίστοιχα στην Εικόνα 7.2 παρουσιάζεται η ίδια σελίδα για κινητό.



Εικόνα 7.1. *Αρχική σελίδα με φόρμα αναζήτησης πτήσεων.*



Εικόνα 7.2. *Αρχική σελίδα με φόρμα αναζήτησης πτήσεων (κινητό).*

Σελίδα αποτελεσμάτων (μία διαδρομή)

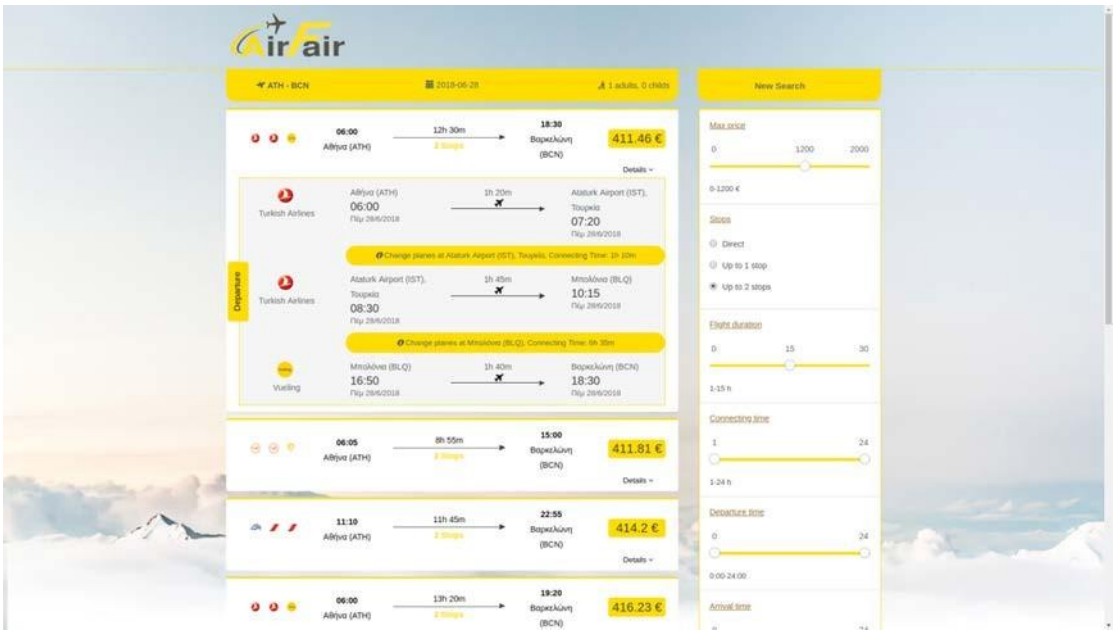
Στη σελίδα αυτή παρουσιάζονται όλα τα δυνατά αποτελέσματα εισιτηρίων ταξινομημένα με βάση την τιμή μαζί με μία μπάρα πληροφοριών και μια στήλη με τα φίλτρα αναζήτησης.

Μπάρα πληροφοριών : Στην μπάρα με τις πληροφορίες υπάρχει ο αρχικός και τελικός προορισμός, η ημερομηνία αναχώρησης καθώς και ο αριθμός των ενήλικων και παιδιών που έχουν δηλωθεί ότι θα ταξιδέψουν. Επίσης υπάρχει και ένα κουμπί για νέα αναζήτηση που σε γυρνάει στη σελίδα με τη φόρμα αναζήτησης.

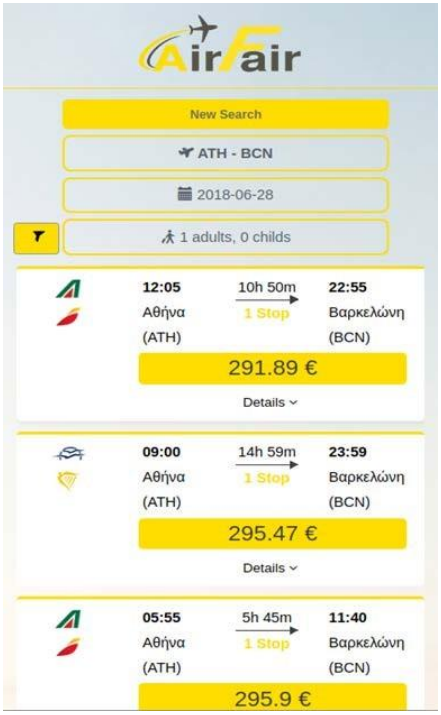
Στήλη με φίλτρα αναζήτησης **:** Στη δεξιά στήλη εμφανίζονται τα φίλτρα όπου ο χρήστης μπορεί να κάνει χρήση τους αν το επιθυμεί για να επιλέξει εισιτήρια της αρεσκείας του. Τα κριτήρια επιλογής είναι με βάση την τιμή, τις ενδιάμεσες στάσεις, την διάρκεια πτήσης, την ώρα αναμονής στους ενδιάμεσους σταθμούς και τέλος την ώρα αναχώρησης και άφιξης.

