



ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Recipe Web Scraper

Άννα Α. Πετρίδου

Επιβλέπων: Αλέξανδρος Ντούλας, Επίκουρος Καθηγητής

ΑΘΗΝΑ

ΙΑΝΟΥΑΡΙΟΣ 2023

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Recipe Web Scraper

Άννα Α. Πετρίδου

A.M.: 1115201600135

ΕΠΙΒΛΕΠΟΝΤΕΣ: Αλέξανδρος Ντούλας, Επίκουρος Καθηγητής

ΠΕΡΙΛΗΨΗ

Σκοπός της εργασίας αυτής είναι η δημιουργία μίας εφαρμογής, που θα συλλέγει δεδομένα από πολλές διαφορετικές πηγές και θα τα προσφέρει συλλογικά στο χρήστη με τρόπο και λειτουργικότητες που θα τον διευκολύνουν.

Η εργασία αυτή λοιπόν αφορά μία εφαρμογή η οποία σαρώνει 5 μεγάλους ιστότοπους με συνταγές μαγειρικής, μαζεύει όση πληροφορία χρειάζεται από αυτούς και την περνάει σε μία βάση δεδομένων. Στη συνέχεια, μέσω του backend και του frontend που έχουν δημιουργηθεί, η εφαρμογή προσφέρει στο χρήστη μία σειρά από λειτουργίες και δυνατότητες τις οποίες μπορεί να εκμεταλλευτεί κάνοντας χρήση της διεπαφής χρήστη που έχουμε υλοποιήσει (user interface).

Πιο συγκεκριμένα, αρχικά δημιουργήσαμε τους 5 web crawlers, που ουσιαστικά σαρώνουν τις ιστοσελίδες, φιλτράροντας όλους τους συνδέσμους τους και αποθηκεύοντας μόνο όσους αφορούν συνταγές, σε ένα ξεχωριστό αρχείο κειμένου. Η υλοποίηση των crawlers έγινε με rython χρησιμοποιώντας το “scrapy”.

Έπειτα, οι 5 web scrapers που έχουμε υλοποιήσει σε java, σαρώνουν κάθε σύνδεσμο που διαβάζουν από τα αρχεία κειμένου και αποθηκεύουν τις επιθυμητές πληροφορίες για την κάθε συνταγή, σε μία βάση δεδομένων MySQL. Το scraping των επιθυμητών δεδομένων από κάθε site με java γίνεται με τη βοήθεια του εργαλείου “Jsoup”.

Αφού περαστεί όλη η απαραίτητη πληροφορία στη βάση δεδομένων, το REST API που υλοποιήσαμε σε java με “spring boot”, παρέχει μια σειρά από λειτουργίες αναζήτησης στον χρήστη, που φροντίζει να παραδοθούν στο frontend που υλοποιήσαμε.

Τέλος, η διεπαφή χρήστη, δηλαδή το frontend, για το οποίο χρησιμοποιήσαμε “Vue.js”, φροντίζει τα δεδομένα και οι λειτουργίες της εφαρμογής να παρουσιάζονται με τρόπο εύχρηστο στο χρήστη.

Αποτέλεσμα αυτής της εργασίας είναι, η δημιουργία ενός project, περνώντας από όλες τις φάσεις του development, βλέποντας και μαθαίνοντας την πορεία για έναν πλήρη κύκλο σχεδιασμού και υλοποίησης μιας εφαρμογής, από τον οποίο προκύπτει μία εφαρμογή λειτουργική αλλά και χρήσιμη.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Απόξεση – Σάρωση Ιστού

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: εξόρυξη, σάρωση δεδομένων, διεπαφή χρήστη, web crawling, web scraping, backend, frontend

ABSTRACT

The purpose of this project is to create an application, that will collect data from many different sources and offer them collectively to the user in such a way and with functionalities that will facilitate him.

So, this work concerns an application that scans 5 large websites with cooking recipes, collects as much information as it needs from them and passes it to a database. Then, through the backend and frontend that have been created, the application offers the users a series of functions and features that they can take advantage of using the user interface that we have implemented.

More specifically, we initially created the 5 web crawlers, which basically scan the websites, filtering all their links and saving only those related to recipes, in a separate text file. The implementation of the crawlers was done with python using "scrapy".

Then, the 5 web scrapers that we have implemented in java, scan each link they read from the text files and store the desired information for each recipe, in a MySQL database. The scraping of the desired data from each java site is done with the help of the "Jsoup" tool.

After passing all the necessary information to the database, the REST API that we implemented in java with "spring boot", provides a series of search functions to the user, which takes care of being delivered to the frontend that we implemented.

Finally, the user interface, i.e., the frontend, for which we used "Vue.js", makes sure that the data and functions of the application are presented in a user-friendly way.

The result of this work is the creation of a project, going through all the phases of development, seeing, and learning the course for a complete cycle of planning and implementing an application, from which a functional but also useful application results.

SUBJECT AREA: Web Scraping

KEYWORDS: scraping, crawling, backend, frontend, user-interface

Στην οικογένεια και ιδιαίτερα στον παππού μου, Γεώργιο Τσαγκαλάκη.

ΕΥΧΑΡΙΣΤΙΕΣ

Για τη διεκπεραίωση της παρούσας πτυχιακής εργασίας, θα ήθελα να ευχαριστήσω τον επιβλέποντα, επίκουρο καθηγητή Αλέξανδρο Ντούλα, για την αρχική ιδέα, την καθοδήγησή, μα και συνολικά για την πολύτιμη συνεισφορά του στην ολοκλήρωση της.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ	14
1. ΕΙΣΑΓΩΓΗ.....	15
1.1 Γενική Περιγραφή Έργου	15
1.2 Γενικές Απαιτήσεις	15
1.3 Ροή Πληροφορίας.....	20
2. ΕΡΓΑΛΕΙΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ	21
2.1 Διαθέσιμα εργαλεία και τεχνολογίες.....	21
2.1.1 Βάση Δεδομένων	21
2.1.2 Scrapers	22
2.1.3 Crawlers	23
2.1.4 Rest API	23
2.1.5 User Interface.....	24
2.2 Εργαλεία και τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση και οι λόγοι που επιλέχθηκαν	24
2.2.1 Βάση Δεδομένων	24
2.2.2 Scrapers	28
2.2.3 Crawlers	29
2.2.4 Rest API	30
2.2.5 User Interface.....	30
3. ΜΕΘΟΔΟΛΟΓΙΑ ΥΛΟΠΟΙΗΣΗΣ.....	31
3.1 Backend	31
3.1.1 Sequence Diagram	32
3.1.2 Περιγραφή αρχιτεκτονικής spring-boot.....	33
3.2 Frontend	37
3.3 Testing	38
4. USER MANUAL – APPLICATION DEMO	39

4.1	Παρουσίαση Εκτέλεσης	39
4.2	Πίνακες με χρήσιμα στατιστικά και πληροφορίες.....	86
5.	ΣΥΜΠΕΡΑΣΜΑΤΑ	87
5.1	Γενικά Συμπεράσματα.....	87
5.2	Δυνατότητες Επέκτασης.....	87
	ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ	88
	ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ	89
	ΑΝΑΦΟΡΕΣ	90

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Εικόνα 1: Χρόνος εκτέλεσης από τον ιστότοπο "simplyrecipes.com"	17
Εικόνα 2: Χρόνος εκτέλεσης από τον ιστότοπο "allrecipes.com"	17
Εικόνα 3: Αναζήτηση με βάση το πλήθος των συστατικών	34
Εικόνα 4: Αναζήτηση με βάση τον τίτλο και τον συγγραφέα	35
Εικόνα 5: Αναζήτηση με βάση τον χρόνο της συνταγής	35
Εικόνα 6: Παράδειγμα πιο σύνθετης αναζήτησης, με βάση τις θερμίδες, την κατηγορία και τη δυσκολία	36
Εικόνα 7: Εκτέλεση του script για το crawling του bbcgoodfood.com	39
Εικόνα 8: Κατά τη διάρκεια του crawling του bbcgoodfood.com	40
Εικόνα 9: Η εικόνα στο terminal μετά την ολοκλήρωση του crawling	40
Εικόνα 10: : Η τελική εικόνα στο terminal μετά την ολοκλήρωση του crawling	41
Εικόνα 11: Τα links που ανακτήθηκαν για το bbcgoodfood.com	41
Εικόνα 12: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 15010 σύνδεσμοι	42
Εικόνα 13: Εκτέλεση του script για το crawling του allrecipes.com	42
Εικόνα 14: Εκτέλεση του script για το crawling του cookingclassy.com	43
Εικόνα 15: Εκτέλεση του script για το crawling του simplyrecipes.com	43
Εικόνα 16: Εκτέλεση του script για το crawling του tasty.co	43
Εικόνα 17: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 13613 σύνδεσμοι	44
Εικόνα 18: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 1713 σύνδεσμοι	45
Εικόνα 19: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 2996 σύνδεσμοι	46
Εικόνα 20: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 2642 σύνδεσμοι	47
Εικόνα 21: Database model μέσα στο MySQL Workbench	48
Εικόνα 22: Κλικ στο "Forward Engineer"	49
Εικόνα 23: Κλικ στο "Continue"	49
Εικόνα 24: Κλικ στο "continue"	50
Εικόνα 25: Κλικ στο "continue"	51

Εικόνα 26: Κλικ στο “continue”	52
Εικόνα 27: Κλικ στο “close”	53
Εικόνα 28: Ο άδειος πίνακας που επιστρέφει το query.....	54
Εικόνα 29: Ο κώδικας της main.java.....	55
Εικόνα 30: Κατά τη διάρκεια του scraping	56
Εικόνα 31: Ο πίνακας “recipe” μετά το scraping	57
Εικόνα 32: Ο πίνακας “category”	58
Εικόνα 33: Ο πίνακας “difficulty”	58
Εικόνα 34: Ο πίνακας “ingredient”	59
Εικόνα 35: Ο πίνακας “ingredient_group”	60
Εικόνα 36: Ο πίνακας “ingredient_group_has_ingredient”.....	61
Εικόνα 37: Ο πίνακας “nutrition_info”	62
Εικόνα 38: Ο πίνακας “recipe”	62
Εικόνα 39: Ο πίνακας “scraper”	63
Εικόνα 40: Ο πίνακας “step”	64
Εικόνα 41: Το αρχείο κώδικα “RecipesApplication.java”	64
Εικόνα 42: Το terminal όταν εκκινείται το spring boot	65
Εικόνα 43: Make serve στο visual studio code	66
Εικόνα 44: Η homepage σελίδα του user interface.....	67
Εικόνα 45: Η αρχική σελίδα μετά από scroll down.....	67
Εικόνα 46: Αποτέλεσμα ενδεικτικής συνταγής	68
Εικόνα 47: Σελίδα συνταγής με όλη τη σχετική πληροφορία	69
Εικόνα 48: Σελίδα συνταγής μετά από scroll down	70
Εικόνα 49: Σελίδα συνταγής μετά από περισσότερο scroll down.....	70
Εικόνα 50: Το tab του “nutritional info”	71
Εικόνα 51: Κουμπί αναζήτησης στην αρχική σελίδα	71

Εικόνα 52: Σελίδα με φίλτρα	72
Εικόνα 53: Αποτελέσματα φίλτρων	72
Εικόνα 54: Αποτελέσματα μετά από περισσότερα φίλτρα	73
Εικόνα 55: Τρόπος αναζήτησης από το μενού στα αριστερά	73
Εικόνα 56: Κλικ στο “recipes” και “search recipes”	74
Εικόνα 57: Τα πεδία των φίλτρων	74
Εικόνα 58: Search examples στο postman	75
Εικόνα 59: Παράδειγμα αναζήτησης με περιορισμούς στο postman	76
Εικόνα 60: Τα αρχεία των crawlers	76
Εικόνα 61: Το αρχείο του crawler για το allrecipes.com	77
Εικόνα 62: Το αρχείο του crawler για το cookingclassy.com	78
Εικόνα 63: Η main των scrapers	79
Εικόνα 64: Συνέχεια της main	79
Εικόνα 65: Συνέχεια της main	80
Εικόνα 66: Backend - Main	80
Εικόνα 67: Αρχεία του backend	81
Εικόνα 68: Κώδικας controller για category	82
Εικόνα 69: Κώδικας DAO για category	83
Εικόνα 70: Κώδικας DAO για recipe	84
Εικόνα 71: Κώδικας mapper για category	85

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Σχήμα 1: Το γενικό σχήμα ροής της πληροφορίας	20
Σχήμα 2: Σχήμα Βάσης Δεδομένων.....	25
Σχήμα 3: Sequence Diagram.....	32

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πίνακας 1: Links που εισήχθησαν στη βάση δεδομένων	86
Πίνακας 2: Μνήμη που απαιτήθηκε για κάθε αρχείο txt που περιλαμβάνει τα links με τις συνταγές του κάθε σάιτ.....	86
Πίνακας 3: Μνήμη που απαιτήθηκε για τη βάση δεδομένων	86
Πίνακας 4: Μέσος χρόνος scraping και crawling.....	86

ΠΡΟΛΟΓΟΣ

Η πτυχιακή εργασία πραγματοποιήθηκε στην Αθήνα το ακαδημαϊκό έτος 2022 και έχει στόχο τον σχεδιασμό και την υλοποίηση μίας εφαρμογής που θα ανακτά πληροφορία μέσω σάρωσης ιστού και θα παρέχει στο χρήστη μία διεπαφή με μία σειρά από δυνατότητες. Ευχαριστώ θερμά τον κύριο Α. Ντούλα για τη βοήθεια και καθοδήγησή του καθόλη τη διάρκεια εκπόνησης της εργασίας.

1. ΕΙΣΑΓΩΓΗ

1.1 Γενική Περιγραφή Έργου

Σήμερα υπάρχουν άπειρες εφαρμογές στο διαδίκτυο, δεδομένου ότι ο καθένας πλέον μπορεί να φτιάξει, γεγονός το οποίο έχει αποφέρει τον κατακερματισμό της πληροφορίας.

Ένα από τα συνήθη προβλήματα με τα οποία έρχεται αντιμέτωπος κάθε χρήστης, είναι η ανάκτηση πληροφορίας όταν αυτή είναι διαμοιρασμένη σε πολλές διαφορετικές πηγές. Υπάρχουν τόσες πολλές εφαρμογές και ιστοσελίδες με διαφορετική πληροφορία, που είναι αρκετά δύσκολο να αναζητήσουμε αυτό που θέλουμε, καθώς πρέπει να ψάξουμε σε κάθε πηγή ξεχωριστά.

Για παράδειγμα, όσον αφορά τις συνταγές μαγειρικής, υπάρχουν πάρα πολλά sites(ιστοσελίδες) με διαφορετικό περιεχόμενο και ο χρήστης για να μπορέσει να αναζητήσει μέσα στην πληροφορία των μεγάλων sites, θα πρέπει να μπει να ψάξει σε καθένα ξεχωριστά.

Για να επιλύσουμε το πρόβλημα αυτό, θέλαμε να δημιουργήσουμε μία εφαρμογή η οποία συγκεντρώνει πληροφορία από διαφορετικές πηγές (δηλαδή από διαφορετικές βάσεις δεδομένων) με αυτοματοποιημένο τρόπο, την επεξεργάζεται και προσφέρει στο χρήστη, μέσω μίας εύχρηστης διεπαφής, μια σειρά από δυνατότητες.

Ειδικότερα, αυτό που έχουμε υλοποιήσει ανακτά πληροφορία από διαφορετικές βάσεις δεδομένων και αφού την ενοποιήσει, την εισάγει σε μία δική μας βάση δεδομένων. Στη συνέχεια, έχουμε μία διεπαφή χρήστη σε μορφή ιστοσελίδας, που προσφέρει στο χρήστη τη δυνατότητα να αναζητά συνταγές με βάση πολλά διαφορετικά φίλτρα όπως π.χ. αριθμός θερμίδων, συστατικά, πλήθος υλικών κτλ. με τα οποία ο χρήστης μπορεί να κάνει την αναζήτησή του πιο στοχευμένη.

1.2 Γενικές Απαιτήσεις

Αρχικά, η επιλογή των κατάλληλων εργαλείων και τεχνολογιών αποτέλεσε σε ένα βαθμό δυσκολία, καθώς χρειάστηκε χρόνος και έρευνα για να αναζητήσουμε, να συγκρίνουμε και να αποφασίσουμε τι ταιριάζει καλύτερα στις ανάγκες μας.

Σχετικά με την υλοποίηση της εφαρμογής, η διαδικασία αποτέλεσε πρόκληση, διότι δεν έχουμε προσπέλαση στις βάσεις δεδομένων τις οποίες θέλουμε να ενοποιήσουμε. Αν είχαμε προσπέλαση στις ίδιες τις βάσεις των 5 ιστοσελίδων που επιλέξαμε, θα ήταν αρκετά απλό να μεταφέρουμε πληροφορία από τη μία βάση στην άλλη.

Ωστόσο, εμείς έπρεπε να «σαρώσουμε» την πληροφορία, για το οποίο αποφασίσαμε να χρησιμοποιήσουμε scraping [1], το οποίο έχει τις δικές του δυσκολίες καθώς το ίντερνετ είναι δυναμικό. Η δυσκολία στην ανάκτηση της πληροφορίας έγκειται στις διαφορές που έχουν τα sites μεταξύ τους τόσο στο περιεχόμενο όσο και στην οργάνωση της πληροφορίας τους. Συνεπώς, εμείς έπρεπε να λάβουμε την πληροφορία από τα sites αυτά που έχουν διαφορετικό σκελετό και να την ενοποιήσουμε σε ένα δικό μας κοινό σχήμα. Κληθήκαμε δηλαδή να παράγουμε ένα κοινό σχήμα για βάσεις δεδομένων των οποίων δε γνωρίζουμε το δικό τους.

Ακόμη και στην περίπτωση που γνωρίζαμε την εικόνα των βάσεων αυτών, η υλοποίηση θα μπορούσε να αποτελέσει πρόκληση, πόσο μάλλον τώρα που όλη την πληροφορία την λαμβάνουμε από τη διεπαφή τους.

Συνεπώς, χρειάστηκε ο σχεδιασμός της δίκης μας βάσης να είναι αρκετά ευέλικτος, ώστε το σχήμα της να ικανοποιεί όλους τους περιορισμούς και να μπορεί να αποθηκεύσει την πληροφορία από όλα τα αλλά sites.

Η υλοποίηση μας προσφέρει τη δυνατότητα ενημέρωσης της βάσης δεδομένων αυτόματα, αφού το crawling(ανίχνευση) [2] και αντιστοίχως το scraping(σάρωση) μπορούν να εκτελούνται συνέχεια στο παρασκήνιο και να ενημερώνουν διαρκώς τη βάση.

Επόμενη πρόκληση αποτέλεσε, τόσο η δημιουργία των scrapers(σαρωτές) όσο και του backend(πίσω μέρος μιας εφαρμογής). Πιο συγκεκριμένα, αφού αποφασίσαμε να αντλήσουμε δεδομένα για τις συνταγές από 5 διαφορετικούς ιστότοπους, έπρεπε να φτιάξουμε και 5 διαφορετικούς scrapers. Τα sites αυτά περιέχουν από εκατοντάδες έως και χιλιάδες συνταγές το καθένα, τις οποίες εμείς θέλαμε να συλλέξουμε με έναν ομοιόμορφο τρόπο και να τις αποθηκεύσουμε στη βάση. Μία από τις βασικές προκλήσεις ήταν η επιτυχής ανάκτηση των δεδομένων μέσω των scrapers, καθώς όλα τα πεδία και ο κώδικας των sites είναι διαφορετικός.

Για παράδειγμα, όπως βλέπουμε στις παρακάτω εικόνες, ο χρόνος μαγειρέματος απεικονίζεται διαφορετικά στους 2 ιστότοπους (στον έναν σε λεπτά στον άλλον σε ώρες).

Chicken and Rice Casserole



PREP TIME 25 mins **COOK TIME 70 mins** TOTAL TIME 95 mins

SERVINGS
6 servings

NOTE: This recipe assumes the rice requires about 1 1/2 cups of liquid per cup of rice to cook. Some rice varieties, such as brown rice, require more liquid (and a longer cooking time). Adjust recipe accordingly.

If you are avoiding cooking with alcohol, skip the sherry and deglaze the pan

Εικόνα 1: Χρόνος εκτέλεσης από τον ιστότοπο "simplyrecipes.com"

Εικόνα 2: Χρόνος εκτέλεσης από τον ιστότοπο "allrecipes.com"

Συνεπώς εμείς έπρεπε τόσο να καταφέρουμε να ανακτήσουμε και να αναπαραστήσουμε σωστά τα δεδομένα, παρόλο που ήταν διαφορετικά οργανωμένα

αλλά και να κάνουμε τις κατάλληλες μετατροπές πριν την εισαγωγή τους στη βάση δεδομένων.

Επίσης, έπρεπε να δημιουργήσουμε κατάλληλους αλγοριθμικούς μηχανισμούς, ώστε να αναγνωρίζουμε από ποιες συνταγές ενδεχομένως λείπουν (από το συγγραφέα) βασικές πληροφορίες που όμως περιέχονται σε άλλες. Για να μη χαλάσει η ομοιομορφία λοιπόν και για να λειτουργήσει σωστά μετέπειτα η αναζήτησή στο backend, έπρεπε να σκεφτούμε κριτήρια και μηχανισμούς με τους οποίους θα παραλείπονται αυτές οι συνταγές “εξαιρέσεις” τελείως από τη διαδικασία. Αυτό απαιτούσε ειδικούς τρόπους διαχωρισμού και επισήμανσης των σελίδων αυτών ως μη κατάλληλες.

Όσον αφορά τον crawler(ανιχνευτή), έπρεπε να σχεδιαστεί μία λύση η οποία θα σαρώνει τις ιστοσελίδες και θα βρίσκει όλους τους συνδέσμους που περιέχονται σε αυτή, εξαιρώντας σωστά όσους είναι εξωτερικοί(σύνδεσμοι προς άλλα site), αλλά και των εσωτερικών συνδέσμων που όμως, δεν αφορούν συνταγές. Όλους αυτούς του συνδέσμους τους εξάγουμε σε ένα αρχείο κειμένου, το οποίο δίνουμε αργότερα ως input(εισαγόμενα δεδομένα) στους scrapers για να σαρώσουν τις σελίδες.

Αρκετές απαιτήσεις υπήρχαν φυσικά και στο σωστό σχεδιασμό της βάσης δεδομένων, καθώς εκεί θα βασίζονταν αργότερα όλες οι αναζητήσεις, αλλά και στη δημιουργία των φίλτρων αναζήτησης, ώστε να επιστρέφουν καλά αποτελέσματα σε ό,τι αναζητήσεις μπορεί να πραγματοποιήσει ο χρήστης.

Ακόμη, ήταν αρκετά περίπλοκη η υλοποίηση του user interface(διεπαφή χρήστη) σε συνδυασμό με την υλοποίηση του backend. Έπρεπε να υλοποιηθούν όλα τα features(δυνατότητες) ξεχωριστά και έπειτα να παρουσιαστούν τα αποτελέσματα με τον κατάλληλο τρόπο, μέσα από τη διεπαφή.

Τέλος, το project(έργο) αυτό εμπλέκει τεχνολογίες που είναι end-to-end(από άκρη σε άκρη). Χρησιμοποιήσαμε διαφορετικές τεχνολογίες για το crawling, το scraping, το frontend(μπροστά μέρος μιας εφαρμογής), το backend, τη βάση δεδομένων κτλ, τις οποίες μετά έπρεπε να συντονίσουμε και να ενώσουμε.

Το project αυτό επιλέχθηκε και υλοποιήθηκε για να δώσει αξία στο χρήστη, να τον διευκολύνει και να επιταχύνει τον τρόπο με τον οποίο πραγματοποιεί τις αναζητήσεις του. Δημιουργώντας κάτι χρήσιμο, και εύχρηστο ταυτόχρονα, για τον άνθρωπο, μάθαμε να υλοποιούμε scrapers και crawlers, που χρησιμοποιούνται πολύ σε όλες τις

σύγχρονες εφαρμογές/ μηχανές αναζήτησης. Επιπλέον, ο συνδυασμός αυτών των τεχνολογιών μαζί με τη διεπαφή χρήστη, παρέχει μία πλήρως λειτουργική εφαρμογή που μπορεί να χρησιμοποιήσει οποιοσδήποτε θέλει να βρει κάτι γρήγορα και στοχευμένα ανάμεσα σε χιλιάδες συνταγές.

Η δημιουργία και χρήση της βάσης δεδομένων και του REST API, είναι γνώσεις υψίστης σημασίας και μέσα από το project αυτό μας δόθηκε η ευκαιρία να τις ενισχύσουμε.

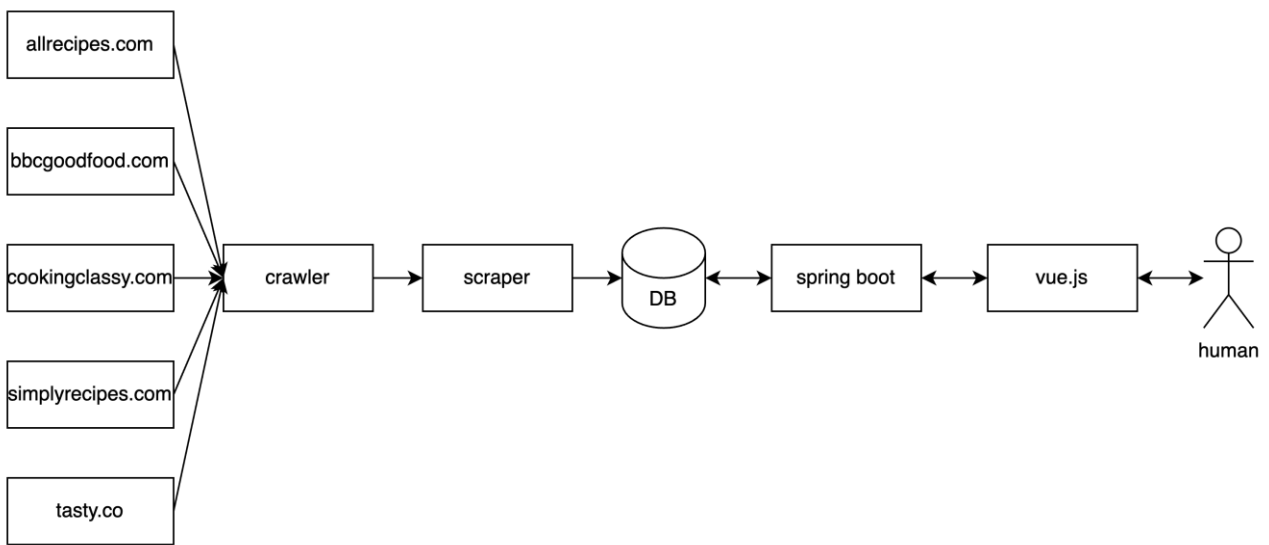
Αρχικά χρειάστηκε προετοιμασία και αναζήτησή για το ποιες τεχνολογίες και εργαλεία υπάρχουν διαθέσιμα, να τα δοκιμάσουμε και να τα συγκρίνουμε μεταξύ τους. Χρειάστηκαν γνώσεις HTML και κατανόηση του πώς κάνουμε extract τα data(απόσπαση δεδομένων) από τις σελίδες, χρειάστηκαν κάποιες βασικές γνώσεις σχεδιασμού βάσεων δεδομένων και φυσικά κατανόηση των REST αρχιτεκτονικών. Έπρεπε να συγκρίνουμε ποια εργαλεία ήταν διαθέσιμα για να υλοποιήσουμε ένα REST backend (e.g. Laravel, Django etc), από τα οποία τελικά εμείς καταλήξαμε στο “Spring Boot”. Τέλος χρειάστηκε μελέτη και κατανόηση των frontend τεχνολογιών και να σχεδιάσουμε το user interface. Συνολικά για την ολοκλήρωση του project, χρειαστήκαμε γνώσεις γύρω από scrapers, crawlers, REST API [3], frontend [4] και άλλες τεχνολογίες. Επιπλέον χρειαστήκαμε μία βάση δεδομένων, για τα 5 μεγάλα web sites στα οποία βασιστήκαμε και σαρώσαμε δεδομένα. Ακόμη, χρειαστήκαμε ένα template site(πρότυπο ιστοσελίδας) το οποίο χρησιμοποιήσαμε ως βάση για τη δημιουργία της διεπαφής χρήστη.