Λίστα αποτελεσμάτων : Στο κέντρο της σελίδας έχουμε μια λίστα με τα εισιτήρια που βρέθηκαν, ταξινομημένα με βάση την τιμή. Κάθε δρομολόγιο αναγράφει όλες τις πληροφορίες του ταξιδιού όπως διάρκεια πτήσεις, αριθμό πιθανών στάσεων, ώρες αναχώρησης και άφιξης, τιμή εισιτήριου καθώς και την αεροπορική-ές εταιρεία-ες του ταξιδιού. Επίσης, πατώντας το κουμπί "λεπτομέρειες" μπορεί να δει περισσότερες πληροφορίες αναφορικά με το ταξίδι όπως την αεροπορική εταιρεία κάθε πτήσης, τις ώρες και ημερομηνίες των ενδιάμεσων σταθμών, τη διάρκεια πτήσεων από και προς αυτούς καθώς και τους χρόνους αναμονής στα αεροδρόμια των ενδιάμεσων αυτών σταθμών.

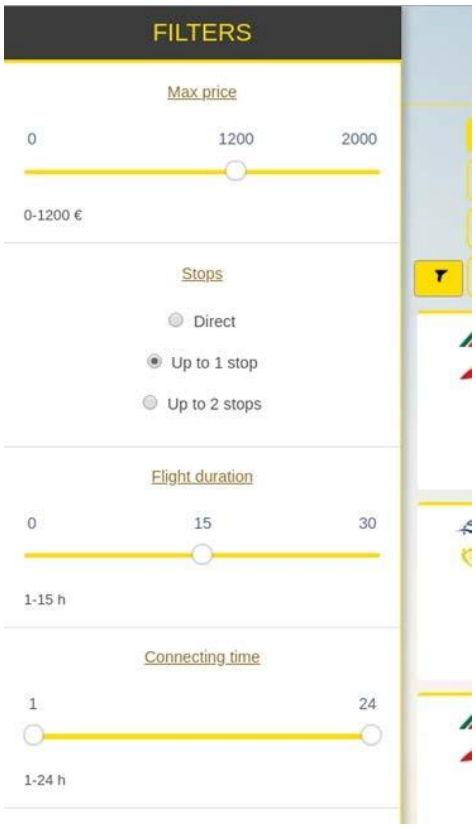
Η εικόνα 7.3 παρουσιάζει τη σελίδα με τα αποτελέσματα πτήσεων και τα φίλτρα (desktop εφαρμογή). Στην Εικόνα 7.4 και στην Εικόνα 7.5 παρουσιάζεται η σελίδα με τα αποτελέσματα και τα φίλτρα αντίστοιχα για κινητό.



Εικόνα 7.3. *Σελίδα με τα αποτελέσματα πτήσεων και τα φίλτρα*



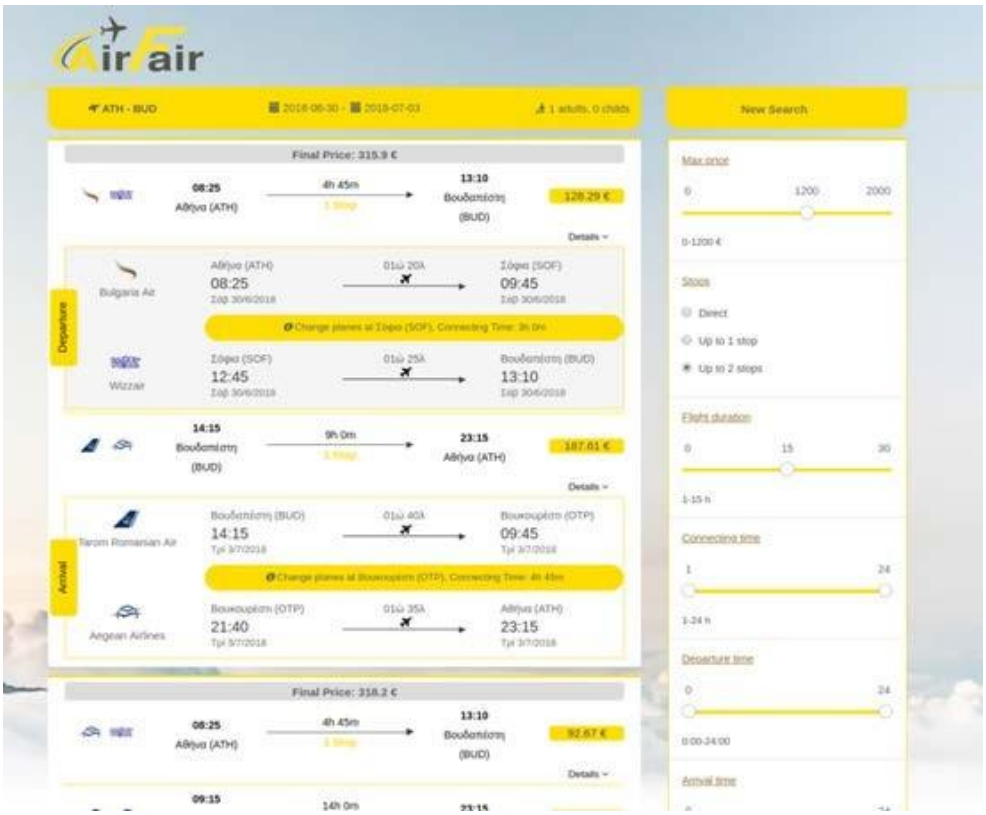
Εικόνα 7.4. *Τα αποτελέσματα των πτήσεων (κινητό)*



Εικόνα 7.5. *Τα φίλτρα αναζήτησης (κινητό)*

Σελίδα αποτελεσμάτων (Με επιστροφή)

Στη σελίδα αυτή προστίθεται στην μπάρα με τις πληροφορίες η ημερομηνία επιστροφής και στη λίστα με τα αποτελέσματα πτήσεων έχουμε ένα πεδίο που αναγράφει την τελική τιμή του ταξιδιού καθώς και πληροφορίες για το δρομολόγιο αναχώρησης και επιστροφής. Στην Εικόνα 7.6 παρουσιάζεται η σελίδα με τα αποτελέσματα πτήσεων ταξιδιού με επιστροφή, ενώ στην Εικόνα 7.7 παρουσιάζεται η ίδια σελίδα για κινητό.



Εικόνα 7.6. *Σελίδα με τα αποτελέσματα πτήσεων ταξιδιού με επιστροφή*

Εικόνα 7.7. *Αποτελέσματα πτήσεων ταξιδιού με επιστροφή (κινητό)*

Αξιολόγηση Ευχρηστίας Συστήματος

Ένα άλλο πολύ σημαντικό και αναγκαίο κομμάτι της ανάπτυξης της εφαρμογής είναι η αξιολόγηση (Christodoulakis, 2012) η οποία έρχεται αφού τελειώσει ο σχεδιασμός των σελίδων και συνεχίζει σε κάθε στάδιο ανάπτυξης της εφαρμογής. Στην εφαρμογή μας έγινε αξιολόγηση με τη μέθοδο think aloud τόσο σε αρχικό όσο και σε τελικό στάδιο.

Think aloud

Το πρωτόκολλο think aloud είναι ένα πρωτόκολλο που χρησιμοποιείται για να συλλέξει δεδομένα από δοκιμές ευχρηστίας (usability testing). Εμπλέκει χρήστες οι οποίοι αναλαμβάνουν να πραγματοποιήσουν κάποιες καθορισμένες ενέργειες οι οποίες τους έχουν ζητηθεί, ενώ παράλληλα πρέπει να εκφράζουν φωναχτά την οποιαδήποτε σκέψη. Πιο συγκεκριμένα, οι χρήστες πρέπει να λένε ό,τι βλέπουν, ό,τι τους φάνηκε περίεργο, ό,τι νιώθουν, και αυτό να κρατείται (αποτυπώνεται;) από τους παρατηρητές με τη μορφή σημειώσεων, ή ακόμη καλύτερα με τη μορφή βίντεο. Στην τελευταία περίπτωση οι προγραμματιστές μπορούν να χρησιμοποιήσουν το βίντεο για να δουν πώς αντέδρασαν οι χρήστες σε αυτά που τους ζητήθηκαν να κάνουν και σε αυτά που είδαν κατά τη διάρκεια της δοκιμής. Κατ' αυτό τον τρόπο σχηματίζεται μια πιο ολοκληρωμένη εικόνα πάνω στις λειτουργικότητες που πρέπει να αναπτυχθούν, στη σχεδίαση των σελίδων καθώς και στην οργάνωση και επιμέλεια του έργου.

Αρχικό στάδιο

Αφού τελειώσει ο σχεδιασμός των σελίδων εφαρμόζουμε το πρωτόκολλο think aloud. Για τις ανάγκες της εφαρμογής έγινε αξιολόγηση από 3 χρήστες των οποίων η επιλογή δεν έγινε καθόλου τυχαία.

Οι δύο χρήστες ήταν σχεδιαστές διαδικτυακών ιστοσελίδων και ο τρίτος ένας φοιτητής. Οι δύο πρώτοι εστίασαν στην απλότητα που έπρεπε να δείχνουν πιο πολύ οι διεπαφές που είχαν σχεδιαστεί στο χαρτί. Έδωσαν συμβουλές για την πλοηγησιμότητα της εφαρμογής και για την θέση της πλαϊνής στήλης με τα αποτελέσματα και προέτρεψαν σε αλλαγή θέσης της. Ο φοιτητής έδωσε έμφαση σε λειτουργικότητες που έλειπαν και θα ήθελε να έχει η εφαρμογή. Αφού λοιπόν τελείωσε η αξιολόγηση και έγιναν οι απαραίτητες διορθώσεις ρωτήθηκαν πάλι και αυτή τη φορά τα αποτελέσματα ήταν πολύ θετικά.