Τα 5 sites από τα οποία ανακτούμε συνταγές είναι τα παρακάτω:

- allrecipes.com
- bbcgoodfood.com
- tasty.co
- cookingclassy.com
- simplyrecipes.com

1.3 Ροή Πληροφορίας

Το γενικό σχήμα ροής της πληροφορίας είναι το παρακάτω:



Σχήμα 1: Το γενικό σχήμα ροής της πληροφορίας

2. ΕΡΓΑΛΕΙΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ

2.1 Διαθέσιμα εργαλεία και τεχνολογίες

2.1.1 Βάση Δεδομένων

Υπάρχουν διάφοροι τύποι βάσεων δεδομένων και το ποιος είναι ο καταλληλότερος εξαρτάται κάθε φορά από τις απαιτήσεις και ανάγκες του κάθε project.

Ενδεικτικά υπάρχουν οι εξής:

1. Αντικειμενοστραφής βάση δεδομένων: Σε μια αντικειμενοστραφή βάση δεδομένων, οι πληροφορίες αποθηκεύονται με τρόπο όμοιο με αντικείμενα.
2. Σχεσιακή βάση δεδομένων [5]: Μια σχεσιακή βάση δεδομένων είναι βασισμένη σε πίνακες όπου κάθε bit δεδομένων έχει μια σύνδεση με κάθε άλλο bit δεδομένων.
3. Μη σχεσιακή ή NoSQL βάση δεδομένων [6]: Μια βάση δεδομένων χωρίς SQL χρησιμοποιεί μια ποικιλία μορφών, όπως έγγραφα, γραφήματα, μεγάλες στήλες κτλ, γεγονός που προσφέρει μεγάλη ευελιξία και επεκτασιμότητα σε ένα σχεδιασμό βάσης δεδομένων.

Παρόλα αυτά οι βάσεις δεδομένων είναι ευρέως διαχωρισμένες σε δύο μεγάλες κατηγορίες, τις Σχεσιακές Βάσεις Δεδομένων (Relational or Sequence Databases) και τις Μη Σχεσιακές Βάσεις Δεδομένων (Non-relational/Non-sequence databases/No SQL databases). Οποιοσδήποτε μπορεί να τις χρησιμοποιήσει ξεχωριστά και σε συνδυασμό μεταξύ τους, ανάλογα από τη φύση των δεδομένων και τη λειτουργικότητα που απαιτείται.

Οι σχεσιακές βάσεις δεδομένων είναι ο πιο συνηθισμένος τύπος βάσης. Χρησιμοποιεί ένα σχήμα(schema) το οποίο είναι το πρότυπο(template) που χρησιμοποιείται για να καθορίσει τη δομή των δεδομένων που θα αποθηκευτούν στη βάση.

2.1.2 Scrapers

Για τη δημιουργία των scrapers υπάρχουν πολλά διαθέσιμα εργαλεία. Τα πιο γνωστά είναι:

- το Scrapy: Το Scrapy είναι ένα δωρεάν και ανοιχτού κώδικα framework(δομή) ανίχνευσης ιστού (web-crawling) γραμμένο σε Python. Αρχικά σχεδιασμένο για web scraping, μπορεί επίσης να χρησιμοποιηθεί για εξαγωγή δεδομένων χρησιμοποιώντας API ή ως web crawler γενικής χρήσης.
- το Octoparse: Το Octoparse είναι μια cloud-based(βασισμένη στο «σύννεφο») λύση εξαγωγής δεδομένων ιστού που βοηθά τους χρήστες να εξάγουν σχετικές πληροφορίες από διάφορους τύπους ιστότοπων. Επιτρέπει σε χρήστες να ανακτούν μη δομημένα δεδομένα και να τα αποθηκεύουν σε διαφορετικές μορφές, όπως Excel, απλό κείμενο και HTML.
- το BeautifulSoup: Το BeautifulSoup είναι ένα πακέτο Python για την ανάλυση(parsing) εγγράφων HTML και XML (συμπεριλαμβανομένης της ύπαρξης λανθασμένης σήμανσης, δηλαδή μη κλειστών ετικετών). Δημιουργεί ένα δέντρο ανάλυσης(parse tree) για αναλυμένες σελίδες που μπορεί να χρησιμοποιηθεί για την εξαγωγή δεδομένων από HTML, που είναι χρήσιμο για το web scraping.
- το Selenium: Το Selenium είναι ένα έργο “ομπρέλα” ανοιχτού κώδικα για μια σειρά εργαλείων και βιβλιοθηκών που στοχεύουν στην υποστήριξη της αυτοματοποίησης του προγράμματος περιήγησης. Παρέχει ένα εργαλείο αναπαραγωγής για τη σύνταξη λειτουργικών δοκιμών στα περισσότερα σύγχρονα προγράμματα περιήγησης ιστού, χωρίς την ανάγκη εκμάθησης μιας δοκιμαστικής γλώσσας δέσμης ενεργειών (Selenium IDE).
- και το Jsoup: Το jsoup είναι μια βιβλιοθήκη Java ανοιχτού κώδικα που έχει σχεδιαστεί για την ανάλυση, εξαγωγή και χειρισμό δεδομένων που είναι αποθηκευμένα σε έγγραφα HTML.

2.1.3 Crawlers

Για τη δημιουργία των crawlers υπάρχουν πολλά διαθέσιμα εργαλεία.

Τα πιο γνωστά είναι:

- το Scrapy [7]: όπως περιγράφηκε παραπάνω.
- το Apache Nutch [8]: Το Apache Nutch είναι ένα εξαιρετικά μεγάλο, αναλυτικό και επεκτάσιμο έργο λογισμικού ανιχνευτή ιστού(web crawler) ανοιχτού κώδικα.
- το Heritrix [9]: Το Heritrix είναι ένας ανιχνευτής ιστού(web crawler) σχεδιασμένος για αρχειοθέτηση ιστού(web archiving). Γράφτηκε από το Internet Archive. Είναι διαθέσιμο με άδεια ελεύθερου λογισμικού και γραμμένο σε Java.

2.1.4 Rest API

Για τη δημιουργία του REST API υπάρχουν πολλά διαθέσιμα εργαλεία. Τα πιο γνωστά είναι:

- Spring Boot(Java) [10]: Το Spring Boot είναι ένα framework ανοιχτού κώδικα που βασίζεται σε Java που χρησιμοποιείται για τη δημιουργία ενός micro Service(μικρο υπηρεσίας api). Αναπτύχθηκε από την Pivotal Team και χρησιμοποιείται για την κατασκευή αυτόνομων και έτοιμων για παραγωγή spring εφαρμογών.
- Laravel [11]: Το Laravel είναι ένα δωρεάν PHP web framework και ανοιχτού κώδικα, που δημιουργήθηκε από τον Taylor Otwell και προορίζεται για την ανάπτυξη διαδικτυακών εφαρμογών σύμφωνα με το αρχιτεκτονικό μοτίβο model-view-controller (MVC).
- Django(Python) [12]: Το Django είναι ένα δωρεάν και ανοιχτού κώδικα web framework βασισμένο σε Python που ακολουθεί το αρχιτεκτονικό μοτίβο model-template-views (MTV).

2.1.5 User Interface

Για τη δημιουργία των user interface υπάρχουν πολλά διαθέσιμα εργαλεία. Τα πιο γνωστά είναι:

- React [13]: Η React (γνωστή και ως React.js ή ReactJS) είναι μια δωρεάν και ανοιχτού κώδικα βιβλιοθήκη JavaScript front-end για τη δημιουργία διεπαφών χρήστη με βάση UI components(συστατικά). Συντηρείται από τη Meta (πρώην Facebook) και μια κοινότητα μεμονωμένων προγραμματιστών και εταιρειών.
- Vue.js [14]: Η Vue.js (κοινώς αναφέρεται ως Vue, προφέρεται "προβολή") είναι ένα JavaScript framework ανοιχτού κώδικα για τη δημιουργία διεπαφών χρήστη και εφαρμογών μιας σελίδας.
- Angular [15]: Η Angular (αναφέρεται επίσης ως "Angular 2+") είναι ένα web application framework δωρεάν και ανοιχτού κώδικα που βασίζεται σε TypeScript, το οποίο διευθύνεται από την Angular Team της Google και από μια κοινότητα ατόμων και εταιρειών.

2.2 Εργαλεία και τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση και οι λόγοι που επιλέχθηκαν

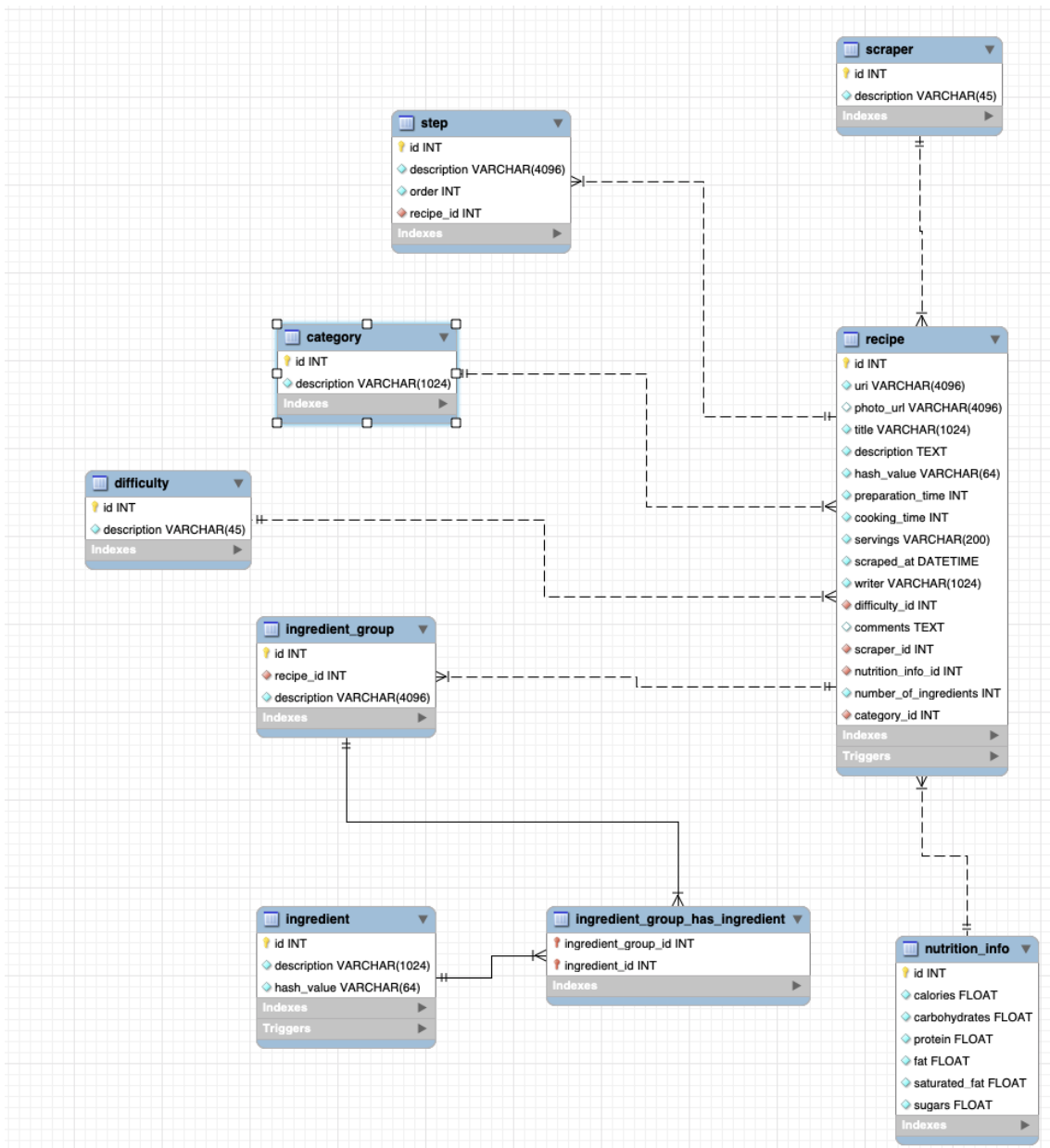
2.2.1 Βάση Δεδομένων

Όσον αφορά τη βάση δεδομένων, η επιλογή ήταν ανάμεσα σε MySQL[16] και PostgreSQL[17] διότι αυτές ήταν δωρεάν και είχαμε εξοικειωθεί με τη χρήση των sql βάσεων μέσα από μαθήματα του τμήματος. Στο συγκεκριμένο project θεωρήσαμε πως δε θα είχαν μεγάλη διαφορά και πως ήταν και οι δύο εξίσου κατάλληλες. Οι βάσεις αυτές είναι και οι δύο δωρεάν και δεν υπήρχε λόγος στο πλαίσιο αυτής της υλοποίησης να χρησιμοποιηθεί κάποια άλλη, καθώς αυτές υπερκαλύπτουν τις ανάγκες μας.

Ένας από τους λόγους που προτιμήσαμε τη MySQL, είναι πως έχει ευκολότερο deployment(ανάπτυξη). Πιο συγκεκριμένα, ήταν πολύ εύκολο να βρούμε δωρεάν δοκιμαστικούς servers(διακομιστές) στο διαδίκτυο και είχαμε και ήδη εμπειρία μέσα από τα μαθήματα του τμήματος, τόσο στην ίδια τη MySQL όσο και στο εργαλείο "MySQL Workbench" που χρησιμοποιήσαμε. Επομένως, επωφεληθήκαμε των γνώσεων που είχαμε ήδη για να προχωρήσει ομαλότερα και γρηγορότερα η ανάπτυξη της εφαρμογής.

Το “MySQL Workbench” [18], είναι ένα πολύ βολικό και εύχρηστο εργαλείο που έχουμε δει μέσα από διάφορα μαθήματα της σχολής και θα ήταν κρίμα να μην το εκμεταλλευτούμε. Επομένως, η επιλογή έγινε για πρακτικούς λόγους, παρόλο που τα εργαλεία ήταν τελείως ισοδύναμα μπροστά σε αυτό που θέλαμε να υλοποιήσουμε.

Η εικόνα της βάσης δεδομένων σε διάγραμμα είναι όπως φαίνεται παρακάτω:



Σχήμα 2: Σχήμα Βάσης Δεδομένων

Πιο συγκεκριμένα, η βάση περιέχει τους εξής πίνακες:

recipe:

Περιλαμβάνει μια σειρά από δεδομένα της κάθε συνταγής και συνδέεται με σχέσεις με όλους τους άλλους πίνακες εκτός του ingredient και ingredient_group_has_ingredient, με τα οποία συνδέεται έμμεσα μέσω του ingredient_group. Με τους πίνακες step και ingredient_group συνδέεται με σχέση 1-N (δηλαδή μία συνταγή μπορεί να έχει πολλά βήματα/step και πολλές ομάδες συστατικών/ingredient_group). Με τους υπόλοιπους πίνακες συνδέεται με σχέση N-1, δηλαδή η καταχώριση κάθε πίνακα(πχ στον difficulty, scraper κτλ) μπορεί να αντιστοιχεί σε πάνω από μία συνταγή και πρωτεύον κλειδί είναι το id.

scraper:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment) και μία περιγραφή(VARCHAR4096), που περιλαμβάνει το όνομα του scraper που χρησιμοποιήθηκε για τη σάρωση και εισαγωγή στη βάση της συγκεκριμένης συνταγής. Κανένα από τα δύο πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id.

step:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment), μία περιγραφή(VARCHAR45) που αποτελεί το κείμενο του βήματος, ένα order(ακέραιος αριθμός) που δηλώνει τη σειρά του βήματος μέσα στη συνταγή και ένα recipe_id(ακέραιος αριθμός) που δηλώνει τη συνταγή στην οποία αντιστοιχεί το βήμα. Κανένα από τα πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id. Η περιγραφή της κατηγορίας(description) είναι υποχρεωτικά μοναδική (unique).

category:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment) και μία περιγραφή(VARCHAR1024), που περιλαμβάνει το όνομα της κατηγορίας της συγκεκριμένης συνταγής. Κανένα από τα δύο πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id. Η περιγραφή της κατηγορίας(description) είναι υποχρεωτικά μοναδική (unique).

difficulty:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment) και μία περιγραφή(VARCHAR45), που περιλαμβάνει το όνομα της δυσκολίας της συγκεκριμένης συνταγής. Κανένα από τα δύο πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id. Η περιγραφή της δυσκολίας(description) είναι υποχρεωτικά μοναδική (unique).

ingredient_group:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment), μία περιγραφή(VARCHAR4096) που αποτελεί το όνομα της ομάδας συστατικών και ένα recipe_id(ακέραιος αριθμός) που δηλώνει τη συνταγή στην οποία αντιστοιχεί το βήμα. Κανένα από τα πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id. Η περιγραφή της κατηγορίας(description) είναι υποχρεωτικά μοναδική (unique).

ingredient:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment), μία περιγραφή(VARCHAR1024) που αποτελεί το όνομα του συστατικού και ένα hash_value(VARCHAR64) που δηλώνει τη συνταγή στην οποία αντιστοιχεί το βήμα. Κανένα από τα πεδία δε μπορεί να είναι κενό(null) και πρωτεύον κλειδί είναι το id. Η περιγραφή της κατηγορίας(description) είναι υποχρεωτικά μοναδική (unique).

ingredient_group_has_ingredient:

Περιλαμβάνει ένα ingredient_group_id(ακέραιος αριθμός) που είναι πρωτεύον κλειδί και το παίρνει από τον πίνακα ingredient_group, ένα ingredient_id(ακέραιος αριθμός) που είναι πρωτεύον κλειδί και το παίρνει από τον πίνακα ingredient. Κανένα από τα πεδία δε μπορεί να είναι κενό(null). Η σχέση του πίνακα με τον ingredient_group είναι 1-N, δηλαδή μία ομάδα υλικών μπορεί να έχει πολλά υλικά.

nutrition_info:

Περιλαμβάνει ένα id(ακέραιος αριθμός, auto increment) καθώς και τις διατροφικές πληροφορίες της συνταγής. Πιο συγκεκριμένα, περιλαμβάνει calories(float), carbohydrates(float), protein(float), fat(float), saturated_fat(float), sugars(float). Καθένα από αυτά είναι υποχρεωτικά μοναδικό και πρωτεύον κλειδί είναι το id.

Επιπρόσθετα, χρησιμοποιήθηκαν 2 triggers:

1. Στον πίνακα “ingredient” χρησιμοποιήθηκε πριν την εισαγωγή του στοιχείου(BEFORE INSERT trigger)[19], το οποίο κάνει hashing το στοιχείο με τον αλγόριθμο MD5[20].
2. Στον πίνακα “recipe” χρησιμοποιήθηκε πριν την εισαγωγή του στοιχείου(BEFORE INSERT trigger), το οποίο κάνει hashing το στοιχείο με τον αλγόριθμο MD5.

2.2.2 Scrapers

Για τον scraper χρησιμοποιήσαμε μια βιβλιοθήκη στη Java, την “Jsoup” [21]. Εδώ δώσαμε προτεραιότητα στην ταχύτητα του scraping και για αυτό προτιμήσαμε εργαλείο Java έναντι κάποιου σε Python. Παρόλο που και το κομμάτι του scraping, όπως και

αυτό του crawling, γίνεται ασύγχρονα, η βιβλιοθήκη Jsoup ήταν αρκετά εύχρηστη, όποτε προτιμήθηκε έναντι κάποιας σε Python που θα ήταν σίγουρα και πιο αργή. Η υλοποίηση σε Java ήταν αρκετά πιο φλύαρη από ό,τι θα ήταν ενδεχομένως σε Python, που θα υλοποιούσαμε το ίδιο σε λιγότερες γραμμές κώδικα, όμως το αποτέλεσμα παραμένει η υψηλότερη ταχύτητα.

Για τα παραπάνω δεν χρησιμοποιήθηκαν συγκεκριμένα Design Patterns(μοτίβα σχεδίασης), αλλά χρησιμοποιήθηκε γενικά στοιχειώδης αντικειμενοστραφής προγραμματισμός.

2.2.3 Crawlers

Για την υλοποίηση του crawler, επιλέξαμε να χρησιμοποιήσουμε το “Scrapy” που είναι εργαλείο σε Python. Ο λόγος που το προτιμήσαμε είναι η ευκολία, η απλότητα και η ταχύτητα της υλοποίησης. Για την κατασκευή ενός crawler μπορούσαμε να επιλέξουμε ανάμεσα σε διάφορα άλλα εργαλεία όπως είναι πχ το “Apache Nutch” το οποίο είναι γραμμένο σε Java.

Ωστόσο, η Python έχει εξελιχθεί πάρα πολύ και έχει πολύ καλές βιβλιοθήκες, οι οποίες παρέχουν με κάποια ελάχιστα imports(εισαγωγές) μια σειρά από δυνατότητες και λειτουργίες. Για τις αντίστοιχες δυνατότητες και λειτουργίες αν χρησιμοποιούσαμε κάποιο άλλο εργαλείο όπως το Nutch, όλη η διαδικασία θα ήταν αρκετά πιο συνθέτη και χρονοβόρα, καθώς μόνο και μόνο για τα αρχικά settings(ρυθμίσεις) και configurations(διαμορφώσεις) πρέπει να μελετήσουμε και να κατανοήσουμε έναν πυκνό όγκο του documentation(τεκμηρίωση) του εργαλείου. Παρόλο που η υλοποίηση ενός crawler χρησιμοποιώντας άλλο εργαλείο θα ήταν πολύ δυσκολότερη, αξίζει να αναφέρουμε πως ενδεχομένως η ταχύτητα του crawling αυτού καθαυτού να ήταν γρηγορότερη, αν χρησιμοποιούσαμε εργαλείο γραμμένο σε Java (όπως το Apache Nutch).

Ωστόσο, στο πλαίσιο της δίκης μας εργασίας, η ταχύτητα του χρόνου ανάπτυξης του crawler τάχθηκε πιο σημαντική από ότι η ταχύτητα εκτέλεσης, καθώς ο crawler μπορεί να τρέχει ασύγχρονα από την υπόλοιπη εφαρμογή. Πιο συγκεκριμένα, το crawling δεν εκκινείται τη στιγμή που κάνει κάποιο αίτημα ο χρήστης (κάποια αναζήτηση), αλλά γίνεται εκ των προτέρων και μπορεί να επαναληφθεί ανά πάσα στιγμή ανεξάρτητα ή και παράλληλα με την χρήση της υπόλοιπης εφαρμογής από τον άνθρωπο.

2.2.4 Rest API

Για το backend υλοποιήσαμε ένα REST API με Spring Boot. Για το REST API είχαμε να διαλέξουμε ανάμεσα σε πολλά και γνωστά εργαλεία όπως το Django που είναι σε Python, το Spring Boot που είναι σε Java, μπορούσαμε με Node.js που είναι σε Javascript, Laravel που είναι σε PHP, μπορούσαμε με .NET Core που είναι σε C# και άλλες πολλές τεχνολογίες.

Η επιλογή της Java έγινε γιατί είναι portable («φορητή»), τρέχει σε οποιοδήποτε υπολογιστή (σε αντίθεση με τη .NET Core ας πούμε). Επίσης, η ταχύτητα εδώ παίζει πολύ σημαντικό ρόλο και για αυτό προτιμήσαμε Java, σε αντίθεση με το Django για παράδειγμα, που είναι σε python και είναι αρκετά πιο αργό.

Τέλος ένας ακόμη λόγος, ήταν πως το Spring Boot χρησιμοποιείται πολύ στην αγορά. Σε σχέση δηλαδή με το Django που θα έλεγε κανείς ότι είναι το ανταγωνιστικό του, η ζήτηση για Spring Boot φαίνεται να είναι πολύ μεγαλύτερη. Συνολικά, η υλοποίηση REST API ξεκινώντας από το μηδέν γίνεται αρκετά γρήγορα σε Spring Boot, το οποίο αποτελεί επίσης ένα βασικό πλεονέκτημά του.

Ωστόσο εννοείται πως και το σύνολο των άλλων γνωστών εργαλείων όπως το Laravel(PHP) ή το Django(Python) θα μπορούσαν να καλύψουν κάθε μας ανάγκη για την εργασία αυτή.

2.2.5 User Interface

Στην επιλογή του frontend, καθοριστικό ρόλο έπαιξε η ευκολία του εργαλείου. Για το λόγο αυτόν επιλέξαμε Vue.js. Στις βασικές μας επιλογές υπήρχαν η React(Meta) και η Angular(Google). Στο παρελθόν δεν είχε τύχει να αποκτήσουμε εμπειρία σε κάποια από αυτές τις τεχνολογίες, οπότε η Angular κρίθηκε πολύ χρονοβόρα στην εκμάθηση της. Η συγκεκριμένη χρησιμοποιείται από μεγάλες εταιρίες και θα απαιτούσε αρκετή εκμάθηση. Την είδαμε και τη μελετήσαμε, παρόλα αυτά για το πλαίσιο της εργασίας θεωρήθηκε μάλλον ακατάλληλη καθώς θα καθυστερούσε την ανάπτυξη σημαντικά. Η React είναι η ίσως πρώτη σε ζήτηση αυτή τη στιγμή, όμως ήταν κι αυτή αρκετά σύνθετη, οπότε επιλέξαμε τη Vue.js που ήταν η πιο απλή από τις διαθέσιμες επιλογές.