Τελικό στάδιο

Μετά την υλοποίηση των ιστοσελίδων έγινε και η αξιολόγηση σε πραγματικό χρόνο με περιήγηση στις διαδικτυακές πια ιστοσελίδες. Η αξιολόγηση έγινε από τρεις χρήστες. Και σε αυτή την περίπτωση η επιλογή δεν έγινε τυχαία.

Η πρώτη αξιολόγηση έγινε από ένα προγραμματιστή διεπαφών χρήστη με μεγάλη εμπειρία στη σχεδίαση και κατασκευή ιστοσελίδων. Οι σκέψεις του κινήθηκαν γύρω από την

αισθητική των σελίδων τα χρώματα και τη λειτουργικότητά τους. Πιο συγκεκριμένα, στη σελίδα των αποτελεσμάτων τόνισε την ανάγκη ενός κουμπιού για νέα αναζήτηση εισιτηρίων και ανέφερε κάποιες αλλαγές στα χρώματα της λίστας με τα αποτελέσματα των πτήσεων.

Η δεύτερη αξιολόγηση έγινε από ένα φοιτητή ο οποίος εστίασε περισσότερο στην λειτουργία της εφαρμογής από κινητό τηλέφωνο. Εξέφρασε σκέψεις για τη διάταξη των αποτελεσμάτων των πτήσεων καθώς και μια δυσανασχέτηση για τη θέση και τον όγκο που καταλάμβαναν τα φίλτρα των πτήσεων στη σελίδα με τα αποτελέσματα.

Η τρίτη και τελευταία αξιολόγηση έγινε από ένα χρήστη ηλικίας 50 ετών ο οποίος δεν είχε μεγάλη εξοικείωση με το διαδίκτυο και τα έξυπνα τηλέφωνα. Οι σκέψεις του ωστόσο βοήθησαν σε μεγάλο βαθμό, ώστε να εξεταστεί η απλότητα της εφαρμογής και η λειτουργικότητά της. Οι σκέψεις κινήθηκαν γύρω από τις θέσεις των φορμών αναζήτησης, των κουμπιών, των φίλτρων, καθώς και γύρω από την πλοήγηση στη σελίδα των αποτελεσμάτων όπου μέσα από την πλοήγηση του χρήστη στις διάφορες πτήσεις και την εισαγωγή φίλτρων εξετάστηκε το κατά πόσο είναι κατανοητά τα αντικείμενα της σελίδας και η διάταξή τους.

Αφού έγιναν οι απαραίτητες διορθώσεις και σε αυτό το στάδιο της εφαρμογής ρωτήθηκαν πάλι αν είναι ικανοποιημένοι από αυτές και η απάντηση ήταν θετική.

Απόδοση του Συστήματος

Στο κεφάλαιο αυτό παρουσιάζουμε αναλυτικά τις μετρήσεις που έγιναν για να μελετηθεί η απόδοση του συστήματός μας.

Οι μετρήσεις που πήραμε ήταν γύρω από τους χρόνους που απαιτούνται για την εύρεση των εισιτηρίων. Με κριτήριο λοιπόν την απόσταση ανάμεσα σε αεροδρόμιο αναχώρησης και άφιξης, το μέγεθος των ενδιάμεσων αεροδρομίων που επέλεξε ο χρήστης, καθώς και το μέγεθος της ακτίνας που θα βρίσκονται τα ενδιάμεσα αυτά αεροδρόμια υπολογίζεται για κάθε ταξίδι ο αριθμός των ενδιάμεσων αεροδρομίων που πρέπει να ψάξει το σύστημα για τυχόν πτήσεις και εισιτήρια. Πιο αναλυτικά οι δοκιμές έγιναν για 0, 20, 60 και 100 ενδιάμεσα αεροδρόμια σε τοπικό server με τα εξής χαρακτηριστικά : Μνήμη RAM 8gb, 4 CPU.

Ενδιάμεσοι σταθμοί	Χρόνος (sec)
0	13
20	50
60	120
100	250

Από τα αποτελέσματα συμπεραίνουμε ότι ο χρόνος που χρειάζεται για την εύρεση εισιτηρίων εξαρτάται από το πλήθος των ενδιάμεσων αεροδρομίων που καλείται να ψάξει το σύστημα. Όσο λιγότερα είναι τα ενδιάμεσα αεροδρόμια τόσο πιο γρήγορη είναι και η αναζήτηση. Αυτό οφείλεται στο γεγονός ότι το εργαλείο scraping που χρησιμοποιούμε χρειάζεται περισσότερο χρόνο να φορτώσει περισσότερες σελίδες ταυτόχρονα, σε ένα Η/Υ όπως έγινε και στην περίπτωσή μας. Ίσως οι χρόνοι αυτοί να ήταν διαφορετικοί αν η εφαρμογή ‘‘έτρεχε’’ σε έναν εξυπηρετητή με καλύτερους πόρους.

Συγκριτικά με άλλες εφαρμογές αναζήτησης αεροπορικών εισιτηρίων που υπάρχουν στο διαδίκτυο, ο χρόνος που χρειάζεται η εφαρμογή μας είναι περισσότερος. Αυτό συμβαίνει διότι οι εφαρμογές αυτές χρησιμοποιούν διεπαφές προγραμματισμού (API) όπου γίνεται εύκολη και αυτοματοποιημένη ανταλλαγή δεδομένων κι αιτήσεων μεταξύ εφαρμογής και προγραμμάτων, πάντα σε συνεργασία με διάφορες αεροπορικές εταιρείες και εταιρείες εύρεσης εισιτηρίων.

Απεναντίας όμως, μπορεί ο χρόνος να μην είναι αυτός που θα περίμενε ο χρήστης όμως τα αποτελέσματα είναι πλήρως ικανοποιητικά για τους στόχους που είχαν τεθεί. Την αναζήτηση δηλαδή εισιτηρίων μη λαμβάνοντας υπ’ όψη συνεργασίες εταιρειών και συνδυάζοντας ενδιάμεσους σταθμούς σε αεροδρόμια που το μέγεθος και η απόσταση επιλέγεται από το χρήστη.

8 Μελλοντικές Επεκτάσεις

Το σύστημα που αναπτύχθηκε στο πλαίσιο της εργασίας αποτελεί προϊόν τόσο συστηματικής σχεδίασης των πολλαπλών ενοτήτων που το απαρτίζουν, όσο και εντατικής υλοποίησής του σε προγραμματιστικό επίπεδο. Στο πλαίσιο αυτό, δεν ήταν δυνατή η χρήση του συστήματος ως ενός παραγωγικού εργαλείου, παρά η παρουσίασή του ως ένα πρωτότυπο, στο οποίο μπορεί να βασιστεί μελλοντική εργασία και μελλοντικές επεκτάσεις.

Η συστηματική εργασία με το σύστημα αυτό και η θέση του σε λειτουργία για την εύρεση εισιτηρίων, έθεσε το υπόβαθρο για ορισμένες σκέψεις, οι οποίες συνεισφέρουν σε επεκτάσεις του, τόσο λειτουργικά όσο και τεχνολογικά.

Αναδρομική αναζήτηση ενδιάμεσων στάσεων

Το σύστημα αποδείχθηκε ιδιαίτερα αποδοτικό στην εξεύρεση των πιο οικονομικών εισιτηρίων, δεδομένου ότι συμπεριλαμβάνει στις αναζητήσεις του εναλλακτικές διαδρομές μεταξύ αφετηρίας και προορισμού, οι οποίες απαρτίζονται από ένα έως 3 σκέλη (δύο ενδιάμεσοι προορισμοί). Ο μηχανισμός αυτός ενσωματώθηκε τόσο στη βάση δεδομένων, όπου καταχωρούνται για κάθε αεροδρόμιο τα γειτονικά του (βάσει των γεωγραφικών τους συντεταγμένων) όσο και στον controller του διακομιστή, ο οποίος και αναζητά τις διαδρομές και είναι σε θέση να διεξάγει υπολογισμούς για δύο μόνο ενδιάμεσους σταθμούς μεταξύ αφετηρίας και προορισμού. Αυτόματα δημιουργείται η απαίτηση να ενσωματωθεί στο σύστημα αλγόριθμος τέτοιος ο οποίος θα υπολογίζει με αναδρομικό τρόπο οσεσδήποτε ενδιάμεσες στάσεις, μέχρις ενός ορίου που θα θέτει ο πελάτης. Αυτή η δυνατότητα θα αποδειχτεί ιδιαίτερα χρήσιμη στο σχεδιασμό αεροπορικών ταξιδιών μεγάλων αποστάσεων, όπου η ύπαρξη ενδιάμεσων σταθμών μπορεί να οδηγήσει σε δραστική μείωση του συνολικού κόστους του ταξιδιού.

Εξόρυξη δεδομένων από πολλαπλές πηγές

Σε αυτό το επίπεδο η εφαρμογή εξορύσσει δεδομένα ανακαλώντας σελίδες μόνο από έναν ιστότοπο κρατήσεων αεροπορικών εισιτηρίων. Με σκοπό να παρέχεται η δυνατότητα ευρύτερης σύγκρισης τιμών και ενδεχομένως η εύρεση πιο οικονομικών εισιτηρίων για την ίδια αφετηρία και προορισμό, ενδιαφέρον αποκτά η ενσωμάτωση διαδικασίας εξόρυξης και άλλων ιστότοπων κρατήσεων αεροπορικών εισιτηρίων.