Ένας ακόμη βασικός παράγοντας και εδώ είναι φυσικά η ταχύτητα. Η Vue.js είναι η γρηγορότερη από τις επιλογές μας.

3. ΜΕΘΟΔΟΛΟΓΙΑ ΥΛΟΠΟΙΗΣΗΣ

3.1 Backend

Σα μεθοδολογία κατά την υλοποίηση μου, θα μπορούσαμε να πούμε πως τα πήγαμε από τα αριστερά προς τα δεξιά. Παρατηρώντας το διάγραμμα παραπάνω δηλαδή, ξεκινήσαμε με τους crawlers και τελειώσαμε με το frontend. Γενικά θεωρήσαμε πως όταν πραγματοποιούμε περίπλοκες εφαρμογές πολλαπλών επιπέδων όπως αυτή, βολεύει η υλοποίηση με συνεχή τρόπο (με τη σειρά).

Ξεκινώντας την υλοποίηση του crawler, πειραματιστήκαμε για αρκετό χρονικό διάστημα με το “Apache Nutch”, το οποίο όπως προαναφέραμε είναι ένα εργαλείο σε Java με πολλές δυνατότητες. Δυστυχώς το εργαλείο αυτό αποδείχτηκε πολύ σύνθετο τόσο στην κατανόηση όσο και στη χρήση του καθώς ήθελε μια σειρά από settings και configurations για κάθε λειτουργικότητα του και το documentation του, αν και πυκνό, δε φάνηκε πολύ εύχρηστο.

Μετά από αρκετό πειραματισμό και αφού είχαμε καταφέρει ένα κομμάτι του crawling, αλλά όχι όλη τη λειτουργικότητα που θέλαμε, αποφασίσαμε να αναζητήσουμε κάποιο άλλο εργαλείο για crawling.

Ουσιαστικά αυτό που είχαμε κατά νου να υλοποιήσουμε, ήταν ένας crawler, ο οποίος θα μπορεί ξεκινώντας από ένα αρχικό σύνδεσμο ενός ιστοτόπου με συνταγές (πχ allrecipes.com), να βρίσκει όλες τις υποσελίδες, φιλτράροντας και αγνοώντας όσα links δεν αφορούν συνταγές και να τα γράφει σε ένα αρχείο txt(κειμένου). Με τη χρήση του Apache Nutch δυστυχώς είχαμε καταφέρει τη μισή λειτουργικότητα αυτού και έτσι αποφασίσαμε να στραφούμε σε κάποιο εργαλείο στην Python, το οποίο θα ήταν μεν πιο αργό, αλλά θα ήταν αποτελεσματικό.

Πράγματι, μετά από έρευνα επιλέξαμε το “Scrapy” το οποίο με λίγες γραμμές κώδικα κάνει όσα ακριβώς περιγράφουμε παραπάνω, δημιουργώντας τα txt αρχεία με τα links του κάθε site, τα οποία αργότερα θα φορτώσουμε στους scrapers.

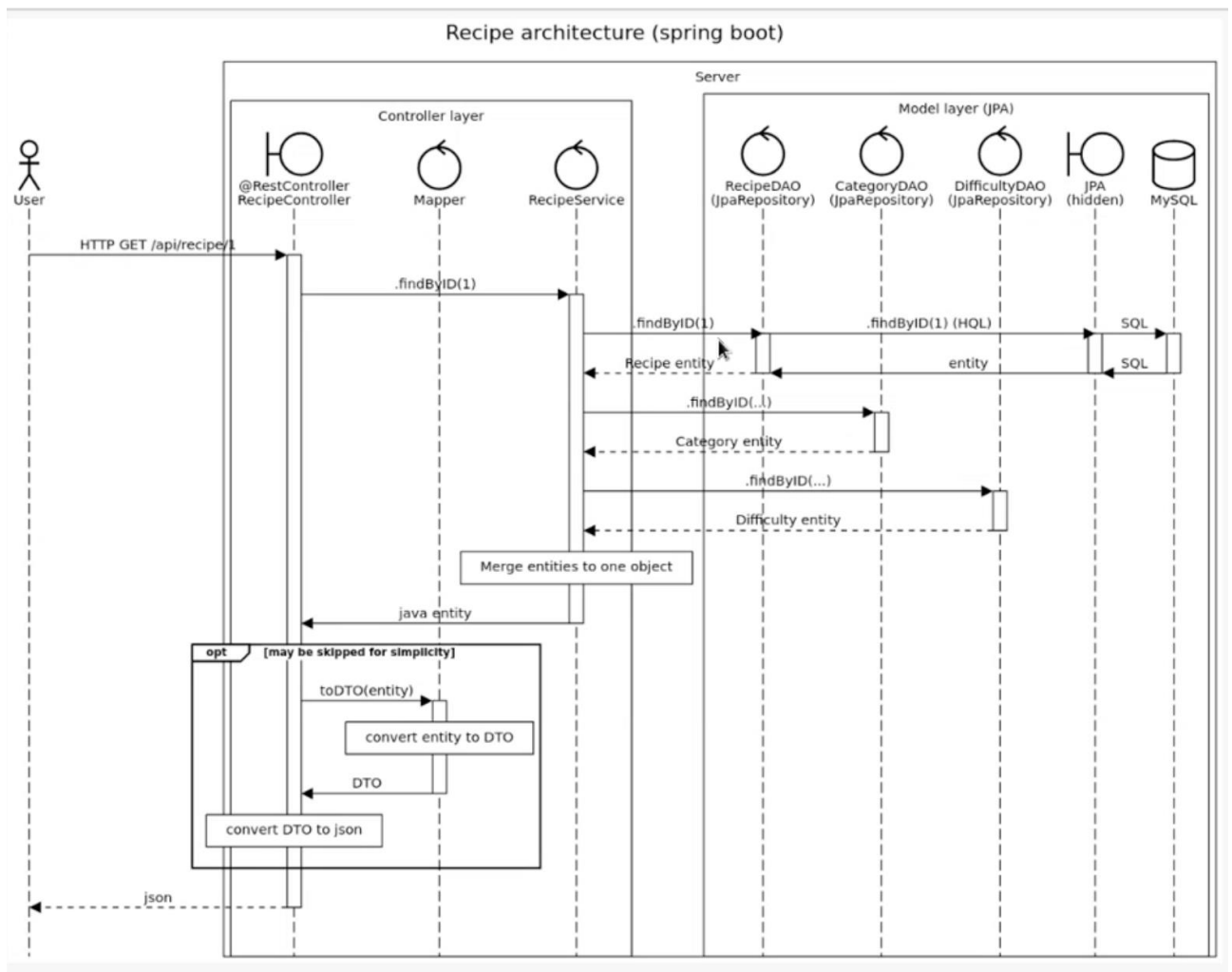
Έπειτα ξεκινήσαμε την υλοποίηση του πρώτου scraper, μελετήσαμε τις ιστοσελίδες, είδαμε τη μορφή τους και τι δεδομένα μπορούμε να εξάγουμε και γράψαμε τον κώδικα που ανακτά τα πρώτα data(δεδομένα) από το ένα site. Στο σημείο αυτό πήγαμε στο MySQL Workbench και σχεδιάσαμε τη βάση δεδομένων μας. Αφού φτιάξαμε το σχήμα και κάναμε forward engineering μέσα από το workbench, η βάση μας ήταν έτοιμη και μπορούσαμε να συνεχίσουμε την υλοποίηση των scrapers.

Υλοποιήσαμε 5 scrapers για τα 5 διαφορετικά sites συνταγών. Ο κάθε scraper διαβάζει από το αντίστοιχο αρχείο .txt του ένα-ένα τα links και από το καθένα ανακτά την επιθυμητή πληροφορία κάνοντας όλες τις κατάλληλες μετατροπές πριν τα εισαγάγει στη βάση δεδομένων, έτσι ώστε να έχουν όλα τα data από όλα τα sites όμοια μορφή.

Για την υλοποίηση του backend χρησιμοποιήσαμε Spring Boot, IntelliJ IDEA [22] και Maven [23].

3.1.1 Sequence Diagram

Ακολουθεί το Sequence Diagram της αρχιτεκτονικής του backend:



Σχήμα 3: Sequence Diagram

3.1.2 Περιγραφή αρχιτεκτονικής spring-boot

Για το σχεδιασμό του backend χρησιμοποιήσαμε τα παρακάτω:

- DAO(Data Access Object) [24]: η δουλειά τους είναι μόνο να παίρνουν data από τη database.
- Rest Controller [25]: κλάση που δέχεται τις αιτήσεις από το χρήστη(HTTP GET). Η δουλειά του είναι όταν πάρει μια αίτηση να ελέγξει αν η αίτηση είναι σωστή. Κάνει validation(επικύρωση) δηλαδή στο request(αίτημα) και επιστρέφει σφάλμα σε περίπτωση που λείπουν για παράδειγμα δεδομένα από κάποιο πεδίο, στην αναζήτηση που έχει υποχρεωτικά τιμές. Από τη στιγμή που η αίτηση περνάει, ο Rest Controller προωθεί την αίτηση στο Service.
- RecipeService: δουλειά του service είναι να πάρει αυτή την αίτηση του χρήστη και να τη στείλει στη βάση δεδομένων. Επιπλέον, αν θέλουμε να συνδυάζουμε δεδομένα από διαφορετικούς πίνακες, είναι δουλειά του να τα μαζέψει.

Πιο συγκεκριμένα, για κάθε πίνακα της database έχουμε ένα διαφορετικό DAO(data Access Object) που είναι κλάση JPA Repository του Spring Boot, δηλαδή μία διαφορετική κλάση που αναλαμβάνει την επικοινωνία με τη βάση δεδομένων. Το Service με το που θα πάρει την αίτηση καλεί ξεχωριστά τη μέθοδο του αντίστοιχου DAO για να πάρει τα αποτελέσματα και να τα ενώσει. Μόλις τα ενώσει, παίρνει αυτό το ενοποιημένο αποτέλεσμα, το στέλνει πίσω στο Rest Controller όπου ο Rest Controller πρέπει να πάρει αυτό το entity(οντότητα) και να το μετατρέψει σε DTO(data transaction object) [26][27].

Η διαφορά των entities από τα DTOs(data transaction objects):

Τα entities μας δείχνουν πώς παρουσιάζεται η πληροφορία στη βάση δεδομένων, ενώ τα DTOs είναι ουσιαστικά τα δεδομένα που ανταλλάζουμε με τον πελάτη μας. Είναι η επικοινωνία δηλαδή ανάμεσα στον controller και το user και όχι η επικοινωνία ανάμεσα στο controller και τη database. Το DTO δηλαδή αφορά το πώς παρουσιάζεται η πληροφορία στο χρήστη.

Άρα πρώτα μαζεύουμε τις πληροφορίες από τη βάση με το Service, μετά δίνουμε το Entity αυτό στον Controller. Ο Controller πλέον έχει όλη την πληροφορία από τη ΒΔ, επομένως από την πληροφορία αυτή πρέπει να αφαιρέσει πράγματα. Ο controller το στέλνει στο mapper, ο οποίος παίρνει το entity και το μετατρέπει σε DTO. Αντιγράφει δηλαδή από το entity όσα χρειάζεται και συνδυάζει όλα τα δεδομένα(αν είναι από

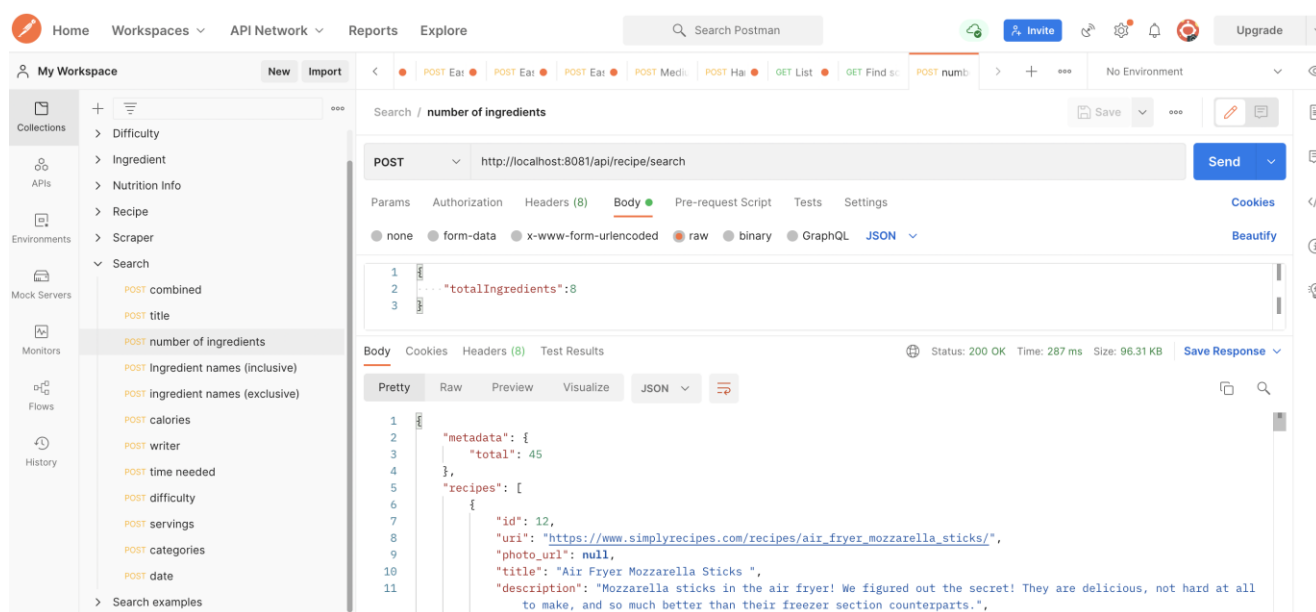
παραπάνω πίνακες) σε ένα τελικό αντικείμενο. Το αντικείμενο αυτό θα μετατραπεί σε json για να σταλεί στον πελάτη.

Οπότε μετά ο mapper το έχει μετατρέψει σε json, το δίνει πίσω στο Rest Controller, ο Rest Controller το μετατρέπει σε json και το στέλνει πίσω στον πελάτη.

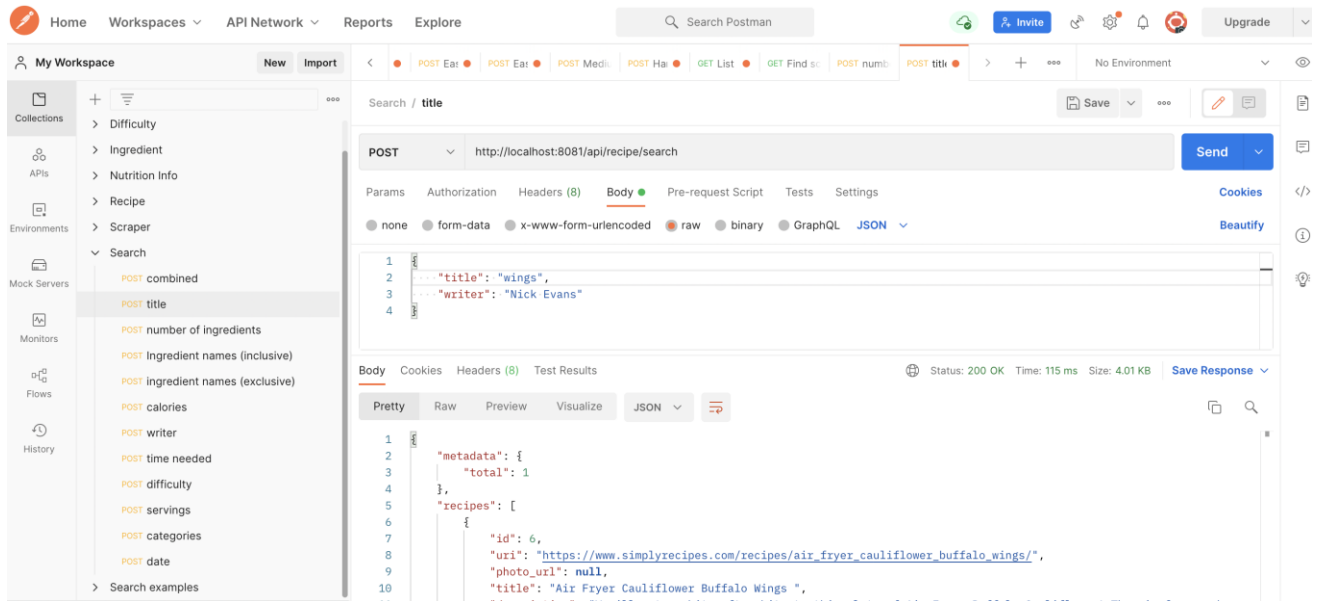
Άρα ο χρήστης στέλνει την αίτηση σε json στον controller, ο controller ελέγχει αν λείπει κάποιο πεδίο ή αν οι τιμές της αίτησης δεν είναι σωστές και αν είναι τις προωθεί στο service. Το service πηγαίνει και επικοινωνεί με τη βάση δεδομένων, παίρνει όλη την πληροφορία που χρειάζεται από έναν ή παραπάνω πίνακες και τα ενώνει όλα σε ένα τελικό αντικείμενο που έχει όλες τις πληροφορίες της database. Το service το δίνει πίσω στον controller, ο controller το δίνει στον mapper. Ο mapper αυτό το entity το μετατρέπει στο αντίστοιχο DTO όπου φιλτράρει τα δεδομένα ή και τα κάνει πιο ευανάγνωστα, αφαιρεί δεδομένα(αν υπάρχουν κάποια που δεν χρειάζονται) και μετά τα προωθεί πίσω στον controller, που τα παίρνει το κάνει json και το στέλνει στον πελάτη.

Μια από τις δυσκολίες ήταν η κατανόηση των αρχιτεκτονικών πολλαπλών επιπέδων όπως αυτή, ώστε να μπορέσουμε να χωρίσουμε τον κώδικα σωστά και να κάνει κάθε κλάση τη δουλειά της.

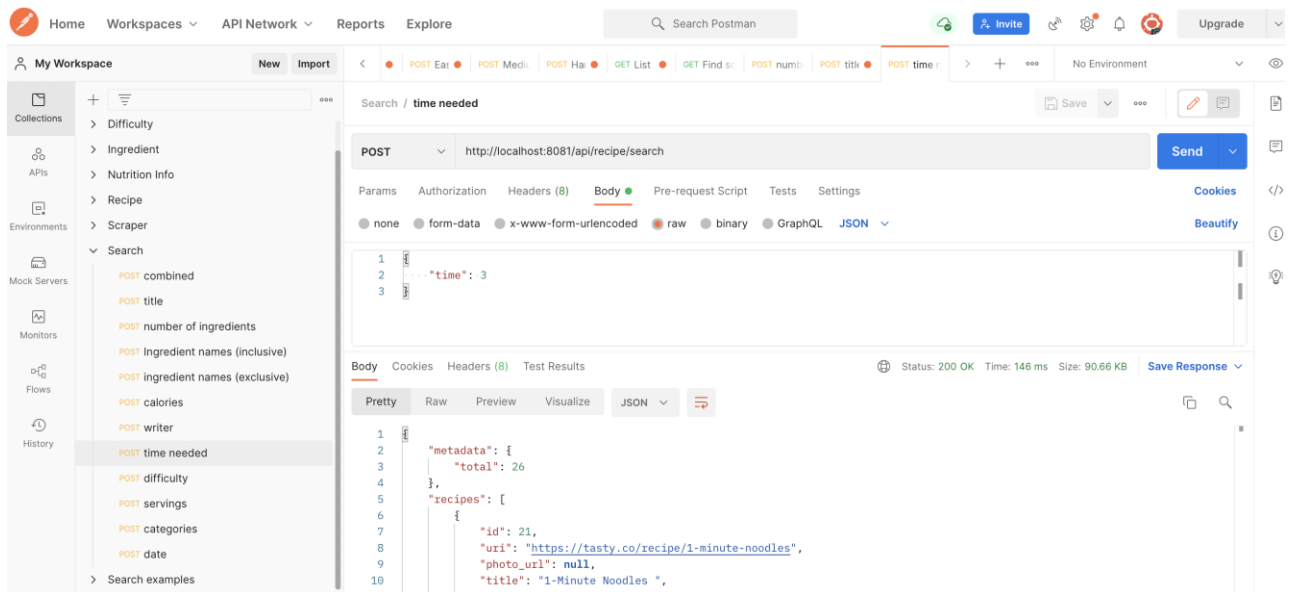
Πριν προχωρήσουμε στο frontend τεστάρουμε την εφαρμογή στο postman για να βεβαιωθούμε ότι λειτουργεί σωστά. Πιο συγκεκριμένα, φτιάξαμε διάφορα test παραδείγματα για τις αναζητήσεις και ελέγξαμε αν τα αποτελέσματα που λάβαμε αντιστοιχούσαν με αυτά στη βάση μας.



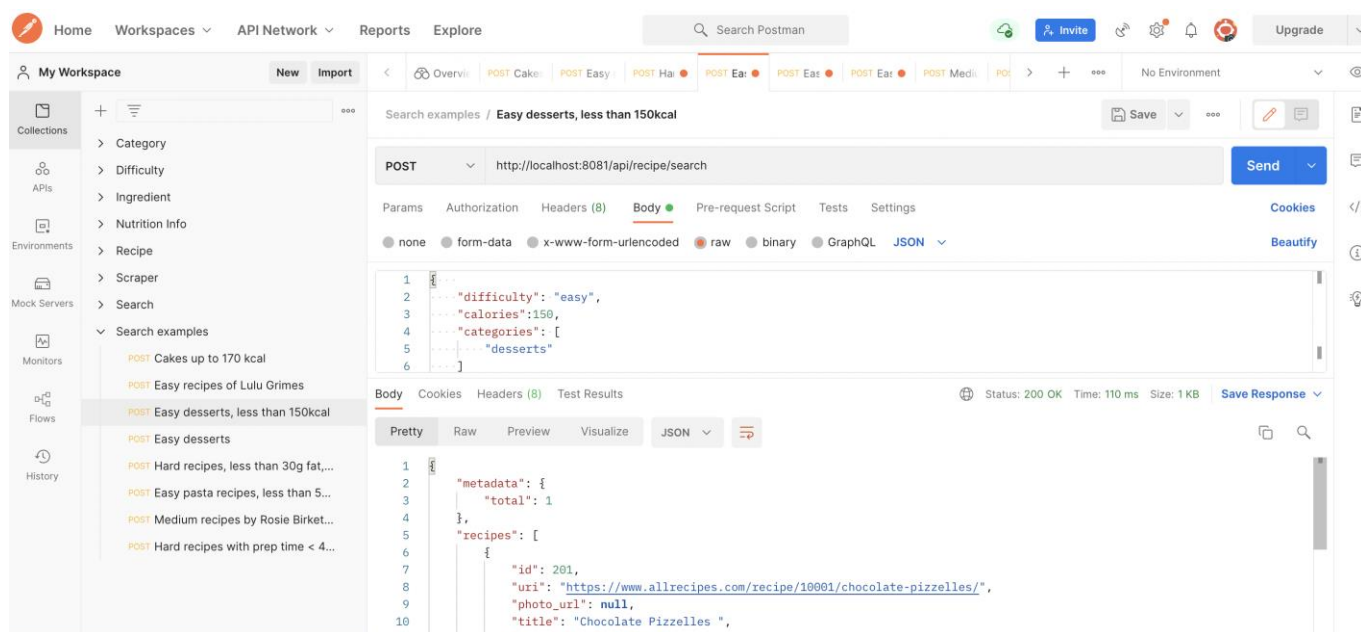
Εικόνα 3: Αναζήτηση με βάση το πλήθος των συστατικών



Εικόνα 4: Αναζήτηση με βάση τον τίτλο και τον συγγραφέα



Εικόνα 5: Αναζήτηση με βάση τον χρόνο της συνταγής



Εικόνα 6: Παράδειγμα πιο σύνθετης αναζήτησης, με βάση τις θερμίδες, την κατηγορία και τη δυσκολία

3.2 Frontend

Για την υλοποίηση του frontend χρησιμοποιήσαμε Nodejs [28], Vuejs, Vuecli [29], Visual Studio Code [30], Nuxt [31] με χρήση Options API (αντί του Composition). Επίσης για τη δημιουργία του frontend χρησιμοποιήσαμε ένα template το οποίο βρήκαμε στο διαδίκτυο και προσαρμόσαμε ώστε να παρουσιάζονται αποτελεσματικά και καλαίσθητα αποτελέσματα στον χρήστη. Το template είναι το "FoodHub - Food Delivery Template" [32], από αυτήν την πηγή "https://store.vuetifyjs.com/products/foodhub-food-delivery-template?utm_source=vuetifyjs.com&utm_medium=themes" και με το license "Personal Use". Τα design patterns που χρησιμοποιήθηκαν για το development του frontend ήταν όπως αυτά προϋπήρχαν στο template. Οι τεχνολογίες που χρησιμοποιεί το template είναι οι Vue, Nuxt.js και vuetify. Επίσης για να χρησιμοποιήσουμε REST στη Vuejs χρησιμοποιήσαμε και τη βιβλιοθήκη Axios. Επίσης χρησιμοποιήσαμε Nuxt, Babel και Sass.

- Vuetify[33]: μια πολύ διαδεδομένη βιβλιοθήκη που αποτελεί ένα σύνολο από έτοιμα components.
- Babel[34]: μεταγλωττιστής(transpiler) που ελέγχει και μετατρέπει τη javascript ώστε να παίζει καταλληλά σε όλους τους browsers(προγράμματα περιήγησης).

Με τη χρήση των παραπάνω δηλαδή το site μας παίζει σε όλους τους browsers και παρουσιάζεται ωραία και σε pc(υπολογιστές) αλλά και σε κινητές συσκευές.

- Nuxt: επιτρέπει το σάιτ να σκαναριστεί εύκολα από μηχανές αναζήτησης ώστε να φαίνεται στο google.

Τα παραπάνω περιλαμβάνονταν στο template και εμείς προσθέσαμε και το nuxtjs/axios το οποίο είναι μια βιβλιοθήκη της vuejs που χρησιμοποιείται για να επικοινωνούμε με το server, για να κάνουμε δηλαδή κλήσεις και να στέλνουμε τα ερωτήματα της αναζήτησης του χρήστη και να λαμβάνουμε την απάντηση.