Προκειμένου η διαδικασία αυτή να μπορεί να γενικευτεί στο πλαίσιο του συστήματος, είναι δυνατή η ανάπτυξη ενός μηχανισμού περιγραφής των ιστοσελίδων αποτελεσμάτων για τον κάθε ιστότοπο που θα ενσωματώνεται στο σύστημα για web scraping. Ο μηχανισμός αυτός θα καταλήγει στη δημιουργία ενός αρχείου περιγραφής της εκάστοτε HTML σελίδας αποτελεσμάτων, το οποίο θα χρησιμοποιείται από τη μηχανή web scraping για την εξόρυξη των αντίστοιχων δεδομένων.

Συνεπώς, είναι δυνατή η δημιουργία μιας πλατφόρμας για την ενσωμάτωση αναζητήσεων σε πολλαπλές υπηρεσίες κρατήσεως αεροπορικών εισιτηρίων, ώστε ο προγραμματιστικός φόρτος για κάθε επιπρόσθετο ιστότοπο να περιορίζεται μόνο στη γλώσσα περιγραφής των σελίδων των αποτελεσμάτων που παρέχει στον τελικό χρήστη.

Εξόρυξη δεδομένων για άλλου είδους μεταφορικά μέσα

Εκτός των αεροπορικών μετακινήσεων, σε παγκόσμιο επίπεδο διαμορφώνεται τάση αναζήτησης οικονομικών τρόπων μετακίνησης με εναλλακτικά μέσα, όπως πλοία ή λεωφορεία. Αποτελεί συνεπώς ένα πεδίο μελλοντικής επέκτασης για το σύστημα που αναπτύχθηκε στο πλαίσιο της εργασίας αυτής η ενσωμάτωση υπηρεσιών για τη χρήση του σε οδικές ή θαλάσσιες μεταφορές. Σε αυτή την περίπτωση θα πρέπει να ενσωματωθούν αντίστοιχοι κανόνες για τους ενδιάμεσους σταθμούς καθώς και να ενημερωθεί η βάση δεδομένων με τα γεωγραφικά στοιχεία σταθμών υπεραστικών λεωφορείων αλλά και επιβατηγών πλοίων.

Οι εφαρμογές ενός συστήματος εξόρυξης δεδομένων από ιστοσελίδες περιορίζονται μόνο από το εύρος της εκάστοτε εφαρμογής και την πολυπλοκότητα των HTML σελίδων που εξορύσσονται. Συνεπώς, και βάσει των ανωτέρω, οι προτεινόμενες επεκτάσεις του συστήματος συντείνουν στη διεύρυνση του πεδίου εφαρμογής αλλά και στη βελτίωση της ποιότητας των παραγόμενων αποτελεσμάτων.

9 *Κατακλείδα*

Στο πλαίσιο της διπλωματικής εργασίας σχεδιάστηκε και υλοποιήθηκε ένα σύστημα εύρεσης αεροπορικών εισιτηρίων, το οποίο αντλεί τα δεδομένα του από ιστοσελίδες οι οποίες διατίθενται ελεύθερα στο διαδίκτυο, μέσω μηχανισμού web scraping. Το λογισμικό σχεδιάστηκε, ώστε να ενσωματώνει πρωτοποριακές τεχνολογίες και προγραμματιστικά πλαίσια, να ανταποκρίνεται σε μεγάλο όγκο δοσοληψιών και να είναι συμβατό τόσο με υπολογιστές γραφείου όσο και με έξυπνα κινητά τηλέφωνα και φορητές συσκευές. Στόχος ήταν η εύρεση αεροπορικών δρομολογίων τα οποία ανταποκρίνονται στις απαιτήσεις του ταξιδιού του χρήστη, αναζητώντας την οικονομικότερη λύση και πραγματοποιώντας κατάλληλους συνδυασμούς ενδιάμεσων αεροδρομίων.

Επίσης ιδιαίτερη προσοχή δόθηκε και στο σχεδιασμό των σελίδων της εφαρμογής, με στόχο να είναι ευκολονόητες και φιλικές προς τους χρήστες ανεξαρτήτου ηλικίας και εμπειρίας σε τέτοιου είδους διαδικτυακές εφαρμογές. Αυτό μεταφράζεται στη δημιουργία ενός απλοποιημένου περιβάλλοντος εισαγωγής των κριτηρίων αλλά και επισκόπησης των αποτελεσμάτων.

Πιο συγκεκριμένα, η αναζήτηση υλοποιήθηκε με βάση κριτήρια του χρήστη τα οποία κάνει γνωστά στο σύστημα συμπληρώνοντας μια φόρμα αναζήτησης. Εκτός από τα απαραίτητα στοιχεία αναζήτησης όπως τα αεροδρόμια και ημερομηνίες αναχώρησης και προορισμού, ο χρήστης έχει τη δυνατότητα να δηλώσει παραμέτρους όπως ο αριθμός ατόμων και εάν στους επιβάτες υπάρχουν παιδιά.

Η εφαρμογή διαφοροποιείται από αντίστοιχες υλοποιήσεις στο γεγονός ότι ενσωματώνει τρεις βασικές και πρωτότυπες λειτουργίες:

- Επιλογή μεγέθους αεροδρομίου για ενδιάμεσες στάσεις. Δίνεται η επιλογή στο χρήστη να επιλέξει το μέγεθος των πιθανών ενδιάμεσων αεροδρομίων στα οποία θα ψάξει το σύστημα. Με αυτό τον τρόπο βρίσκει και πιθανούς ενδιάμεσους σταθμούς σε μικρά αεροδρόμια στα οποία οι φόροι αεροδρομίων ενδέχεται να είναι μικρότεροι και κατ' επέκταση το εισιτήριο να είναι οικονομικότερο.
- Ορισμός ακτίνας για εύρεση ενδιάμεσων αεροδρομίων. Ο χρήστης έχει τη δυνατότητα επιλογής της ακτίνας στην οποία το σύστημα θα ψάξει για ενδιάμεσους προορισμούς, αυξάνοντας έτσι την πιθανότητα να βρει κάποιο οικονομικότερο εισιτήριο.

- Ανεξαρτητοποίηση από τις αεροπορικές εταιρείες. Τα υφιστάμενα συστήματα, βασισμένα σε συνεργασίες με μεγάλες – κυρίως – αεροπορικές εταιρείες, προωθούν συγκεκριμένα δρομολόγια και αντίστοιχα εισιτήρια, υποβιβάζοντας τις φθηνότερες προσφορές. Οι μεγάλες αεροπορικές εταιρείες συνεργάζονται με ιστότοπους κρατήσεων εισιτηρίων, αποκομίζοντας σημαντικά οφέλη για τις ίδιες. Ο καταναλωτής όμως αποκτά δυσκολότερη πρόσβαση σε πιο φθηνές εναλλακτικές λύσεις, οι οποίες δεν προσφέρονται κατά προτεραιότητα από τους ιστότοπους οι οποίοι συνεργάζονται με τις μεγάλες εταιρείες.

Η δυσκολία που παρουσιάστηκε κατά την υλοποίηση της εφαρμογής αφορούσε την επιλογή του κατάλληλου scraping εργαλείου το οποίο θα αναλάμβανε ασύγχρονα να εξορύξει δεδομένα από τις σελίδες ενδιαφέροντος και να τα αποθηκεύσει στη βάση δεδομένων. Με την χρησιμοποίηση του εργαλείου Puppeteer, η δυσκολία αυτή ξεπεράστηκε γρήγορα και τα αποτελέσματα ήταν άκρως ικανοποιητικά.

Συνεχίζοντας, παρουσιάστηκαν λεπτομερώς οι απαιτήσεις του συστήματος, τα storyboards και paper prototypes των διεπαφών χρήστη, η αξιολόγηση με τη μέθοδο think aloud και έγινε ανάλυση της σχεδίασης και υλοποίησης του συστήματος όπου παρουσιάστηκαν οι τεχνολογίες που χρησιμοποιήθηκαν.

Η λειτουργία του μηχανισμού web scraping λειτουργεί απολύτως αξιόπιστα με τον ιστότοπο για τον οποίο προγραμματίστηκε το λογισμικό. Ο προβληματισμός σε ένα τέτοιου είδους σύστημα έγκειται στις τροποποιήσεις που πρέπει να γίνουν στο λογισμικό web scraping σε περίπτωση που η πλατφόρμα κρατήσεων εισιτηρίων τροποποιήσει τη διάταξη και την HTML σήμανση των σελίδων της.

Εν κατακλείδι, μέσα από τη χρήση του συστήματος, αναπτύχθηκαν προβληματισμοί καθώς και ανάγκες οι οποίες μπορούν να καλυφθούν σε μελλοντικές επεκταμένες εκδόσεις του συστήματος, όπως αναφέρθηκε και στο κεφάλαιο «Μελλοντικές Επεκτάσεις».