Για τον crawler και τους scrapers δεν έχει χρησιμοποιηθεί κάποια συγκεκριμένη αρχιτεκτονική ή κάποιο συγκεκριμένο design pattern. Όσον αφορά τη βάση, έχουμε κάνει κανονικοποίηση της ΒΔ μέχρι κανονική μορφή BCNF[35].

Άλλα υποστηρικτικά εργαλεία που χρησιμοποιήσαμε:

Postman[36], HTTPie[37].

Για την ανάπτυξη κώδικα σε Java χρησιμοποιήσαμε το IntelliJ IDEA, για Python το Pycharm[38] και για Vue.js το Visual Studio Code.

3.3 Testing

Όσον αφορά τις δοκιμές των διάφορων κομματιών της εφαρμογής, κάναμε τα παρακάτω:

Για τον κάθε Crawler, η υλοποίηση έγινε σε Pycharm και για να το τεστάρουμε τρέξαμε όλο το crawling και ελέγξαμε ότι το txt αρχείο που παράγεται για κάθε site έχει σωστά urls. Στο σημείο αυτό χρειάστηκε πειραματισμός και μετατροπές ώστε να μην υπάρχουν πολλαπλότητα ή σύνδεσμοι του site που δεν αφορούν συνταγές.

Για τον κάθε Scraper, η υλοποίηση έγινε σε IntelliJ IDEA και για να τεστάρουμε το scraping πηγαίναμε στη βάση δεδομένων και ελέγχαμε ότι τα δεδομένα που ανακτήθηκαν για κάθε link είναι τα αναμενόμενα.

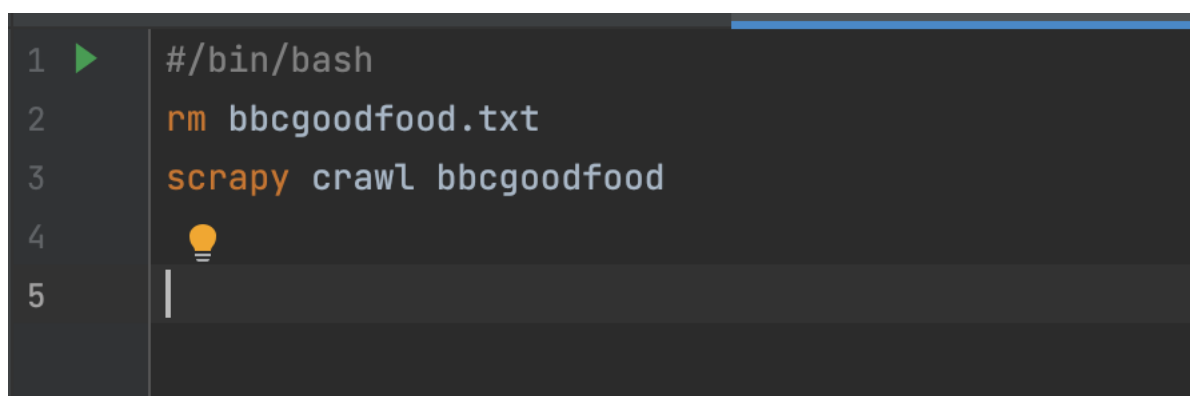
Για το backend(Spring Boot), η υλοποίηση έγινε σε IntelliJ IDEA και το τεστάρουμε με το Postman.

Για το frontend(Vue.js), η υλοποίηση έγινε με Visual Studio Code και το τεστάρουμε χειροκίνητα σαν ανθρώπινος χρήστης.

4. USER MANUAL – APPLICATION DEMO

4.1 Παρουσίαση Εκτέλεσης

Για να τρέξουμε τους crawlers, έχουμε φτιάξει ξεχωριστά bash scripts αρχεία και χρησιμοποιώντας το περιβάλλον του Pycharm, πατάμε απλά Run(εκτέλεση) στο script και τρέχει ο αντίστοιχος crawler. Για παράδειγμα, για να ανακτήσουμε όλα τα links του bbcgoodfood.com, τρέχουμε το script “scrape_bbcgoodfood.sh”.



```
1 ▶ #/bin/bash
2  rm bbcgoodfood.txt
3  scrapy crawl bbcgoodfood
4  📍
5  |
```

Εικόνα 7: Εκτέλεση του script για το crawling του bbcgoodfood.com

Έτσι ξεκινάει το crawling και καταγράφονται όλα τα links που πληρούν τις προϋποθέσεις μας στο “bbcgoodfood.txt”. Το crawling συνεχίζει κάπως έτσι (εικόνα από το terminal(τερματικό) του pycharm):

```

2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/porcini-pancetta-spelt-soup>
{'url': 'https://www.bbcgoodfood.com:443/recipes/porcini-pancetta-spelt-soup', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/soup-recipes?page=4', 'status': 200}
2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/best-british-burgers-triple-cooked-chips>
{'url': 'https://www.bbcgoodfood.com:443/recipes/best-british-burgers-triple-cooked-chips', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/burger-recipes?page=3', 'status': 200}
2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/mango-pineapple-mojito>
{'url': 'https://www.bbcgoodfood.com:443/recipes/mango-pineapple-mojito', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes', 'status': 200}
2022-10-21 17:01:01 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://www.bbcgoodfood.com:443/recipes/bramble> (referer: https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes)
2022-10-21 17:01:01 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://www.bbcgoodfood.com:443/recipes/kiwirikui> (referer: https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes)
2022-10-21 17:01:01 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://www.bbcgoodfood.com:443/recipes/hurricane-cocktail> (referer: https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes)
2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/cherry-bakewell-cocktail>
{'url': 'https://www.bbcgoodfood.com:443/recipes/cherry-bakewell-cocktail', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes', 'status': 200}
2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/beef-burger-sweet-potato-chilli-chips>
{'url': 'https://www.bbcgoodfood.com:443/recipes/beef-burger-sweet-potato-chilli-chips', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/burger-recipes?page=3', 'status': 200}
2022-10-21 17:01:01 [scrapy.core.scrapers] DEBUG: Scraped from <200 https://www.bbcgoodfood.com:443/recipes/bramble>
{'url': 'https://www.bbcgoodfood.com:443/recipes/bramble', 'referer': b'https://www.bbcgoodfood.com/recipes/collection/fruity-cocktail-recipes', 'status': 200}

```

Εικόνα 8: Κατά τη διάρκεια του crawling του bbcgoodfood.com

Τέλος η εικόνα στο terminal είναι αυτή:

```

2022-10-21 17:02:01 [scrapy.core.engine] INFO: Closing spider (finished)
2022-10-21 17:02:01 [scrapy.statscollectors] INFO: Dumping Scrapy stats:
{'downloader/exception_count': 2632,
 'downloader/exception_type_count/scrapy.exceptions.IgnoreRequest': 2566,
 'downloader/exception_type_count/twisted.internet.error.TimeoutError': 66,
 'downloader/request_bytes': 15357435,
 'downloader/request_count': 42600,
 'downloader/request_method_count/GET': 42600,
 'downloader/response_bytes': 2564543142,
 'downloader/response_count': 42534,
 'downloader/response_status_count/200': 36578,
 'downloader/response_status_count/301': 5915,
 'downloader/response_status_count/302': 2,
 'downloader/response_status_count/404': 34,
 'downloader/response_status_count/502': 5,
 'dupefilter/filtered': 3833000,
 'elapsed_time_seconds': 9036.45806,
 'finish_reason': 'finished',
 'finish_time': datetime.datetime(2022, 10, 21, 14, 2, 1, 139675),
 'httpcompression/response_bytes': 13633185522,
 'httpcompression/response_count': 36611,
 'item_scraped_count': 36575,
 'log_count/DEBUG': 85950,

```

Εικόνα 9: Η εικόνα στο terminal μετά την ολοκλήρωση του crawling

```
'item_scraped_count': 36575,
'log_count/DEBUG': 85958,
'log_count/INFO': 47,
'memusage/max': 557367296,
'memusage/startup': 62652416,
'offsite/domains': 4203,
'offsite/filtered': 626184,
'request_depth_max': 186,
'response_received_count': 36612,
'retry/count': 71,
'retry/reason_count/502 Bad Gateway': 5,
'retry/reason_count/twisted.internet.error.TimeoutError': 66,
'robotstxt/forbidden': 2566,
'robotstxt/request_count': 2,
'robotstxt/response_count': 2,
'robotstxt/response_status_count/200': 2,
'scheduler/dequeued': 45164,
'scheduler/dequeued/memory': 45164,
'scheduler/enqueued': 45164,
'scheduler/enqueued/memory': 45164,
'start_time': datetime.datetime(2022, 10, 21, 11, 31, 24, 681615)}
2022-10-21 17:02:01 [scrapy.core.engine] INFO: Spider closed (finished)
anna_petridou@192 crawler_python %
```

Εικόνα 10: : Η τελική εικόνα στο terminal μετά την ολοκλήρωση του crawling

Το text αρχείο δημιουργείται στον ίδιο φάκελο και είναι της μορφής:

```
1 https://www.bbcgoodfood.com/recipes/pork-stroganoff
2 https://www.bbcgoodfood.com/recipes/lamb-ragu
3 https://www.bbcgoodfood.com/recipes/marble-bundt-cake
4 https://www.bbcgoodfood.com/recipes/vegan-waffles
5 https://www.bbcgoodfood.com/recipes/cabbage-rolls
6 https://www.bbcgoodfood.com/recipes/chana-daal-chaat-with-tamarind-herbs
7 https://www.bbcgoodfood.com/recipes/slow-cooker-hot-chocolate
8 https://www.bbcgoodfood.com/recipes/slow-cooker-bio-yogurt
9 https://www.bbcgoodfood.com/recipes/poached-apricots-rosewater
10 https://www.bbcgoodfood.com/recipes/slow-cooker-spiced-root-lentil-casserole
11 https://www.bbcgoodfood.com/recipes/madeira-roast-quince-fool
12 https://www.bbcgoodfood.com/recipes/membrillo-glazed-halloumi-skewers
13 https://www.bbcgoodfood.com/recipes/venison-quince
14 https://www.bbcgoodfood.com/recipes/membrillo-chorizo-cheddar-toastie
15 https://www.bbcgoodfood.com/recipes/toffee-apple-bread-butter-pudding
16 https://www.bbcgoodfood.com/recipes/open-face-pear-berry-pie
17 https://www.bbcgoodfood.com/recipes/squidgy-chocolate-pear-pudding
18 https://www.bbcgoodfood.com/recipes/roasted-aubergine-salad
19 https://www.bbcgoodfood.com/recipes/sweetcorn-chowder
20 https://www.bbcgoodfood.com/recipes/coconut-flour-cookies
21 https://www.bbcgoodfood.com/recipes/fat-rascals
22 https://www.bbcgoodfood.com/recipes/pear-almond-pavlova-trifle
23 https://www.bbcgoodfood.com/recipes/maple-pear-pecan-mascarpone-roulade
24 https://www.bbcgoodfood.com/recipes/chocolate-coconut-cake
25 https://www.bbcgoodfood.com/recipes/chicken-sausage-pasta
26 https://www.bbcgoodfood.com/recipes/salted-caramel-apple-pudding
27 https://www.bbcgoodfood.com/recipes/chicken-leek-brown-rice-stir-fry
28 https://www.bbcgoodfood.com/recipes/orecchiette-with-butter-beans-parsley-chilli-lemon
29 https://www.bbcgoodfood.com/recipes/5-day-tagine
30 https://www.bbcgoodfood.com/recipes/spinach-pancakes-with-harissa-yogurt-poached-eggs
31 https://www.bbcgoodfood.com/recipes/smoked-trout-tartlets
32 https://www.bbcgoodfood.com/recipes/lamb-with-olive-herb-stuffing
33 https://www.bbcgoodfood.com/recipes/celeriac-pancetta-thyme-soup
```

Εικόνα 11: Τα links που ανακτήθηκαν για το bbcgoodfood.com

```

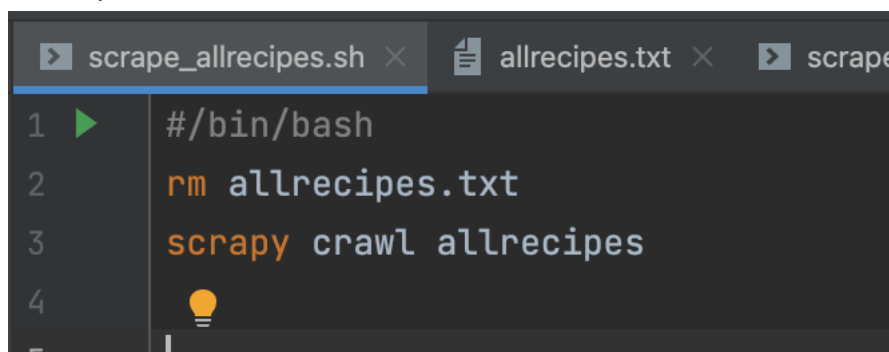
14978 https://www.bbcgoodfood.com/recipes/pasta-meatball-soup-cheesy-cROUTONS
14979 https://www.bbcgoodfood.com/recipes/rose-strawberry-syllabub
14980 https://www.bbcgoodfood.com/recipes/pork-sage-sausage-rolls
14981 https://www.bbcgoodfood.com/recipes/mediterranean-fish-couscous
14982 https://www.bbcgoodfood.com/recipes/raspberry-oat-traybake
14983 https://www.bbcgoodfood.com/recipes/creamy-tagliatelle-fennel
14984 https://www.bbcgoodfood.com/recipes/saucy-prawns
14985 https://www.bbcgoodfood.com/recipes/cheesy-eggy-bread-chunky-salad
14986 https://www.bbcgoodfood.com/recipes/fruity-lamb-kebabs-chilli-mayo
14987 https://www.bbcgoodfood.com/recipes/rice-pudding-spiced-plum-bake
14988 https://www.bbcgoodfood.com/recipes/goats-cheese-spring-onion-hazelnut-tart
14989 https://www.bbcgoodfood.com/recipes/syrup-crunchies
14990 https://www.bbcgoodfood.com/recipes/pumpkin-falafel-pockets
14991 https://www.bbcgoodfood.com/recipes/pea-pesto-prawn-spaghetti
14992 https://www.bbcgoodfood.com/recipes/pork-chops-bubble-n-leek-cakes
14993 https://www.bbcgoodfood.com/recipes/curried-lamb-peas-tomato-onion-salad
14994 https://www.bbcgoodfood.com/recipes/chinese-style-pork-fillet-fried-rice
14995 https://www.bbcgoodfood.com/recipes/clementine-grand-marnier-semi-freddos
14996 https://www.bbcgoodfood.com/recipes/prawn-cocktail-crostini
14997 https://www.bbcgoodfood.com/recipes/brown-sugar-brandy-cream
14998 https://www.bbcgoodfood.com/recipes/mushroom-aubergine-pizza-pie
14999 https://www.bbcgoodfood.com/recipes/crab-avocado-sesame-toasts
15000 https://www.bbcgoodfood.com/recipes/smoked-haddock-cumin-chowder
15001 https://www.bbcgoodfood.com/recipes/oaty-nutty-apple-crumble
15002 https://www.bbcgoodfood.com/recipes/chunky-tomato-avocado-salsa
15003 https://www.bbcgoodfood.com/recipes/sticky-chipolatas
15004 https://www.bbcgoodfood.com/recipes/brie-pecan-bites
15005 https://www.bbcgoodfood.com/recipes/spicy-paneer-skewers
15006 https://www.bbcgoodfood.com/recipes/apple-cornflake-pots
15007 https://www.bbcgoodfood.com/recipes/simple-mint-sauce
15008 https://www.bbcgoodfood.com/recipes/spaghetti-nests
15009 https://www.bbcgoodfood.com/recipes/mustardy-baked-onions
15010 https://www.bbcgoodfood.com/recipes/braised-leeks-apples
15011

```

Εικόνα 12: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 15010 σύνδεσμοι.

Το συγκεκριμένο txt αρχείο περιλαμβάνει συνολικά 15010 links από συνταγές. Με τα αντίστοιχα scripts τρέχουμε και τους άλλους 4 crawlers όπως φαίνεται παρακάτω:

- scrape_allrecipes.sh



```

scrape_allrecipes.sh x allrecipes.txt x scrape
1 #/bin/bash
2 rm allrecipes.txt
3 scrapy crawl allrecipes
4
5

```

Εικόνα 13: Εκτέλεση του script για το crawling του allrecipes.com

- scrape_cookingclassy.sh

```
1 ▶ #/bin/bash
2   rm cookingclassy.txt
3   scrapy crawl cookingclassy
4   
```

Εικόνα 14: Εκτέλεση του script για το crawling του cookingclassy.com

- scrape_simplyrecipes.sh

```
1 ▶ #/bin/bash
2   rm simplyrecipes.txt
3   scrapy crawl simplyrecipes
4   
```

Εικόνα 15: Εκτέλεση του script για το crawling του simplyrecipes.com

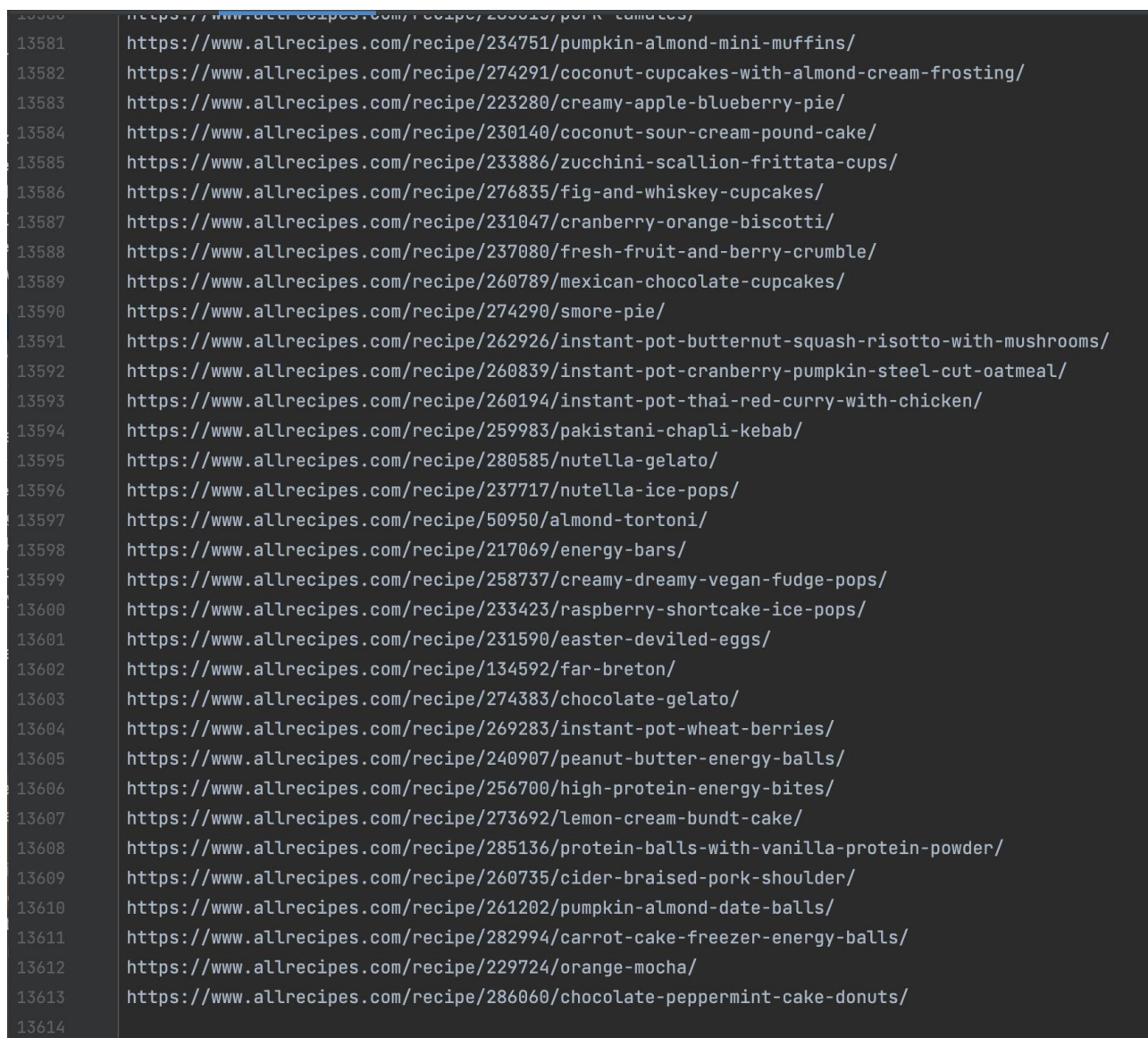
- scrape_tasty.sh

```
1 ▶ #/bin/bash
2   rm tasty.txt
3   scrapy crawl tasty
4   
```

Εικόνα 16: Εκτέλεση του script για το crawling του tasty.co

Τα αντίστοιχα txt αρχεία τους είναι της μορφής:

Για το allrecipes.txt, ανακτήθηκαν 13613 links.



Εικόνα 17: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 13613 σύνδεσμοι.

Για το cookingclassy.com ανακτήθηκαν 1713 links:


```
1681 https://www.cookingclassy.com/vanilla-cupcakes-recipe/
1682 https://www.cookingclassy.com/filet-mignon/
1683 https://www.cookingclassy.com/skeleton-gingerbread-cookies/?utm_source=pushengage&utm_medium=pushn
1684 https://www.cookingclassy.com/iced-pumpkin-cookies/?utm_source=Cooking+Classy&utm_campaign=aeca558
1685 https://www.cookingclassy.com/banana-berry-frozen-yogurt/
1686 https://www.cookingclassy.com/apple-cinnamon-roll-muffins/
1687 https://www.cookingclassy.com/strawberry-greek-frozen-yogurt/
1688 https://www.cookingclassy.com/banana-berry-smoothies-jamba-juice-copycat/
1689 https://www.cookingclassy.com/meringue-cookies/
1690 https://www.cookingclassy.com/triple-vanilla-blondies/
1691 https://www.cookingclassy.com/vanilla-bean-sugar-cookie-bars-with-vanilla-bean-frosting/
1692 https://www.cookingclassy.com/strawberry-meringue-cookies/
1693 https://www.cookingclassy.com/chocolate-chip-blondies/
1694 https://www.cookingclassy.com/caramel-cashew-cookies/
1695 https://www.cookingclassy.com/caramel-apple-cookies/
1696 https://www.cookingclassy.com/chile-relleno-grilled-cheese-sandwich/
1697 https://www.cookingclassy.com/fruit-punch/
1698 https://www.cookingclassy.com/simple-dill-dip-for-veggies-chips-or-crackers/
1699 https://www.cookingclassy.com/browned-butter-french-breakfast-muffins/
1700 https://www.cookingclassy.com/creamy-raspberry-hot-chocolate/
1701 https://www.cookingclassy.com/healthy-cookies/
1702 https://www.cookingclassy.com/raspberry-swirl-rolls/
1703 https://www.cookingclassy.com/pumpkin-cinnamon-swirl-bread/
1704 https://www.cookingclassy.com/maple-apple-bars-with-maple-cream-cheese-glaze/
1705 https://www.cookingclassy.com/rocky-road-oatmeal/
1706 https://www.cookingclassy.com/peanut-butter-bite-size-cookies/
1707 https://www.cookingclassy.com/upgraded-no-bake-cookies-two-ways/
1708 https://www.cookingclassy.com/glazed-lemon-cookies/
1709 https://www.cookingclassy.com/five-fun-ways-to-make-pbj/
1710 https://www.cookingclassy.com/chocolate-covered-bacon-bites/
1711 https://www.cookingclassy.com/campfire-tarts-and-my-first-giveaway/
1712 https://www.cookingclassy.com/monkey-bread-muffins-quite-possibly-the-best-thing-ive-ever-eaten/
1713 https://www.cookingclassy.com/farro-salad/
1714
```

Εικόνα 18: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 1713 σύνδεσμοι.

Για το tasty.co ανακτήθηκαν 2996 links:

```

2963 https://tasty.co/recipe/noon-cheese-dip
2964 https://tasty.co/recipe/one-skillet-samosa-pot-pie
2965 https://tasty.co/recipe/20-minute-one-pan-pizza
2966 https://tasty.co/recipe/southwestern-one-pan-shakshuka
2967 https://tasty.co/recipe/beef-enchiladas
2968 https://tasty.co/recipe/naan-breakfast-pizza
2969 https://tasty.co/recipe/chicken-salad-sandwich
2970 https://tasty.co/recipe/chicken-parm-lasagna
2971 https://tasty.co/recipe/no-bake-nutella-cheesecake
2972 https://tasty.co/recipe/chocolate-bananas-foster-pie
2973 https://tasty.co/recipe/how-to-throw-a-sushi-party
2974 https://tasty.co/recipe/baked-brie-bomb
2975 https://tasty.co/recipe/one-pot-lemon-pepper-chicken-rice
2976 https://tasty.co/recipe/sriracha-mayo-chicken-rice-balls
2977 https://tasty.co/recipe/mango-stuffed-sticky-rice-balls
2978 https://tasty.co/recipe/gyoza-dumplings
2979 https://tasty.co/recipe/easy-sheet-pan-southwestern-dinner
2980 https://tasty.co/recipe/instant-pot-chicken-noodle-soup
2981 https://tasty.co/recipe/roasted-chickpea-and-avocado-salad
2982 https://tasty.co/recipe/low-carb-biscuits-and-gravy
2983 https://tasty.co/recipe/savory-pepita-pan-de-elote-mexican-cornbread-with-charred-poblano-butter
2984 https://tasty.co/recipe/classic-chocolate-truffle
2985 https://tasty.co/recipe/chocolate-walnut-3-ingredient-fudge
2986 https://tasty.co/recipe/vampire-donuts
2987 https://tasty.co/recipe/the-best-crispy-buffalo-wings
2988 https://tasty.co/recipe/honey-garlic-chicken-veggie-skewers
2989 https://tasty.co/recipe/cauliflower-cheddar-fritters
2990 https://tasty.co/recipe/peanut-butter-jelly-doughnut-holes
2991 https://tasty.co/recipe/guacamole-deviled-eggs
2992 https://tasty.co/recipe/banana-berry-fruit-salad
2993 https://tasty.co/recipe/peanut-butter-stuffed-brownie-truffles
2994 https://tasty.co/recipe/tasty-peanut-butter-smores-smash-ice-cream-bars
2995 https://tasty.co/recipe/2-easy-make-ahead-school-lunches
2996 https://tasty.co/recipe/chicken-tortilla-soup
2997

```

Εικόνα 19: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 2996 σύνδεσμοι.

Για το simplyrecipes.com ανακτήθηκαν 2642 links:

```

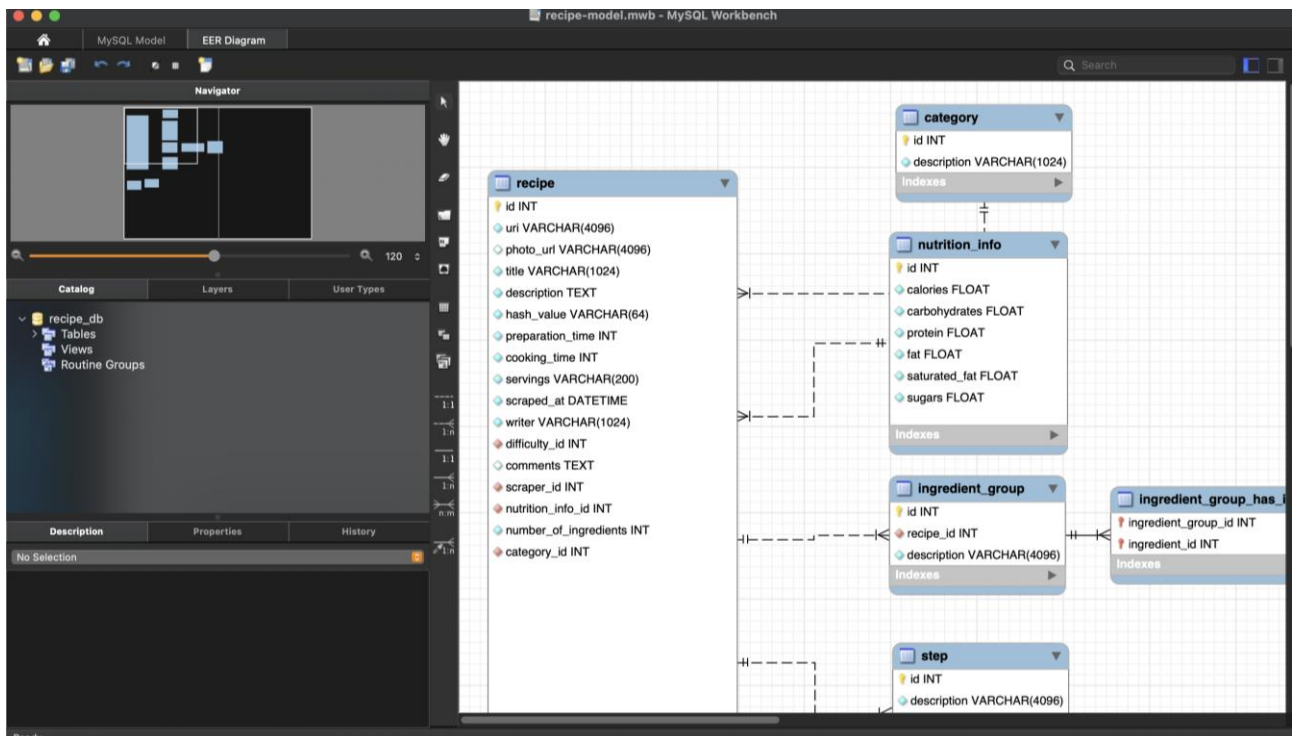
2609 https://www.simplyrecipes.com/recipes/grilled_marinated_flank_steak/
2610 https://www.simplyrecipes.com/recipes/honey_glazed_lemon_roast_chicken/
2611 https://www.simplyrecipes.com/recipes/how_to_dry_brine_and_roast_a_turkey/
2612 https://www.simplyrecipes.com/recipes/cowboy_steak_with_chimichurri_sauce/
2613 https://www.simplyrecipes.com/recipes/giant_ginger_cookies/
2614 https://www.simplyrecipes.com/recipes/coconut_chocolate_chip_cookies/
2615 https://www.simplyrecipes.com/recipes/rum_balls/
2616 https://www.simplyrecipes.com/recipes/candy_cane_cookies/
2617 https://www.simplyrecipes.com/recipes/hot_chocolate/
2618 https://www.simplyrecipes.com/recipes/chocolate_peppermint_swiss_roll/
2619 https://www.simplyrecipes.com/recipes/chipotle_flourless_chocolate_cake/
2620 https://www.simplyrecipes.com/recipes/bittersweet_chocolate_cake/
2621 https://www.simplyrecipes.com/recipes/how_to_make_whipped_cream/
2622 https://www.simplyrecipes.com/recipes/chocolate_layer_cake/
2623 https://www.simplyrecipes.com/recipes/kale_and_shaved_brussels_sprout_salad_with_bacon/
2624 https://www.simplyrecipes.com/recipes/cheesy_potato_casserole/
2625 https://www.simplyrecipes.com/recipes/steak_diane/
2626 https://www.simplyrecipes.com/recipes/roasted_cauliflower_and_mushroom_bolognese/
2627 https://www.simplyrecipes.com/recipes/pork_chops_with_braised_cabbage/
2628 https://www.simplyrecipes.com/recipes/low_carb_cranberry_cooler/
2629 https://www.simplyrecipes.com/recipes/pork_chops_with_dijon_sauce/
2630 https://www.simplyrecipes.com/recipes/easy_smothered_pork_chops/
2631 https://www.simplyrecipes.com/recipes/mexican_chocolate_ice_cream/
2632 https://www.simplyrecipes.com/recipes/triple_chocolate_cheesecake/
2633 https://www.simplyrecipes.com/recipes/chocolate_covered_pretzels/
2634 https://www.simplyrecipes.com/recipes/christmas_cracker_candy/
2635 https://www.simplyrecipes.com/recipes/english_toffee/
2636 https://www.simplyrecipes.com/recipes/almond_roca/
2637 https://www.simplyrecipes.com/recipes/chocolate_peanut_butter_bars/
2638 https://www.simplyrecipes.com/recipes/chocolate_ganache_torte/
2639 https://www.simplyrecipes.com/recipes/butter_pecan_cookies/
2640 https://www.simplyrecipes.com/recipes/peppermint_bark_chocolate_cookies/
2641 https://www.simplyrecipes.com/recipes/walnut_snowball_cookies/
2642 https://www.simplyrecipes.com/recipes/cinnamon_snap_cookies/

```

Εικόνα 20: Συνολικά για αυτό τον ιστότοπο ανακτήθηκαν 2642 σύνδεσμοι.

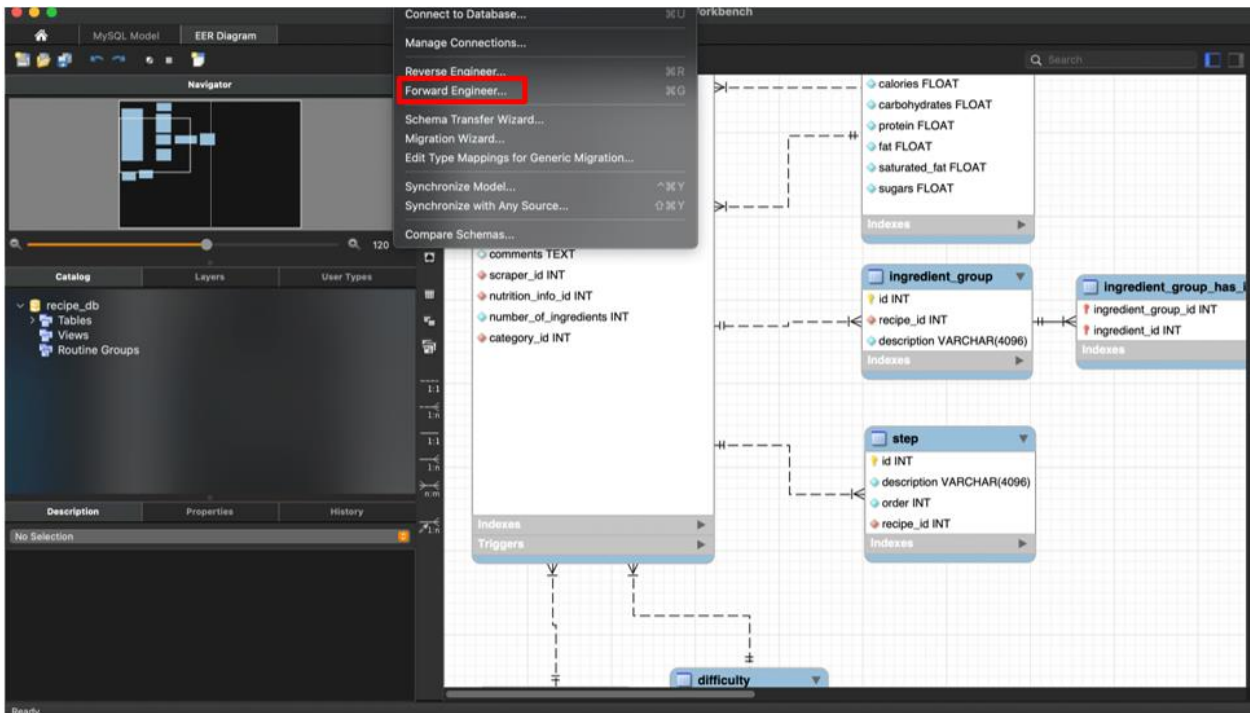
Μετά τους crawlers, προχωράμε και τρέχουμε τους scrapers. Για την εκτέλεση αυτών, πηγαίνουμε στο IntelliJ IDEA και κάνουμε run τη main του project στο φάκελο “scraper”. Πριν εκτελέσουμε τους scrapers, πηγαίνουμε στη βάση δεδομένων και κάνουμε forward engineering. Για τη βάση δεδομένων χρησιμοποιούμε το MySQL Workbench όπως φαίνεται παρακάτω:

Ανοίγουμε το database model που βρίσκεται στο φάκελο database και πηγαίνουμε στο workbench και επιλέγουμε “Database”.



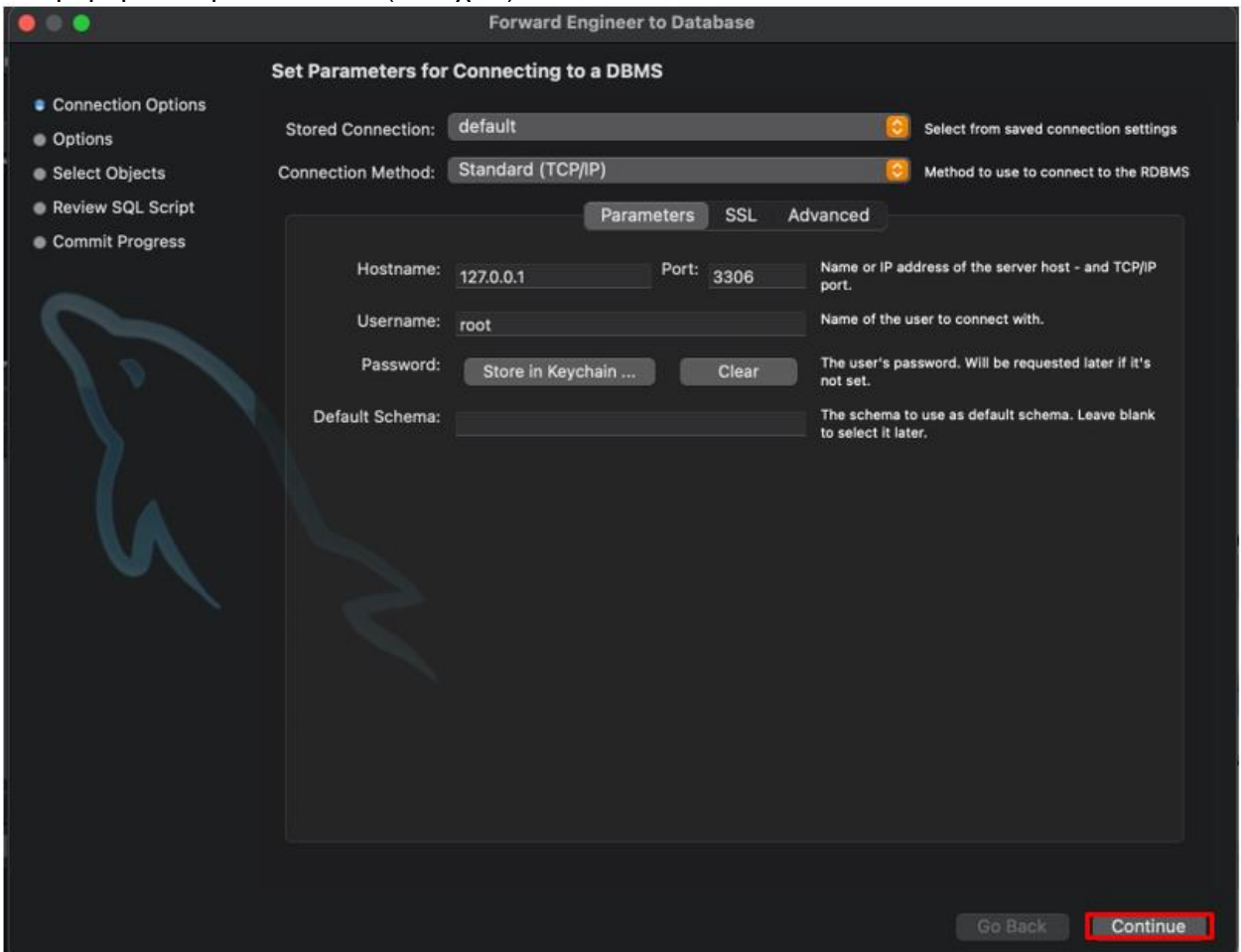
Εικόνα 21: Database model μέσα στο MySQL Workbench

Από το dropdown menu επιλέγουμε το “forward engineer” για να αρχικοποιήσουμε τη βάση δεδομένων.



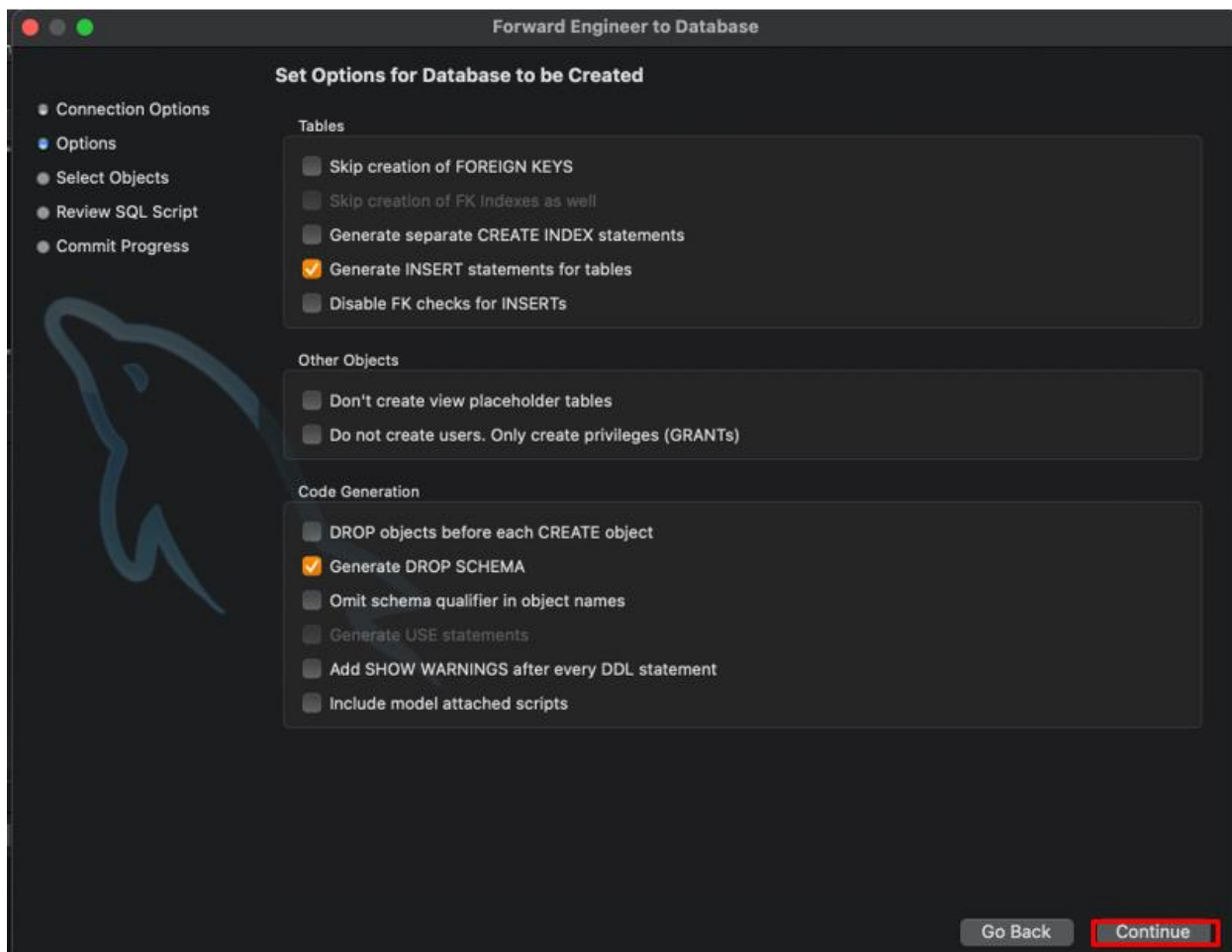
Εικόνα 22: Κλικ στο “Forward Engineer”

Στο ρομπρ πατάμε “Continue”(συνέχεια).

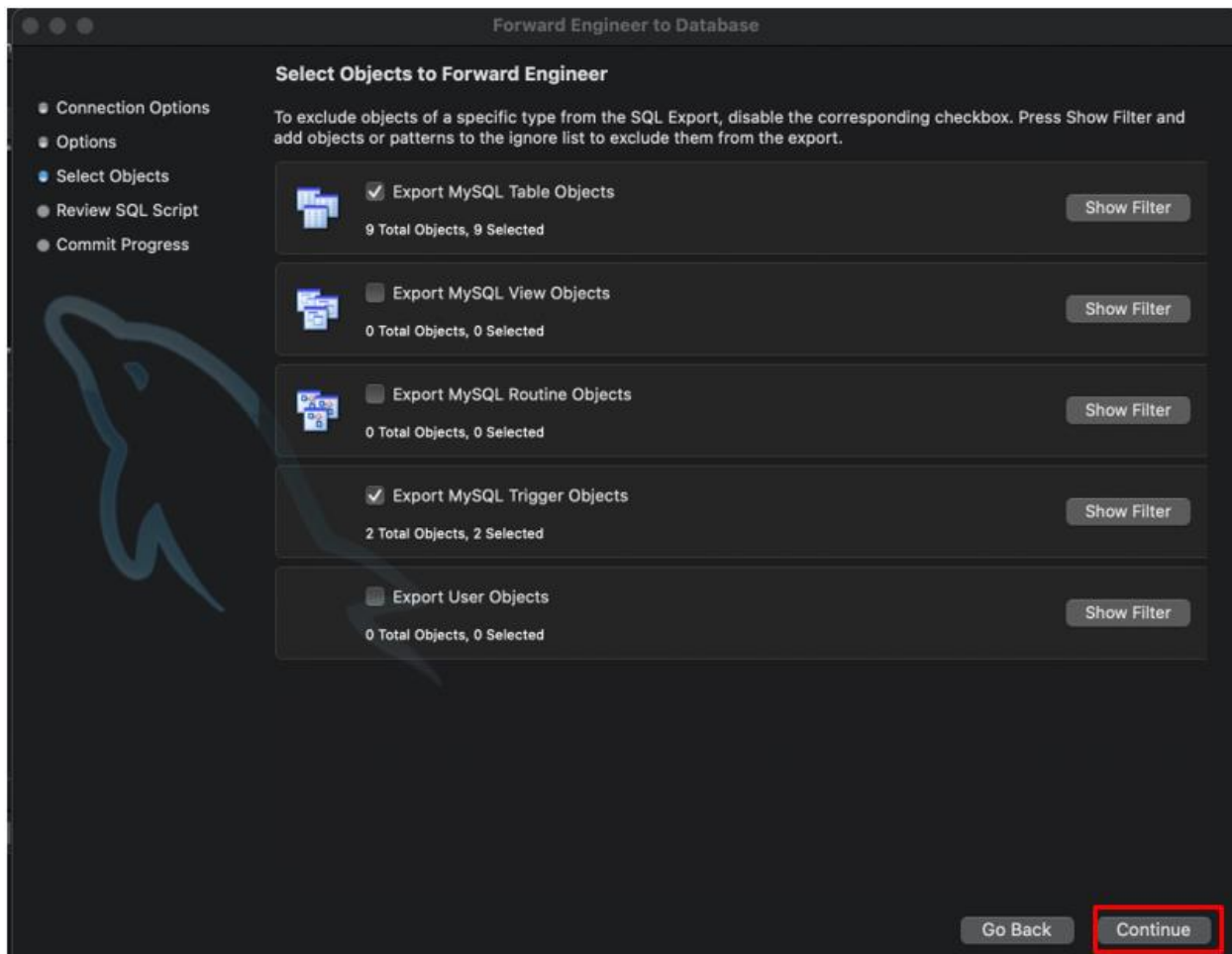


Εικόνα 23: Κλικ στο “Continue”

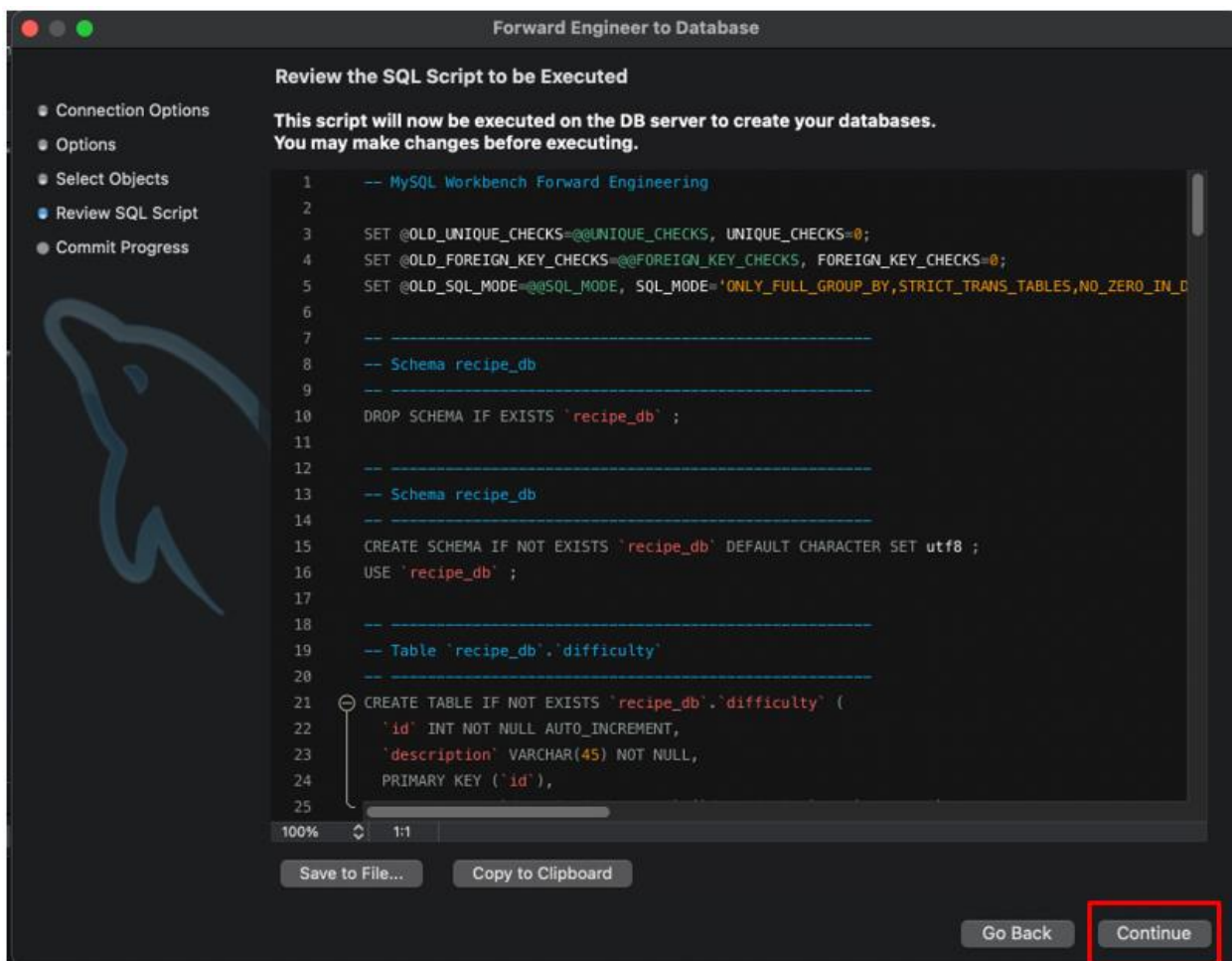
Και ξανά continue σε όλα τα επόμενα.