10 Βιβλιογραφία

1. Anon., n.d. [Ηλεκτρονικό]
   Available at: https://github.com/GoogleChrome/puppeteer
2. Apress, 2009. Using Web Scraping to Create Semantic Relations. *Scripting Intelligence*, pp. 205-228.
3. Berlind, D., 2015. *APIs Are Like User Interfaces-Just With Different Users in Mind.* [Ηλεκτρονικό]
   Available at: https://www.programmableweb.com/news/api-economy-delivers-limitless-possibilities/analysis/2015/12/03
   [Πρόσβαση 5 2018].
4. Black, M. L., 2016. THE WORLD WIDE WEB AS COMPLEX DATA SET: EXPANDING THE DIGITAL HUMANITIES INTO THE TWENTIETH CENTURY AND BEYOND THROUGH INTERNET RESEARCH. *International Journal of Humanities and Arts Computing*, 10(1), p. 95–109.
5. Boyd, D., Keller, E. F. & Tijerina, B., 2016. Supporting Ethical Data Research: An Exploratory Study of Emerging Issues in Big Data and Technical Research. *Data & Society*.
6. Califf, M. E. & Mooney, R. J., 1999. *Relational Learning of Pattern-Match Rules for Information Extraction.* Orlando, Florida, Proceedings of the Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence.
7. Castrillo-Fernández, O., 2015. *Web Scraping: Applications and Tools*, s.l.: European Public Sector Information Platform Topic Report No. 2015 / 10.
8. Christodoulakis, C., 2012. *A Rich Media Mobile Web Application for Visitord and the Community of the Technical University of Crete.* s.l.:s.n.
9. Crummy, 2004. *Beautiful Soup.* [Ηλεκτρονικό]
   Available at: https://www.crummy.com/software/BeautifulSoup/
10. Dayley, B., 2014. *Node.js, MongoDB and AngulaJS Web Development.* s.l.:Pearson Education.

11. Dmello, B., 2016. *What You Need To Know About Node.js.* Birmingham, UK: Packt Publishing.

12. Draxi, V., 2018. *Web Scraping Data Extraction from Websites.* Wien: s.n.

13. EUCAST, 2017. The European Committee on Antimicrobial Susceptibility Testing - EUCAST. *European Society of Clinical Microbiology and Infectious Diseases*, October.

14. Glez-Peña, D. και συν., 2014. Web scraping technologies in an API world. *Briefings in Bioinformatics*, September, 15(5).

15. Google, n.d. [Ηλεκτρονικό]
Available at: https://github.com/GoogleChrome/puppeteer

16. Hahn, E. M., 2016. *Express in Action.* s.l.:Manning.

17. Hyun, J. L. & Soo , D. K., 2010. *Balanced MVC Architecture for Developing Service-Based Mobile Applications.* Shanghai, China, s.n.

18. Jacarta Commons, 2008. *HTTP components.* [Ηλεκτρονικό]
Available at: http://hc.apache.org/httpclient-3.x/tutorial.html

19. Jones, C. και συν., 2009. *Parallelizing the web browser.* Berkeley, California, USENIX.

20. Kristina Chodorow, M. D., 2010. *MondoDB The Definitive Guide.* s.l.:O'REILLY.

21. Madison, M., Barnhill, M., Napier, C. & Godin, J., 2015. NoSQL Database Technologies. *Journal of International Technology and Information Management*, 24(1).

22. Makris, D.-I. E., 2015. *Design and Implemantation of a Platform for the Development and Management of Learning Experiences in Location-Based Mobile Games.* s.l.:s.n.

23. Mithun Atheesh, B. J. D. J. K., 2015. *Web Development with MongoDB and NodeJS.* s.l.:Packt Publishing.

24. Ornbo, G., 2012. *Sams Teach Yourself Node.js in 24 Hours.* Indianapolis, Indiana: Sams Publishing.

25. Pawel Kozlowski, P. B. D., 2013. *Mastering Web Application Development with AngularJS.* s.l.:Packt Publishing.

26. Richard Blewett, A. C., 2013. *Pro Asynchronous Programming with .NET.* s.l.:Apress.

27. Rocha, W., Fukuda, H. & Leger, P., 2015. *Modular Asynchronous Web Programming: Advantages & Challenges.* s.l., s.n.

28. Rod Johnson, J. H. A. A. T. R. C. S., 2005. *Professional Java Development with the Spring Framework.* Indianapolis: Wiley Publishing.

29. Salerno, J. J. & Boulware, D. M., 2003. *Method and apparatus for improved web scraping*, s.l.: United States of America Patent US 7072890 B2.

30. Sanjay Kumar Malik, S. R., 2011. *Information Extraction using Web Usage Mining, Web Scrapping and Semantic Annotation.* s.l.:s.n.

31. Schaumont, P., 2013. *A Practical Introduction to Hardware/Software Codesign.* New York, USA: Springer.

32. Teixeira, P., 2012. *Professional Node.js: Building Javascript Based Scalable Software.* Indianapolis, Indiana: John Wiley & Sons.

33. Thomas, S., Karnik, S. & Barai, R. S., 2010. CAMP: a useful resource for research on antimicrobial peptides. *Nucleic Acids Res*, Τόμος 38, pp. 774-80.

34. Tilkov, S. & Vinoski, S., 2010. Node.js: Using JavaScript to Build High-Performance Network Programs. *IEEE Internet Computing*, 14(6).

35. W3C, 1999. *XML Path Language (XPath) Version 1.0*, s.l.: s.n.

36. Walls, C., 2015. *Spring in action.* 4th edition επιμ. s.l.:Manning.

37. Wang, Z. & Wang, G., 2004. APD: the antimicrobial peptide database. *Nucleic Acids Res*, Τόμος 32, pp. 590-2.