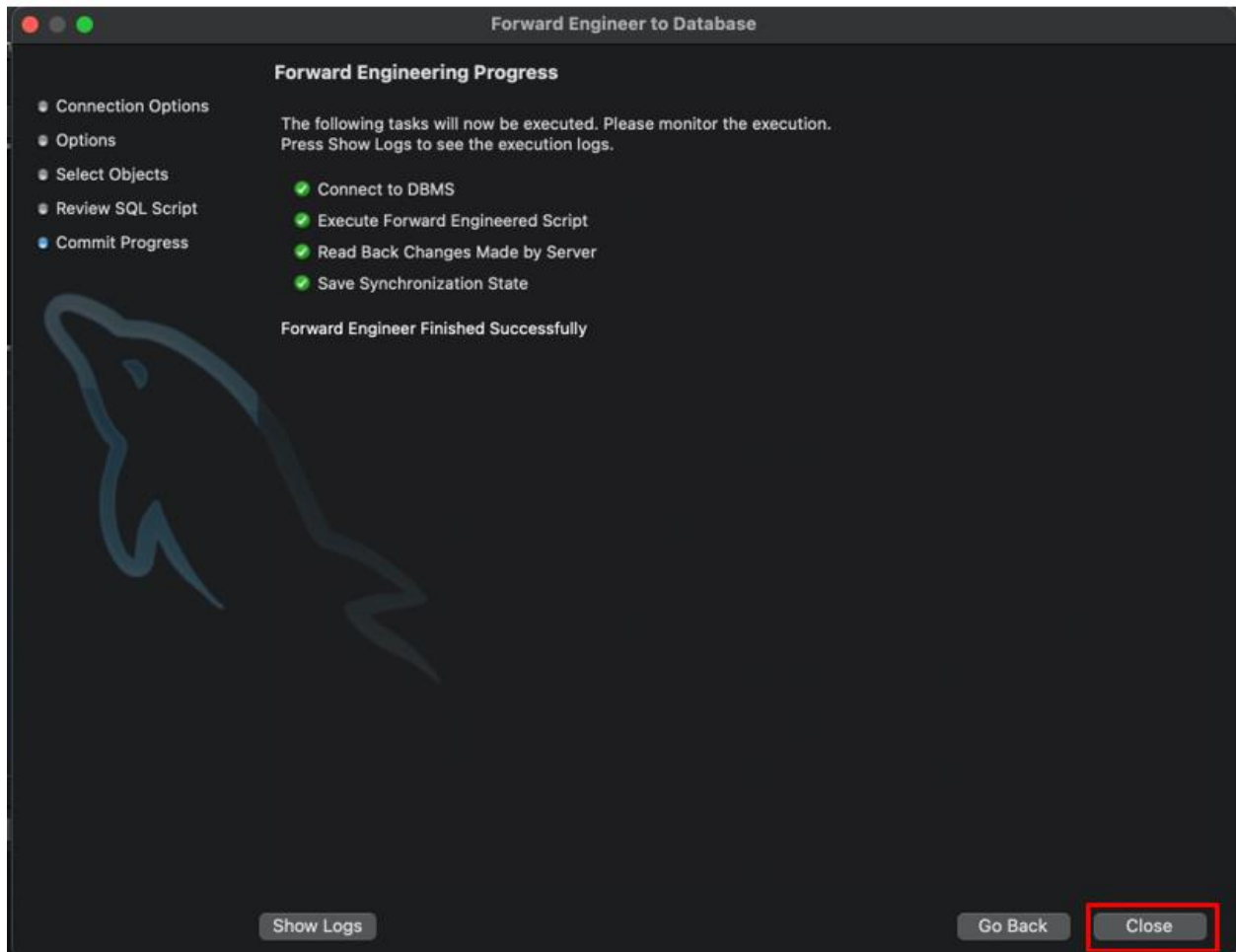
Εικόνα 24: Κλικ στο “continue”



Εικόνα 25: Κλικ στο “continue”

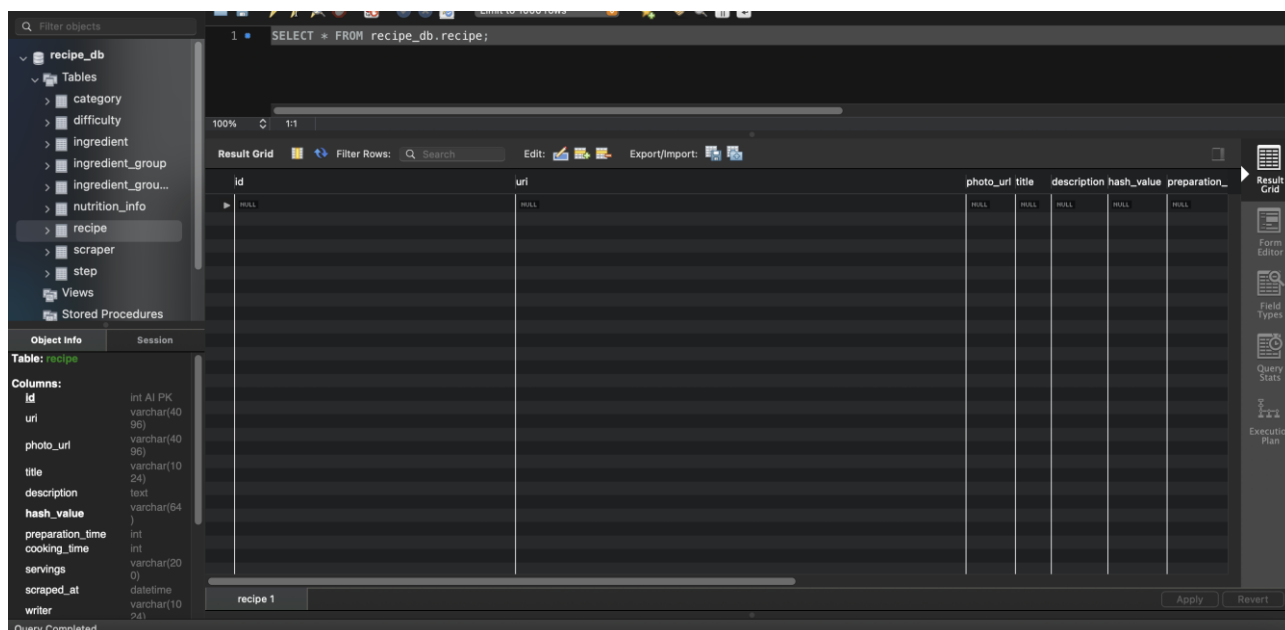


Εικόνα 26: Κλικ στο “continue”



Εικόνα 27: Κλικ στο “close”

Πατάμε “close”(κλείσιμο). Με ένα απλό query(ερώτημα) μπορούμε να ελέγξουμε ότι η βάση είναι άδεια αυτή τη στιγμή.



Εικόνα 28: Ο άδειος πίνακας που επιστρέφει το query

Έπειτα, αντιγράφουμε τα txt αρχεία από το φάκελο
 “../final_project/crawler_python/crawler_python/”
 στον φάκελο
 “../final_project/scraper/src/main/resources/”.

Έτσι πάμε στη main στον φάκελο scraper και τρέχουμε το project:

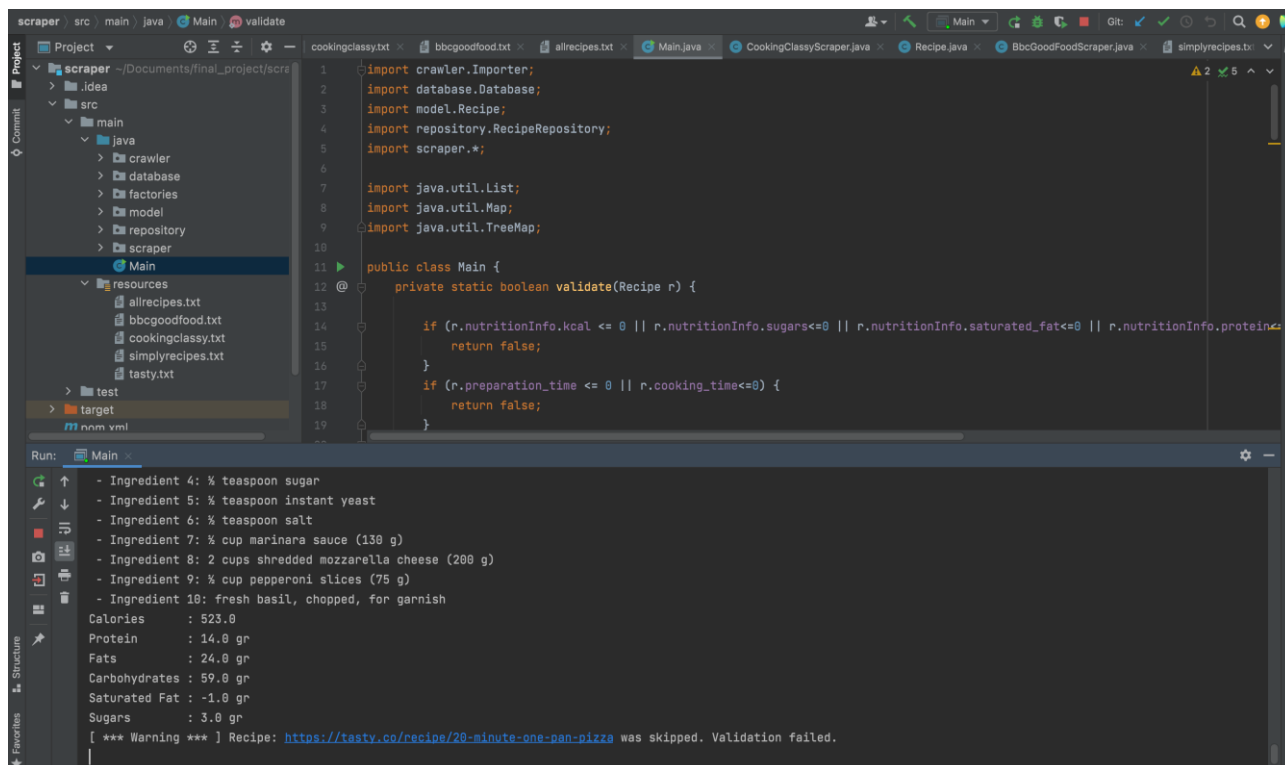
```

1  import crawler.Importer;
2  import database.Database;
3  import model.Recipe;
4  import repository.RecipeRepository;
5  import scraper.*;
6
7  import java.util.List;
8  import java.util.Map;
9  import java.util.TreeMap;
10
11 public class Main {
12     @
13     private static boolean validate(Recipe r) {
14
15         if (r.nutritionInfo.kcal <= 0 || r.nutritionInfo.sugars<=0 || r.nutritionInfo.saturated_fat<=0 || r.nutritionInfo.protein<=0)
16             return false;
17         }
18         if (r.preparation_time <= 0 || r.cooking_time<=0) {
19             return false;
20         }
21         if (r.number_of_ingredients<=0) {
22             return false;
23         }
24         return true;
25     }
26
27     public static void main(String [] args) {
28         ScraperInterface allRecipesScraper = new AllRecipesScraper();
29         ScraperInterface bbcGoodFoodScraper = new BbcGoodFoodScraper();
30         ScraperInterface cookingClassyScraper = new CookingClassyScraper();
31         ScraperInterface simplyRecipesScraper = new SimplyRecipesScraper();
32         ScraperInterface tastyScraper = new TastyScraper();
33
34
35         Importer importer = new Importer();
36         Database db = new Database( autocommit: false, verbose: false);

```

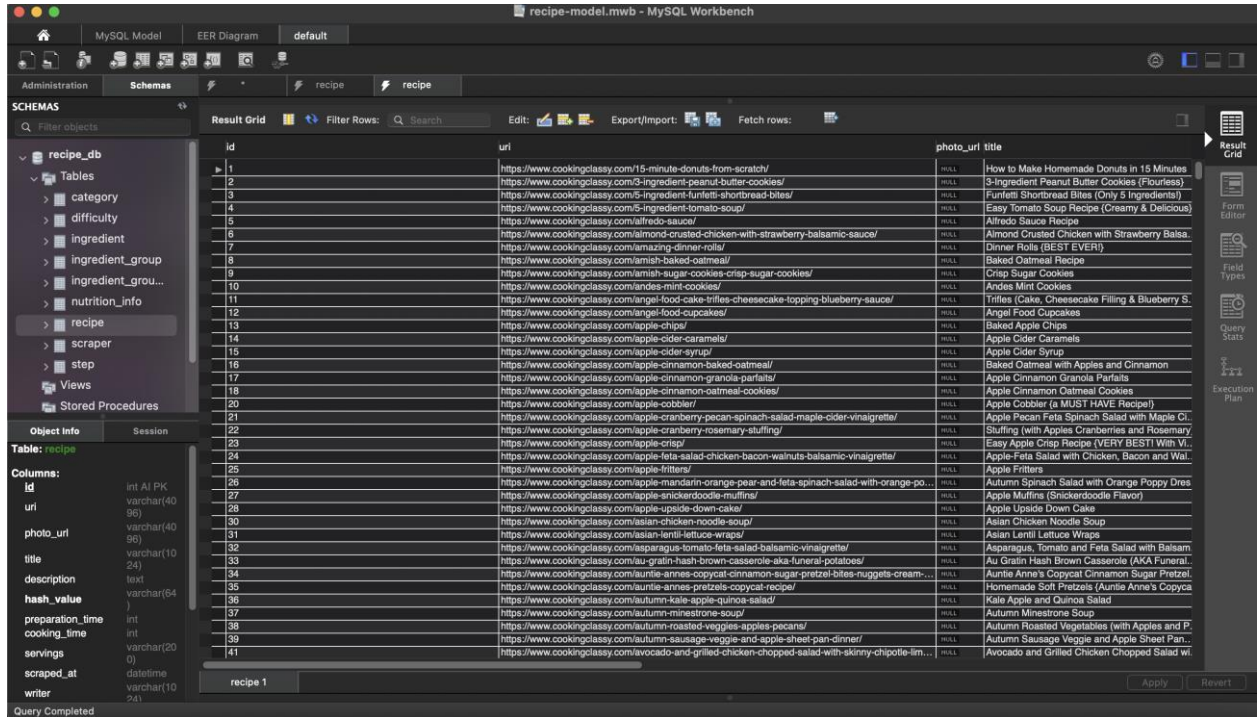
Εικόνα 29: Ο κώδικας της main.java

Όσο γίνεται το scraping:



Εικόνα 30: Κατά τη διάρκεια του scraping

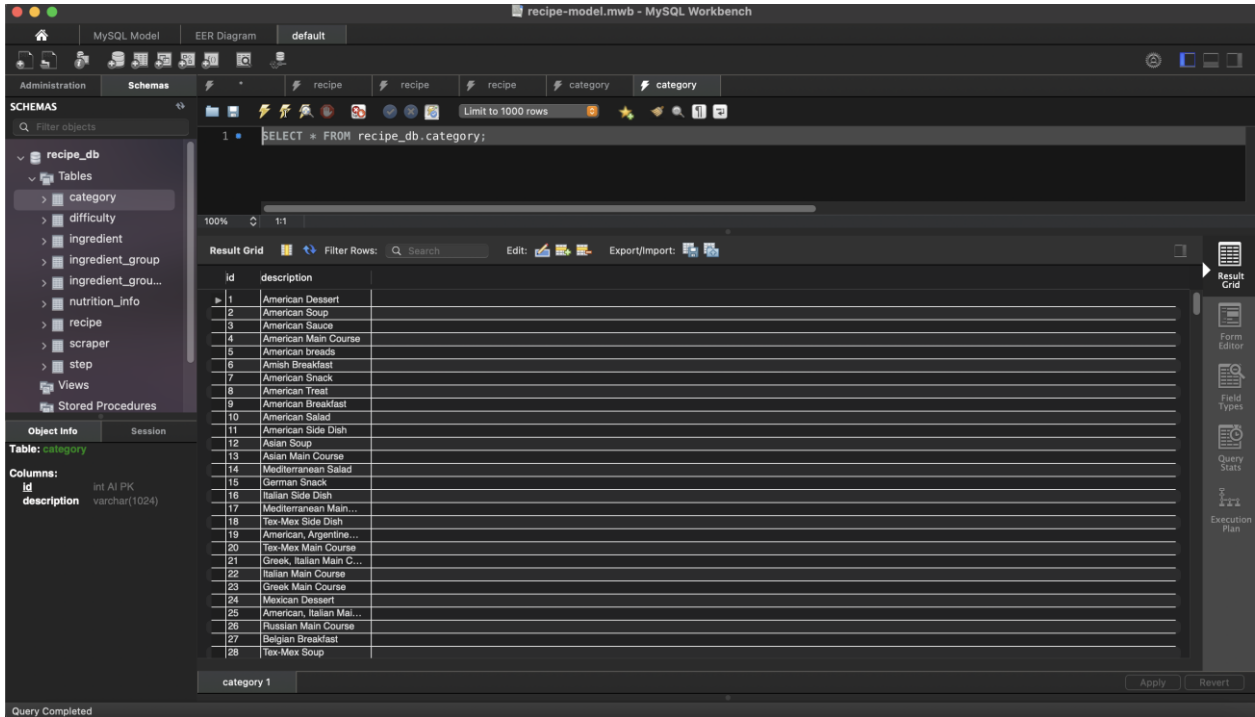
Η εικόνα της βάσης μας μετά το scraping είναι αυτή:



Εικόνα 31: Ο πίνακας “recipe” μετά το scraping

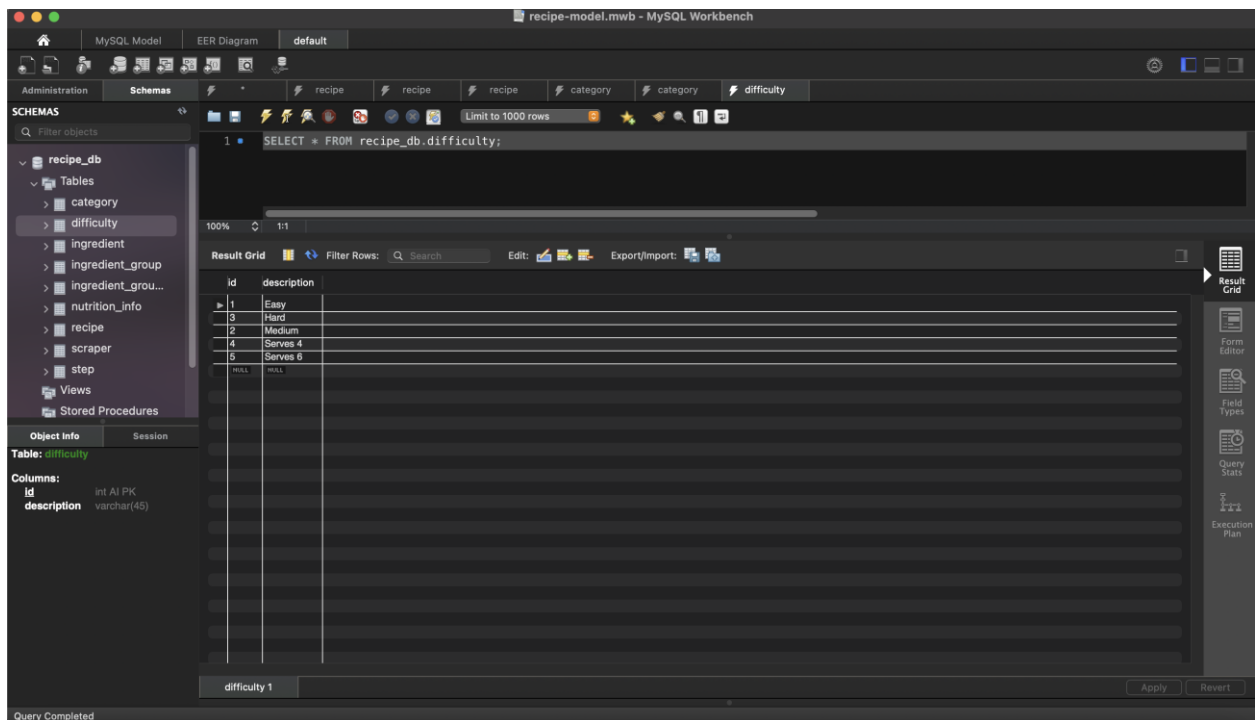
Η εικόνα κάθε πίνακα είναι η ακόλουθη:

Για τον category:



Εικόνα 32: Ο πίνακας “category”

Για τον difficulty:



Εικόνα 33: Ο πίνακας “difficulty”

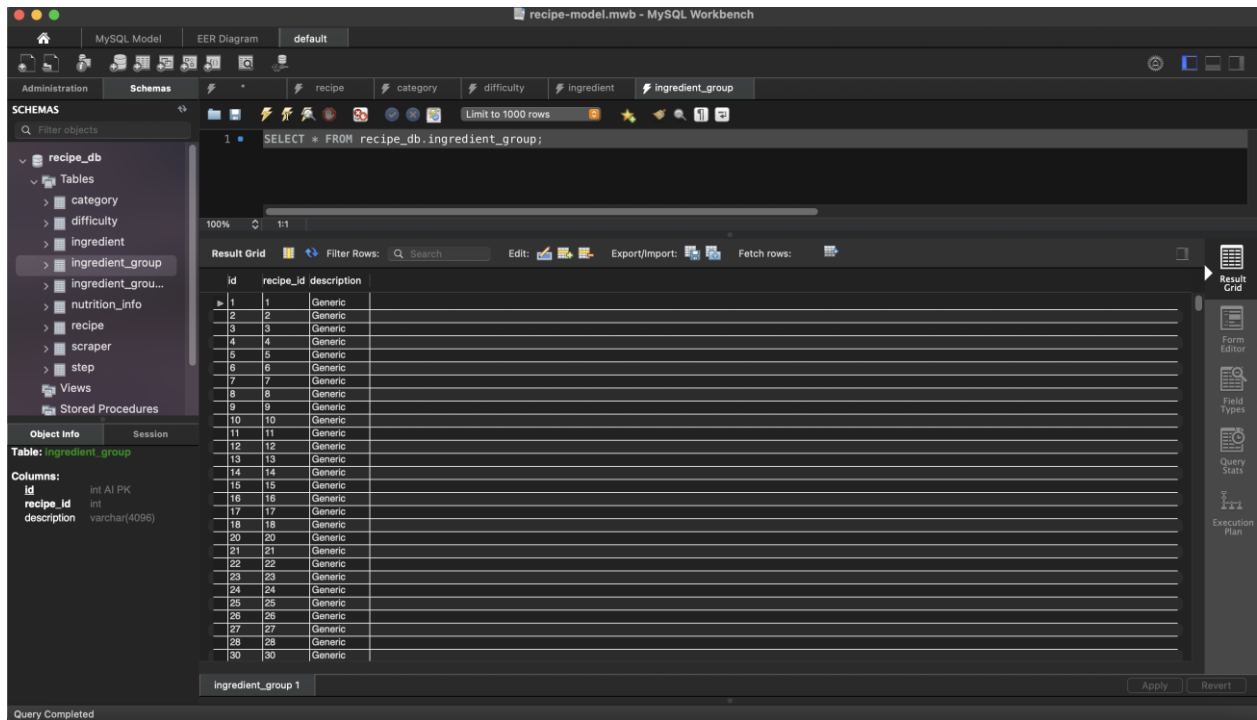
Για τον ingredient:

The screenshot shows a MySQL Workbench interface with a query result grid for the 'ingredient' table. The query executed is 'SELECT * FROM recipe_db.ingredient;'. The result grid displays 28 rows of data, each with an ID, a description, and a hash_value.

id	description	hash_value
1	1 1/4 cups (178g) all-purpose flour (scoop and l...	1c0b2f4de11b6ade2392f29458ae5015
2	2 tsp baking powder	833f7071cb3df1b360c8772865b172f
3	1/4 tsp salt	439e608216710c9967d030b83e56ee2
4	1/2 cup (120 ml) buttermilk*	b46aa408999f9e5468e27a723ae6593
5	1/4 cup (50g) granulated sugar	752430293f1bd8e5527c74225b173e99
6	3 Tbsp (42g) melted butter	67ed1e63d90bb6f668768f6a10ad0c
7	3-4 cups vegetable oil, for frying	e14c311467a1d78262aa8e4cad7c67e
8	1/3 cup (66g) granulated sugar**	c590312f8cc84a3d9574bc3d5296b97
9	1 large egg	0a58208760890430762340509a0eac2
10	1 cup (250g) creamy peanut butter	55685c4c79ac097d56ba158aa0909b9
11	1 cup (100g) granulated sugar	c01eb60d3eaa16dda0841d7003f01c79
12	1 cup (226g) salted butter, cold and diced into 1...	98f54201baa422f6ed6b6304f31f6b
13	3/4 cup (155g) granulated sugar	6f169913d979c0e65f916df158df6a4
14	3/4 tsp almond extract	1f421b9173a7e02c22ef3c81ea3f17
15	2 1/4 cups (318g) all-purpose flour	e1683a16c312f3189e9312b053e9277a
16	3 1/2 Tbsp (42g) nonpareils sprinkles, divided	3fa12ac2729ac59540d0cd6c3cc6348c
17	Canned plum tomatoes or san marzano tomatoes	a0c9896f9a103aae4e24fedde4ccab1
18	Butter	e5961b0baa369b1c565601db04d823c
19	Garlic	26f55ab265d057f9adfae265893c27
20	Chicken broth	8879211612a6e97977a5d10618377e2
21	Fresh Basil	681052b1bc81bc23f114466761199a14
22	Heavy cream (optional)	6880f39c28564b42542967200aee3c3
23	2 (28 oz) cans Whole Plum or San Marzano To...	2b3b1c84df9dec610c3de682a1f6fd
24	4 Tbsp salted butter	57e57abc1429b99ce7908837d0edf6a
25	3 cloves garlic - minced	c2a9ef052058c3f8a20d446db0e8
26	2 cups Progresso™ Chicken Stock	58cc7eb330723c383858e96999f9b
27	1/4 cup chopped fresh basil	a7fae93a114592d8a4e2b0506fad5c3
28	Salt and freshly ground black pepper (optional)	103a3703091914ee25574ea1756e7a83

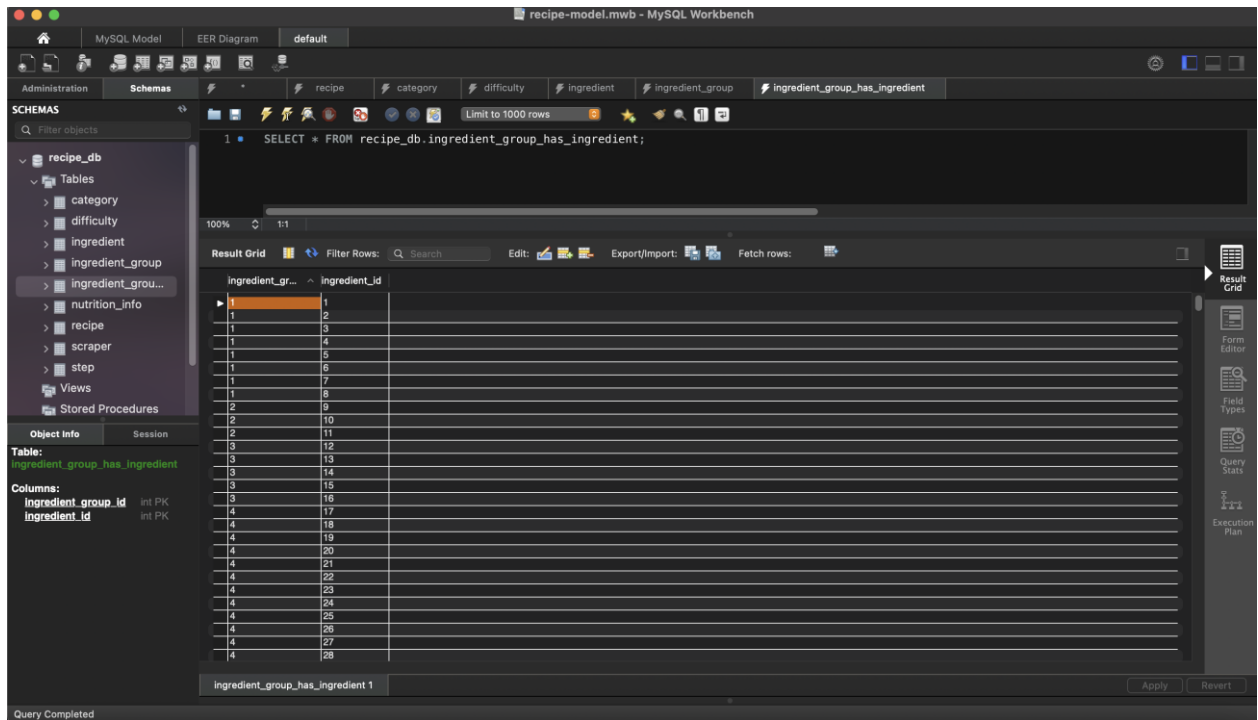
Εικόνα 34: Ο πίνακας “ingredient”

Για τον ingredient_group:



Εικόνα 35: Ο πίνακας “ingredient_group”

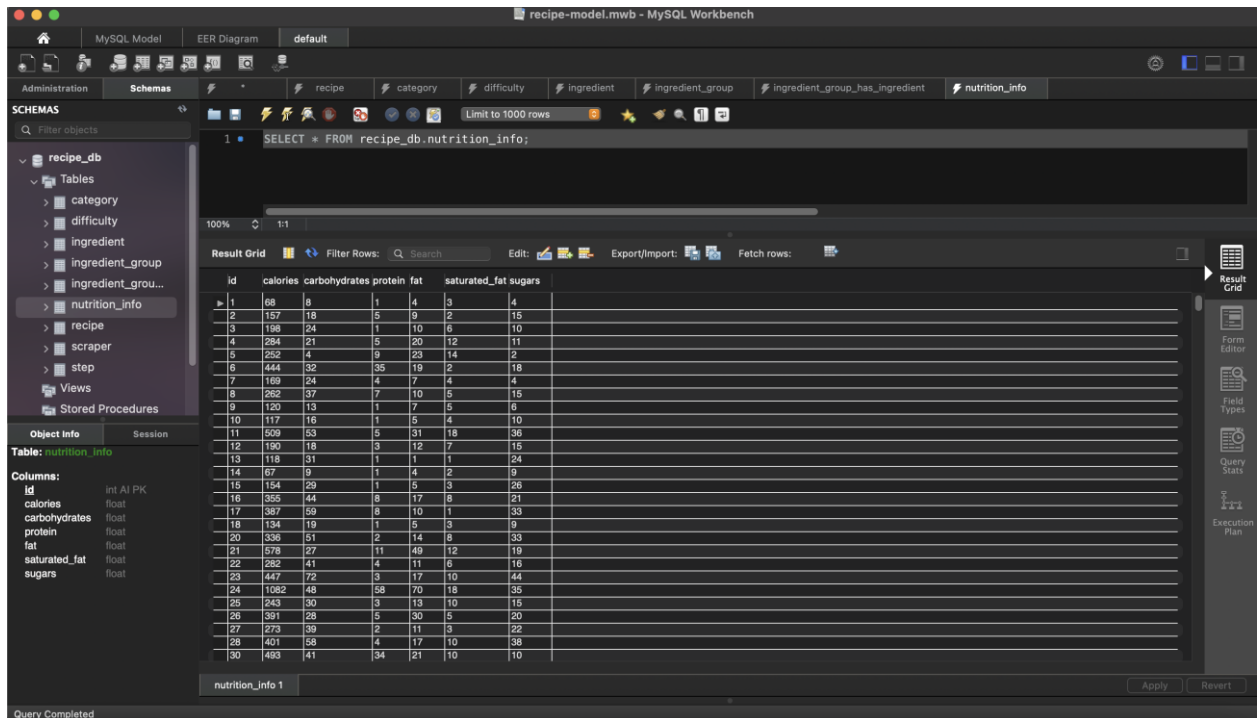
Για τον ingredient_group_has_ingredient:



Εικόνα 36: Ο πίνακας “ingredient_group_has_ingredient”

Για τον nutrition_info:

Recipes Web Scraper

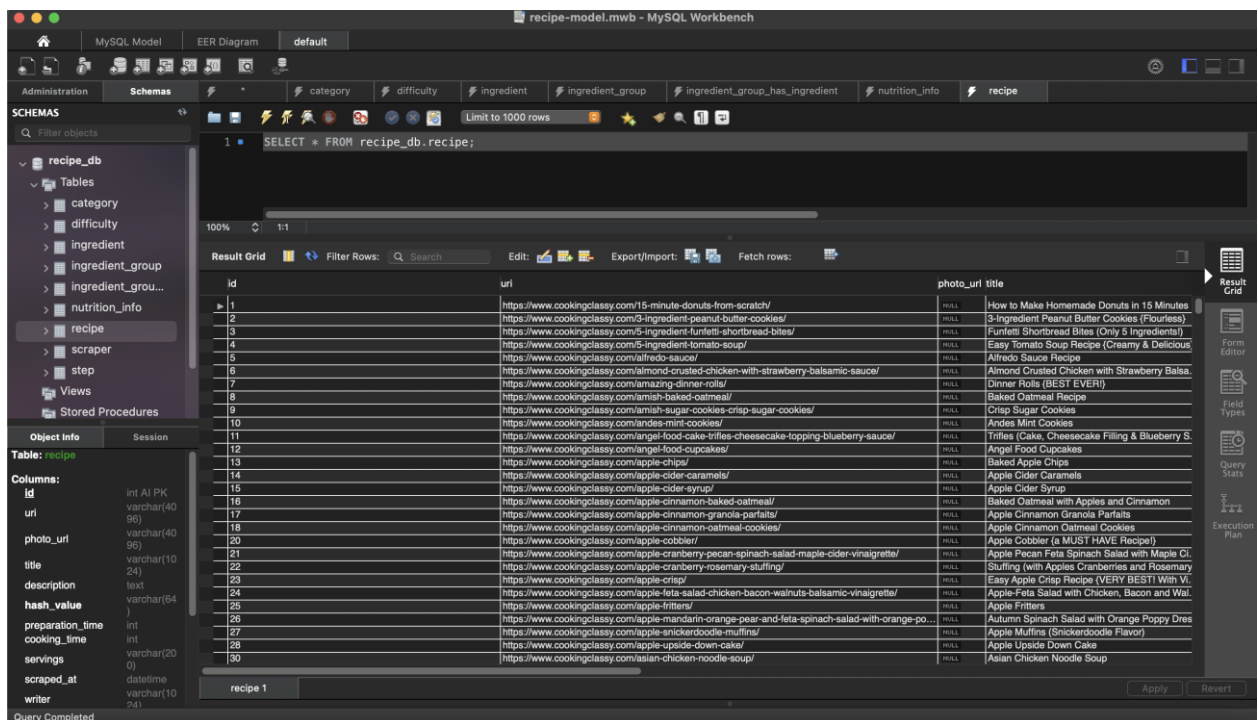


Query: `SELECT * FROM recipe_db.nutrition_info;`

id	calories	carbohydrates	protein	fat	saturated_fat	sugars
1	68	8	1	4	3	4
2	157	18	5	9	2	15
3	198	24	1	10	6	10
4	284	21	5	20	12	11
5	252	4	9	23	14	2
6	444	32	35	19	2	18
7	169	24	4	7	4	4
8	262	37	7	10	5	15
9	120	13	1	7	5	6
10	117	16	1	5	4	10
11	509	53	5	31	19	33
12	190	18	3	12	7	15
13	118	31	1	1	1	24
14	67	9	1	4	2	9
15	154	29	1	5	3	26
16	355	44	8	17	8	21
17	387	59	8	10	1	33
18	134	19	1	5	3	9
20	336	51	2	14	8	33
21	578	27	11	49	12	19
22	262	41	4	11	8	16
23	447	72	3	17	10	44
24	1082	48	58	70	18	35
25	243	30	3	13	10	15
26	391	28	5	30	5	20
27	273	39	2	11	3	22
28	401	59	4	17	10	39
30	493	41	34	21	10	10

Εικόνα 37: Ο πίνακας “nutrition_info”

Για τον recipe:

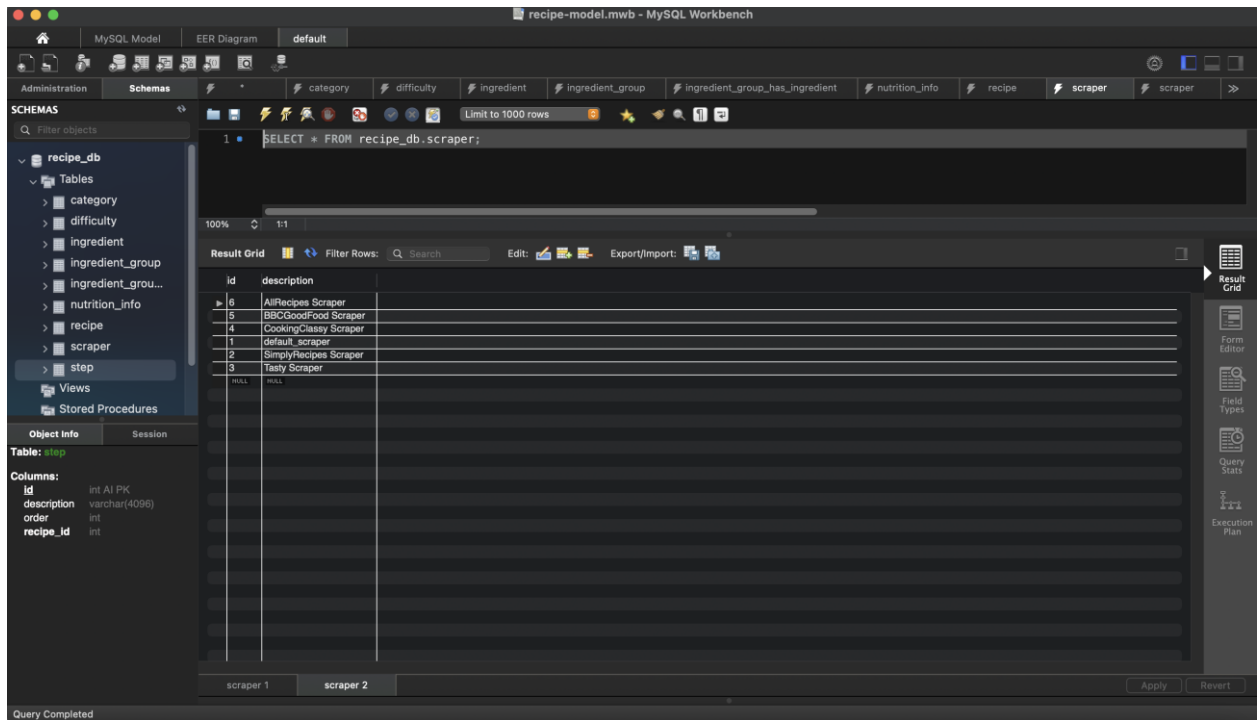


Query: `SELECT * FROM recipe_db.recipe;`

id	uri	photo_url	title
1	https://www.cookingclassy.com/15-minute-donuts-from-scratch/	NULL	How to Make Homemade Donuts in 15 Minutes
2	https://www.cookingclassy.com/3-ingredient-peanut-butter-cookies/	NULL	3-Ingredient Peanut Butter Cookies (Flourless)
3	https://www.cookingclassy.com/5-ingredient-funfetti-shortbread-bites/	NULL	Funfetti Shortbread Bites (Only 5 Ingredients!)
4	https://www.cookingclassy.com/5-ingredient-tomato-soup/	NULL	Easy Tomato Soup Recipe (Creamy & Delicious)
5	https://www.cookingclassy.com/allreds-sauce/	NULL	Allred's Sauce Recipe
6	https://www.cookingclassy.com/almond-crustied-chicken-with-strawberry-balsamic-sauce/	NULL	Almond Crustied Chicken with Strawberry Balsamic Sauce
7	https://www.cookingclassy.com/amazing-dinner-rolls/	NULL	Dinner Rolls (BEST EVER!)
8	https://www.cookingclassy.com/amish-baked-oatmeal/	NULL	Baked Oatmeal Recipe
9	https://www.cookingclassy.com/amish-sugar-cookies-crisp-sugar-cookies/	NULL	Crisp Sugar Cookies
10	https://www.cookingclassy.com/andes-mini-cookies/	NULL	Andes Mint Cookies
11	https://www.cookingclassy.com/angel-food-cake-frites-cheesecake-topping-blueberry-sauce/	NULL	Frites (Chips, Cheesecake Filling & Blueberry S...
12	https://www.cookingclassy.com/angel-food-cupcakes/	NULL	Angel Food Cupcakes
13	https://www.cookingclassy.com/apple-chips/	NULL	Baked Apple Chips
14	https://www.cookingclassy.com/apple-cider-caramels/	NULL	Apple Cider Caramels
15	https://www.cookingclassy.com/apple-cider-syrup/	NULL	Apple Cider Syrup
16	https://www.cookingclassy.com/apple-cinnamon-baked-oatmeal/	NULL	Baked Oatmeal with Apples and Cinnamon
17	https://www.cookingclassy.com/apple-cinnamon-granola-parfaits/	NULL	Apple Cinnamon Granola Parfaits
18	https://www.cookingclassy.com/apple-cinnamon-oatmeal-cookies/	NULL	Apple Cinnamon Oatmeal Cookies
20	https://www.cookingclassy.com/apple-cobbler/	NULL	Apple Cobbler (a MUST HAVE Recipe!)
21	https://www.cookingclassy.com/apple-cranberry-pecan-spinach-salad-maple-cider-vinaigrette/	NULL	Apple Pecan Feta Spinach Salad with Maple Ci...
22	https://www.cookingclassy.com/apple-cranberry-rosemary-stuffing/	NULL	Stuffing with Apples Cranberries and Rosemary
23	https://www.cookingclassy.com/apple-crisp/	NULL	Easy Apple Crisp Recipe (VERY BEST! With Vi...
24	https://www.cookingclassy.com/apple-feta-salad-chicken-bacon-walnut-balsamic-vinaigrette/	NULL	Apple-Feta Salad with Chicken, Bacon and Wal...
25	https://www.cookingclassy.com/apple-fritters/	NULL	Apple Fritters
26	https://www.cookingclassy.com/apple-mandarin-orange-pear-and-feta-spinach-salad-with-orange-po...	NULL	Autumn Spinach Salad with Orange Poppy Dres...
27	https://www.cookingclassy.com/apple-snickerdoodle-muffins/	NULL	Apple Muffins (Snickerdoodle Flavor)
28	https://www.cookingclassy.com/apple-upside-down-cake/	NULL	Apple Upside Down Cake
30	https://www.cookingclassy.com/asian-chicken-noodle-soup/	NULL	Asian Chicken Noodle Soup

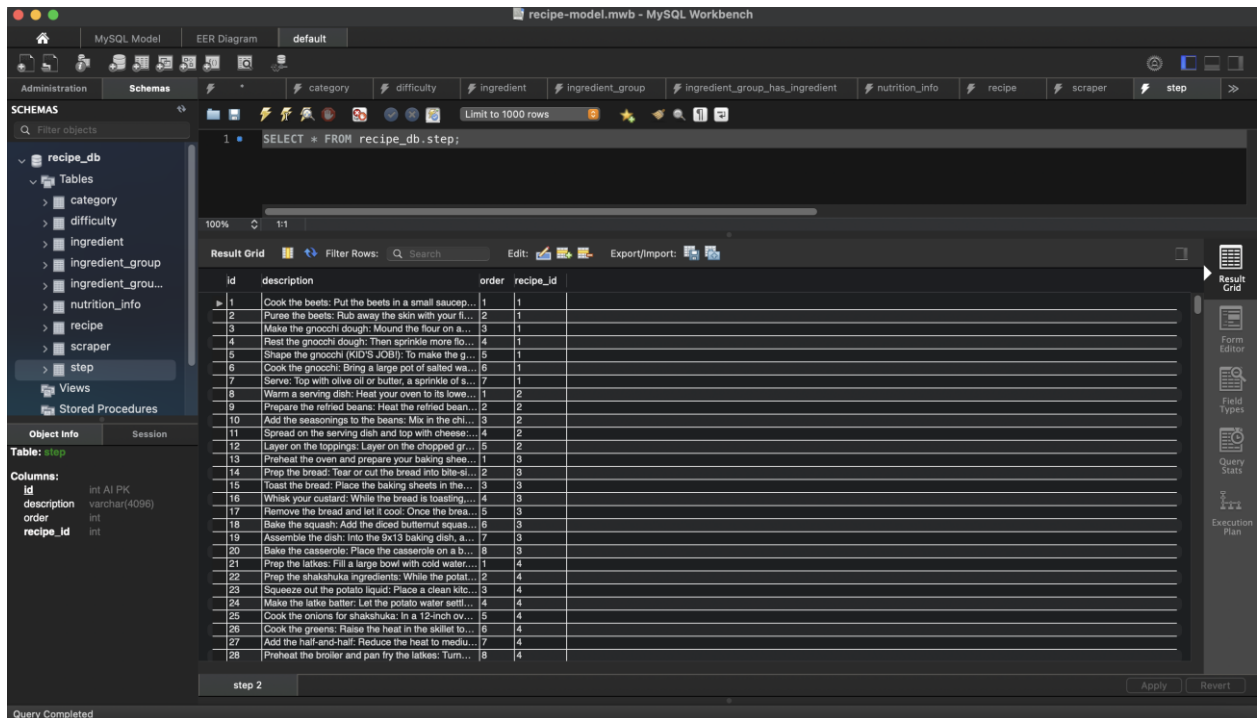
Εικόνα 38: Ο πίνακας “recipe”

Για τον scraper:



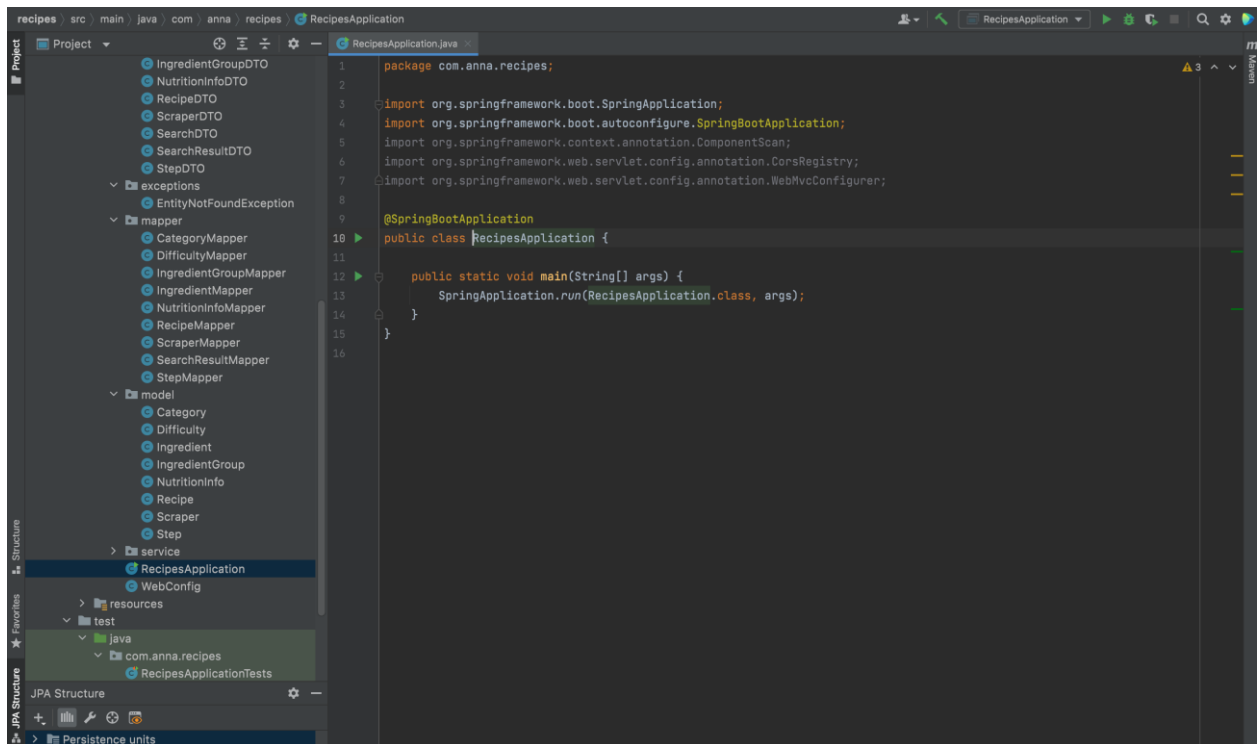
Εικόνα 39: Ο πίνακας “scrapper”

Για τον step:



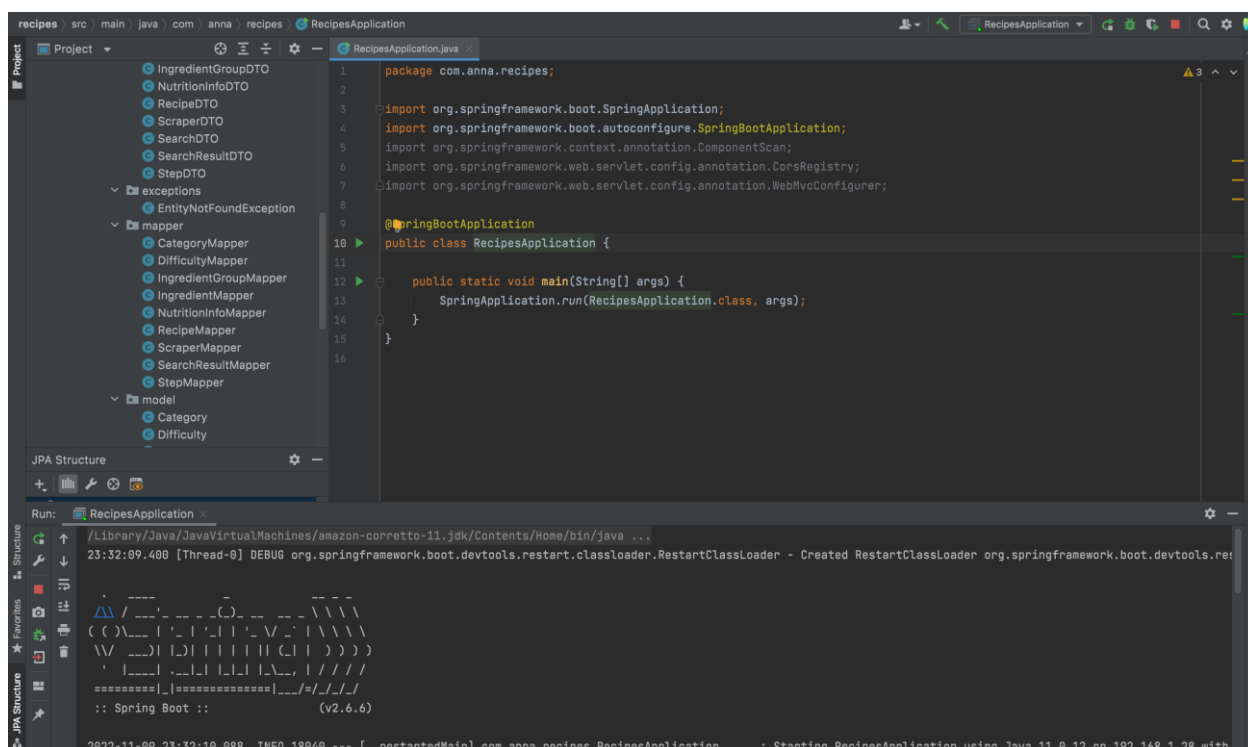
Εικόνα 40: Ο πίνακας “step”

Στη συνέχεια πηγαίνουμε στο IntelliJ IDEA και τρέχουμε το project “Recipes”, δηλαδή το backend.



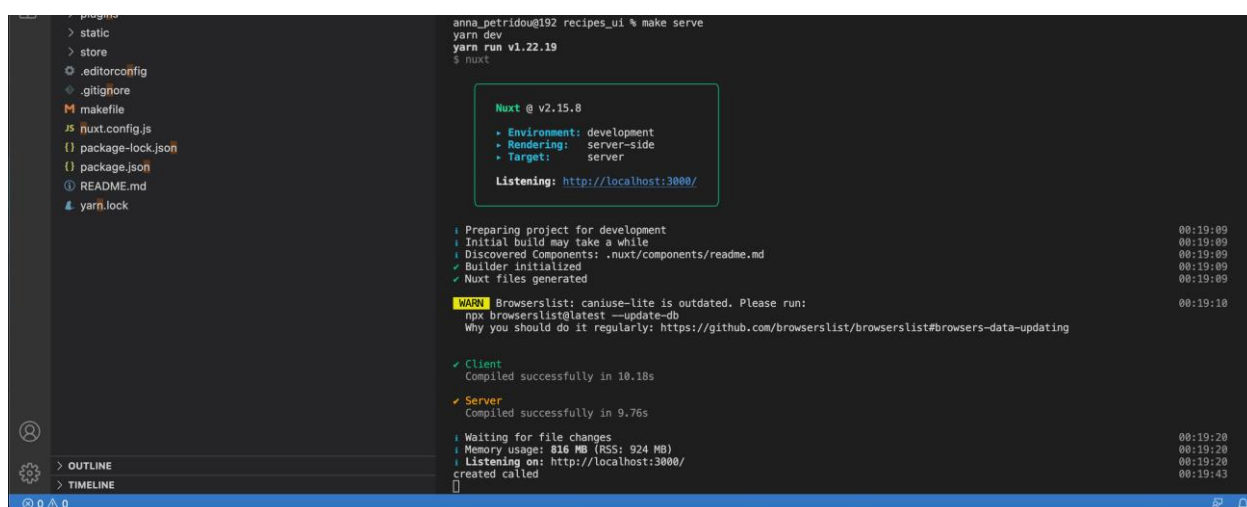
Εικόνα 41: Το αρχείο κώδικα “RecipesApplication.java”

Εκκινείται το spring boot όπως φαίνεται παρακάτω:



Εικόνα 42: Το terminal όταν εκκινείται το spring boot

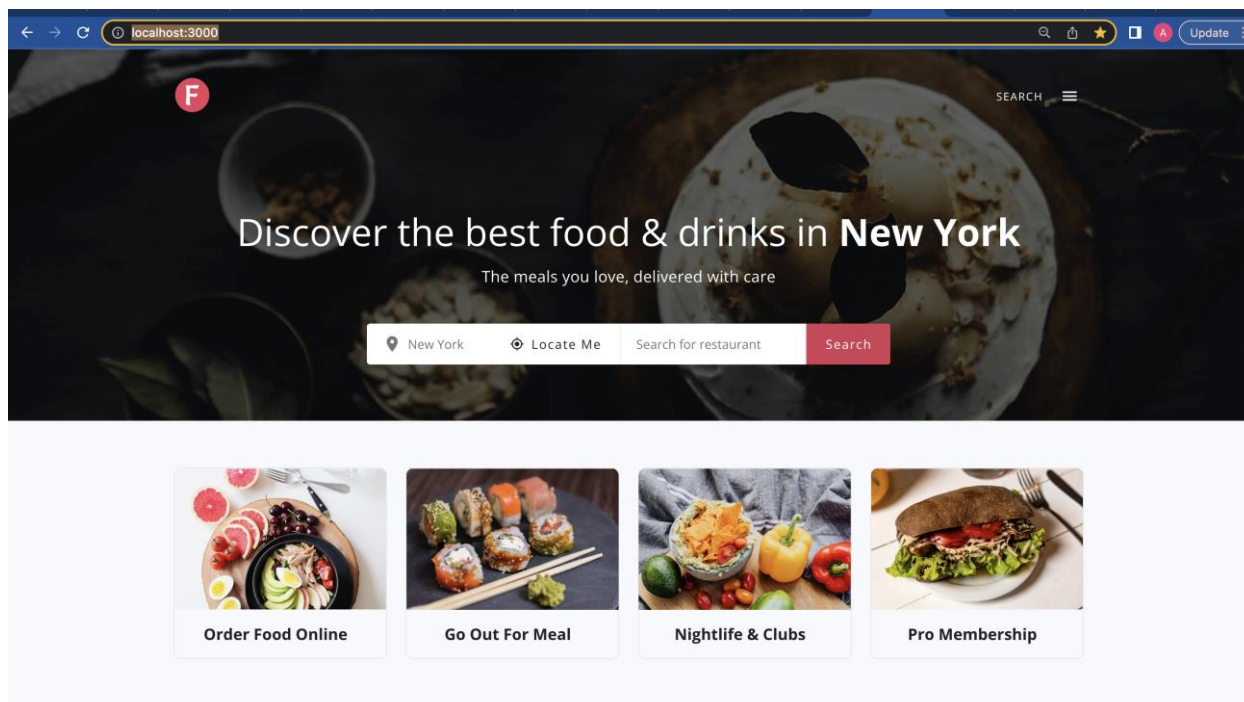
Έπειτα πάμε στο Visual Studio Code για να τρέξουμε το front end. Ανοίγουμε το φάκελο “recipies_ui” μέσα από το Visual Studio Code και τρέχουμε “make serve” στο terminal.



Εικόνα 43: Make serve στο visual studio code

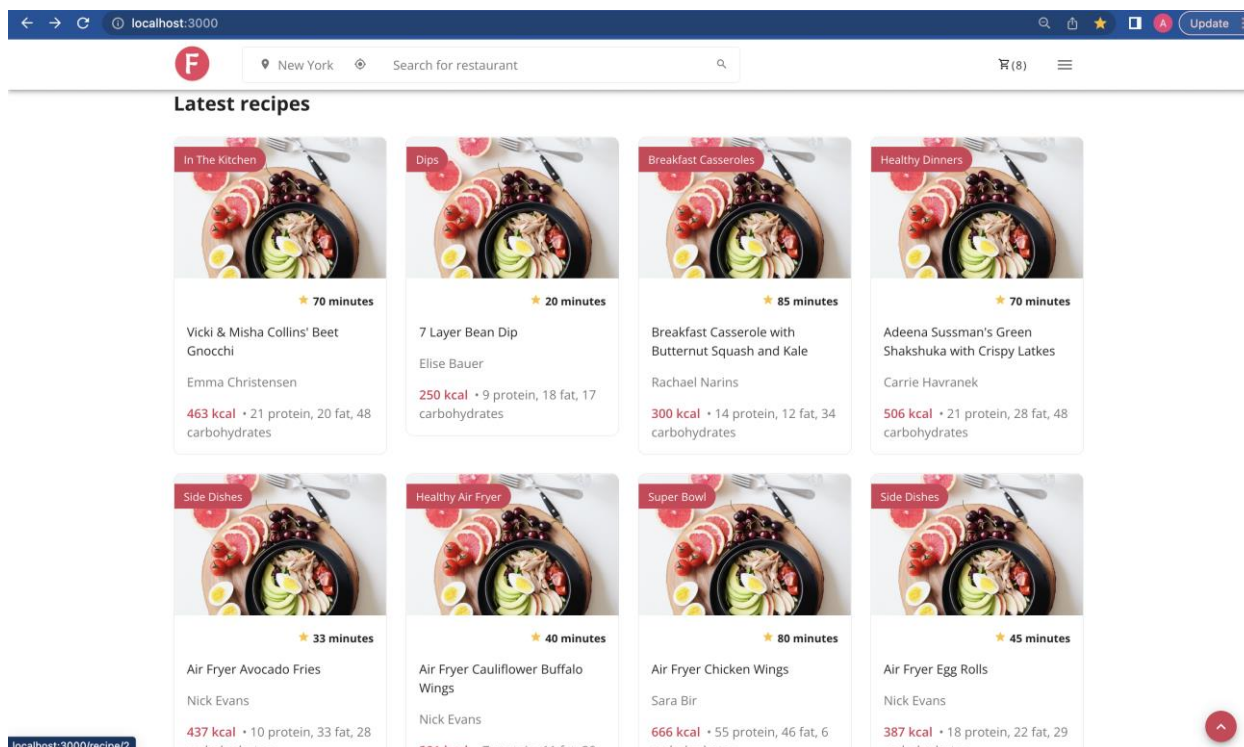
Στο σημείο αυτό, μπορούμε να πάμε στην ιστοσελίδα μας στο <http://localhost:3000/> και να δούμε την τελική εικόνα του user interface μας.

Πιο συγκεκριμένα, η αρχική οθόνη είναι αυτή:



Εικόνα 44: Η homepage σελίδα του user interface


Αν κάνουμε scroll down, μπορούμε να δούμε τις πιο πρόσφατες συνταγές:



Εικόνα 45: Η αρχική σελίδα μετά από scroll down

Μια από τις πιθανές επεκτάσεις που θα μπορούσαν να γίνουν μελλοντικά στην εφαρμογή, είναι η προσθήκη φωτογραφίας για κάθε συνταγή. Το αντίστοιχο πεδίο έχει ήδη δημιουργηθεί στη βάση δεδομένων.

Σε κάθε μία από αυτές τις συνταγές μπορούμε να παρατηρήσουμε και διάφορες πληροφορίες τους, όπως την κατηγορία, τον τίτλο, το συγγραφέα, τις θερμίδες, τις διατροφικές πληροφορίες και το χρόνο εκτέλεσης.



Healthy Dinners

★ 70 minutes

Adeena Sussman's Green Shakshuka with Crispy Latkes

Carrie Havranek

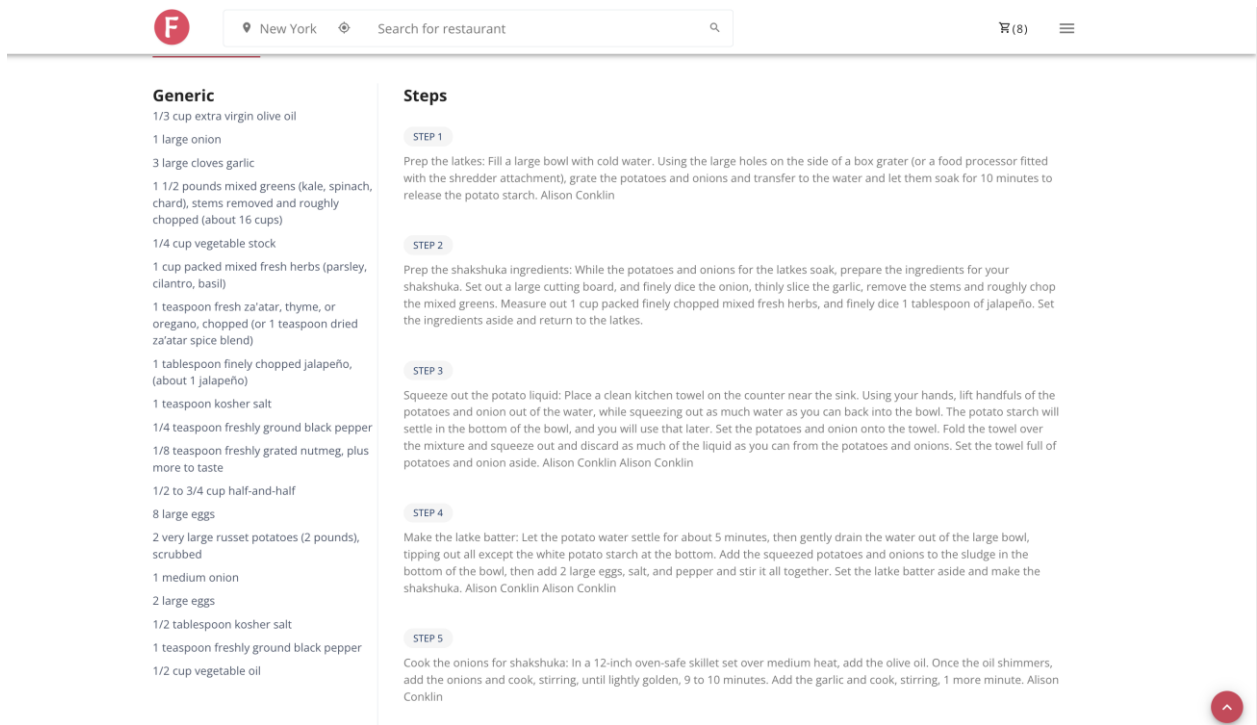
506 kcal • 21 protein, 28 fat, 48 carbohydrates

Εικόνα 46: Αποτέλεσμα ενδεικτικής συνταγής

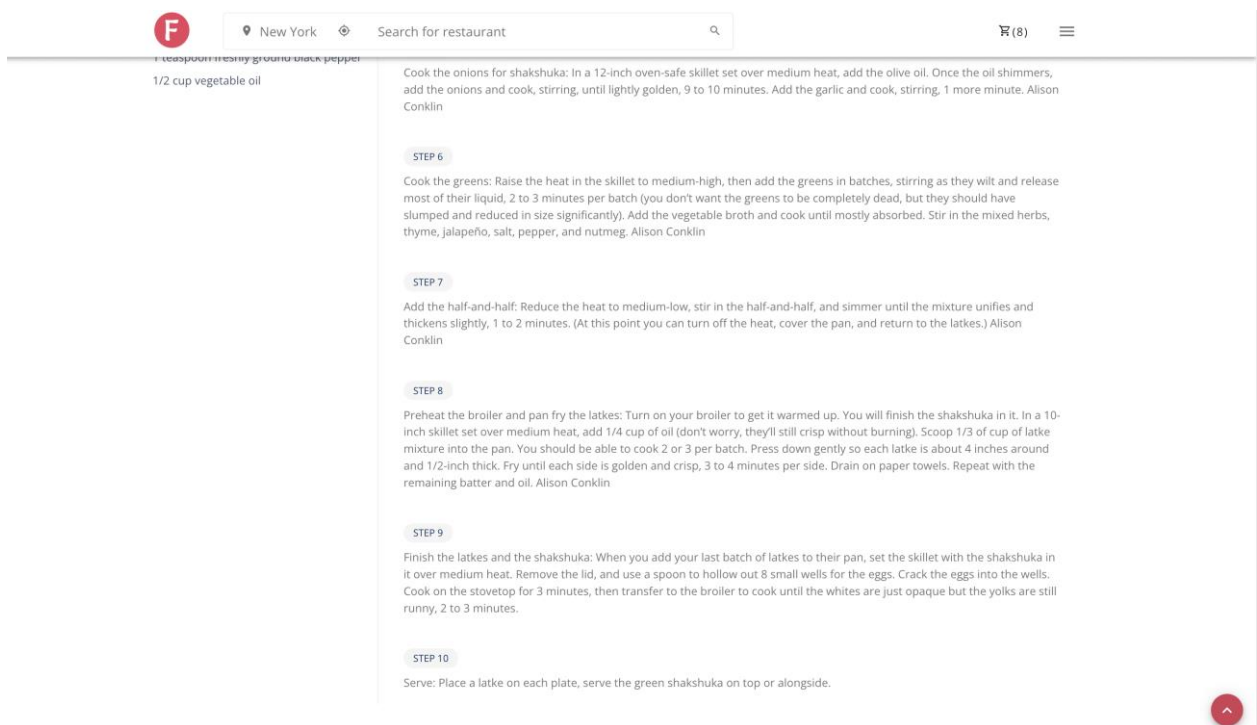
Αν κάνουμε κλικ σε κάποια από αυτές, θα μας ανοίξει στην παρακάτω μορφή, όπου μπορούμε να διαβάσουμε όλη την πληροφορία της συνταγής:

The image shows a screenshot of a recipe page. At the top, there is a navigation bar with a red 'F' logo, a search bar containing 'New York' and 'Search for restaurant', and a shopping cart icon with '(8)' items. Below the navigation bar is a breadcrumb trail: 'Home > Recipes > Healthy Dinners > Adeena Sussman's Green Shakshuka with Crispy Latkes'. The main content area features a large photograph of a wooden cutting board with sliced grapefruit, cherry tomatoes, and a black bowl containing a shakshuka dish with green vegetables and eggs. Below the photo, the title 'Adeena Sussman's Green Shakshuka with Crispy Latkes' is displayed in bold, followed by 'Medium' and a star icon with '70 minutes'. The author 'Carrie Havranek' is listed below the title. Underneath, it says 'Preparation: 30, Cooking: 40' and 'Yield: 4'. A short description follows: 'Shakshuka with mixed greens, herbs, onions, garlic, and spices—and eggs that cook right in the pan. Make it for dinner with crispy latkes, and eat the rest for breakfast the next day!'. At the bottom, there are three tabs: 'How To Make' (selected), 'Nutritional Info', and 'Comments'. Under the 'How To Make' tab, the 'Generic' section is active, showing '1/3 cup extra virgin olive oil'. The 'Steps' section is also visible but empty.

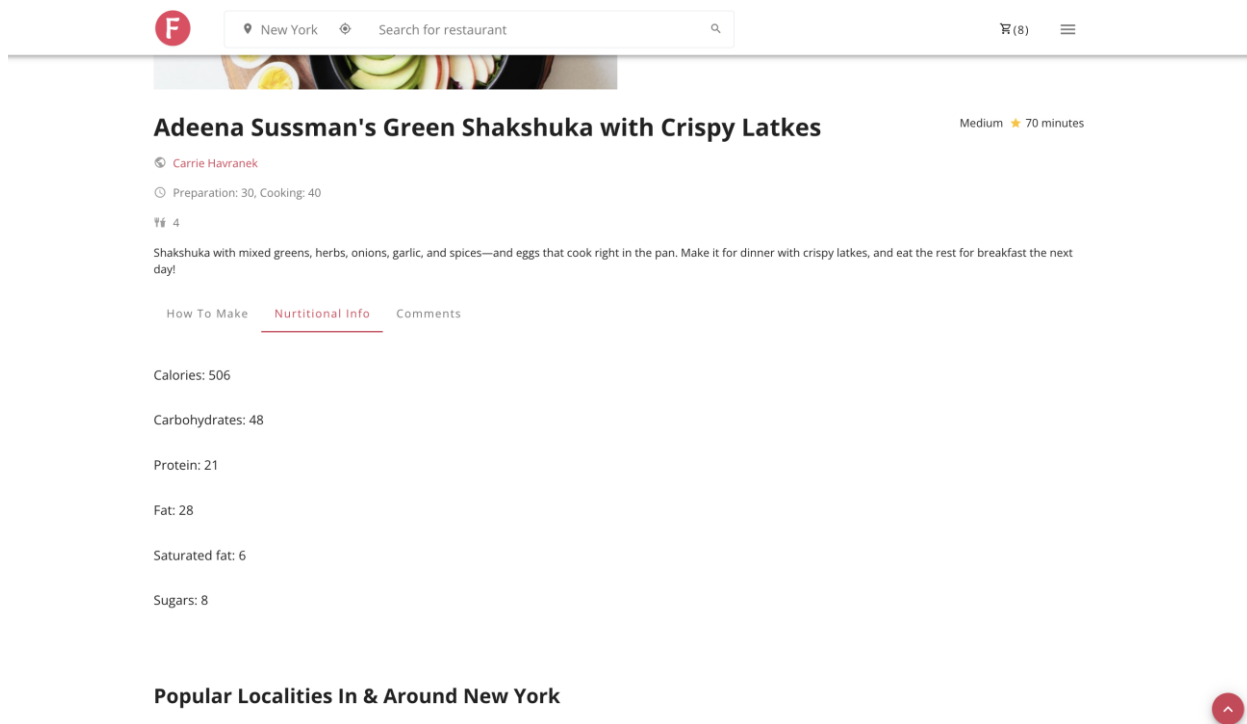
Εικόνα 47: Σελίδα συνταγής με όλη τη σχετική πληροφορία



Εικόνα 48: Σελίδα συνταγής μετά από scroll down

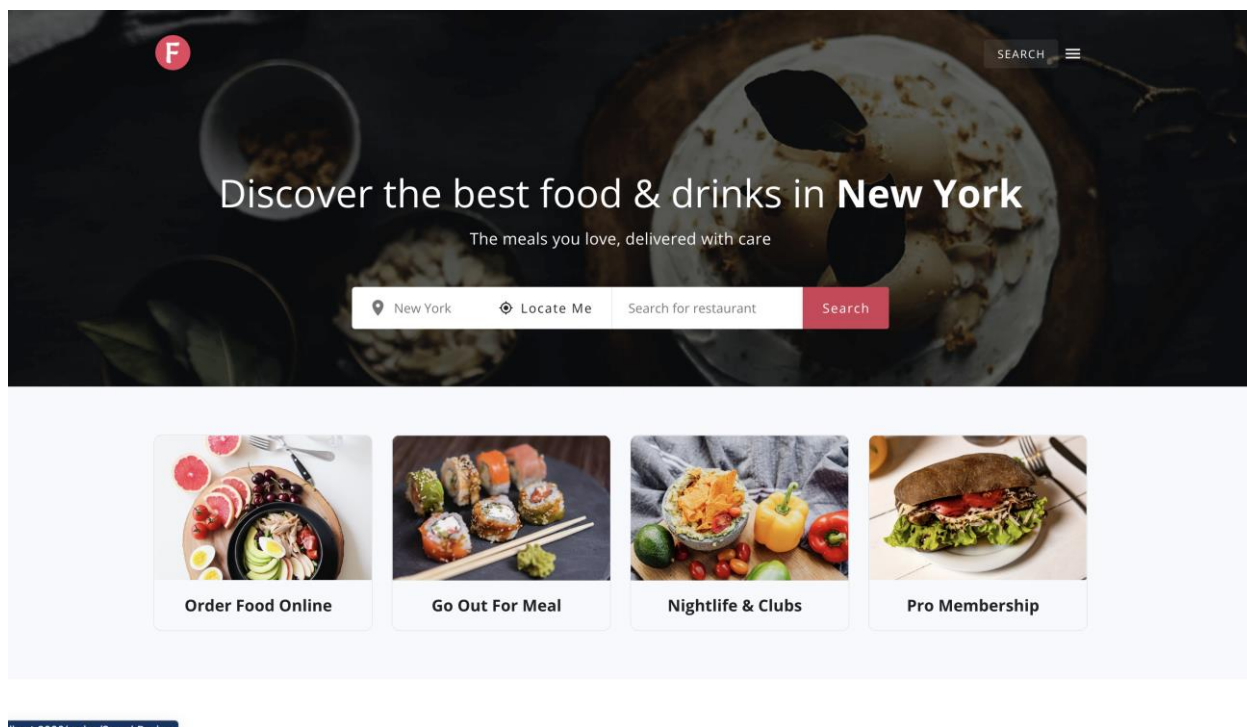


Εικόνα 49: Σελίδα συνταγής μετά από περισσότερο scroll down



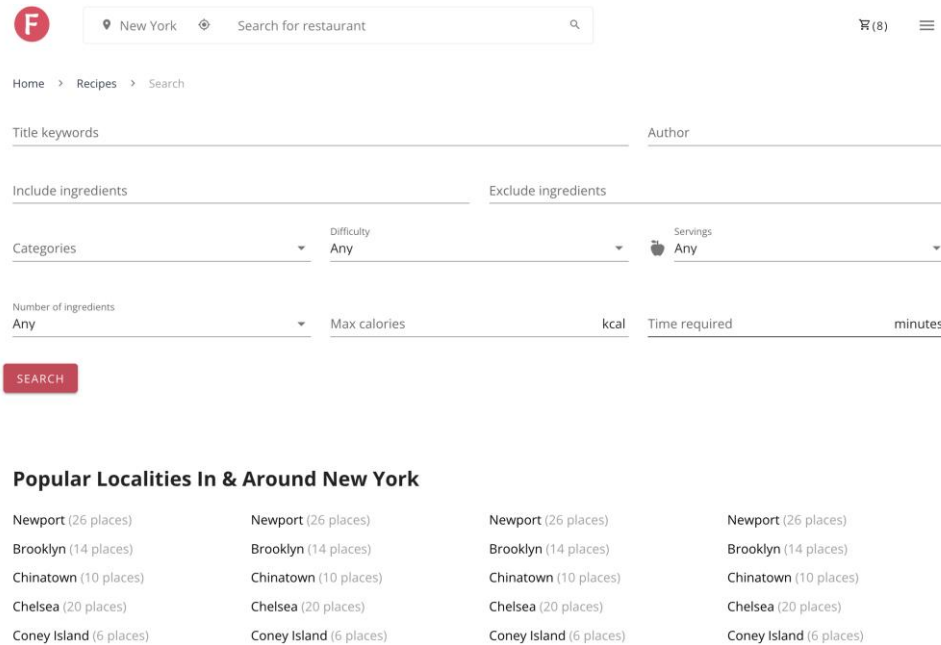
Εικόνα 50: Το tab του “nutritional info”

Μπορούμε να πατήσουμε το “Search”(αναζήτηση) πάνω δεξιά για να πραγματοποιήσουμε αναζητήσεις.



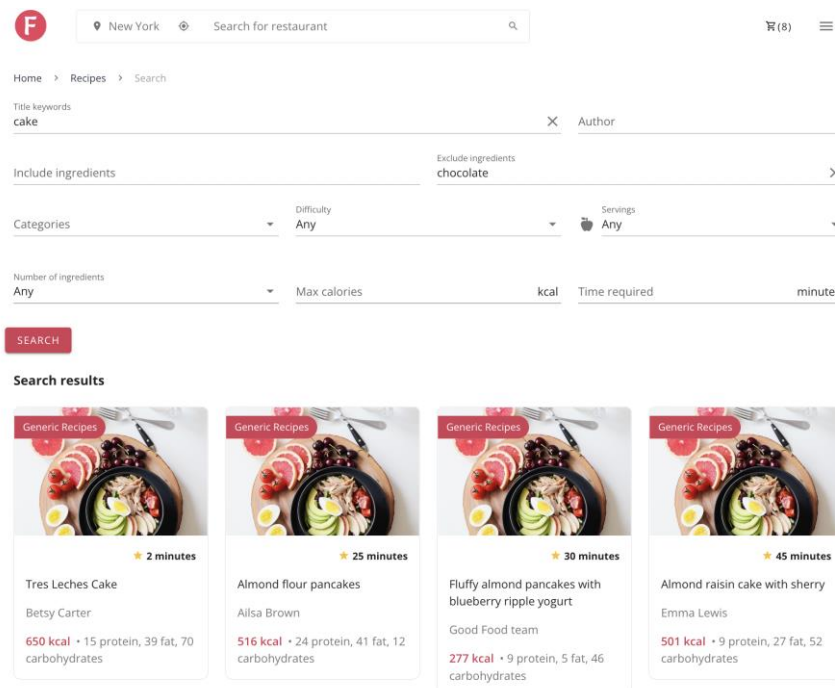
Εικόνα 51: Κουμπί αναζήτησης στην αρχική σελίδα

Μας μεταφέρει στη σελίδα με τα φίλτρα από όπου μπορούμε να επιλέξουμε όσα φίλτρα θέλουμε για να περιορίσουμε τα αποτελέσματα και να κάνουμε την αναζήτησή μας πιο στοχευμένη.



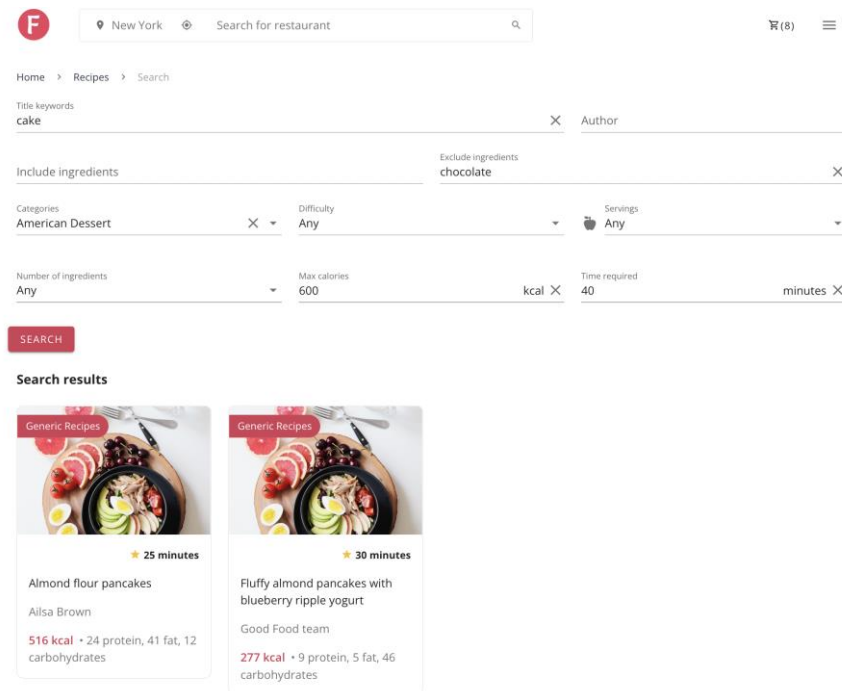
Εικόνα 52: Σελίδα με φίλτρα

Για παράδειγμα, έστω ότι βάζουμε τη λέξη κλειδί “cake” στον τίτλο και αποκλείουμε από τα συστατικά τη σοκολάτα, τα πρώτα αποτελέσματα που θα πάρουμε είναι αυτά.



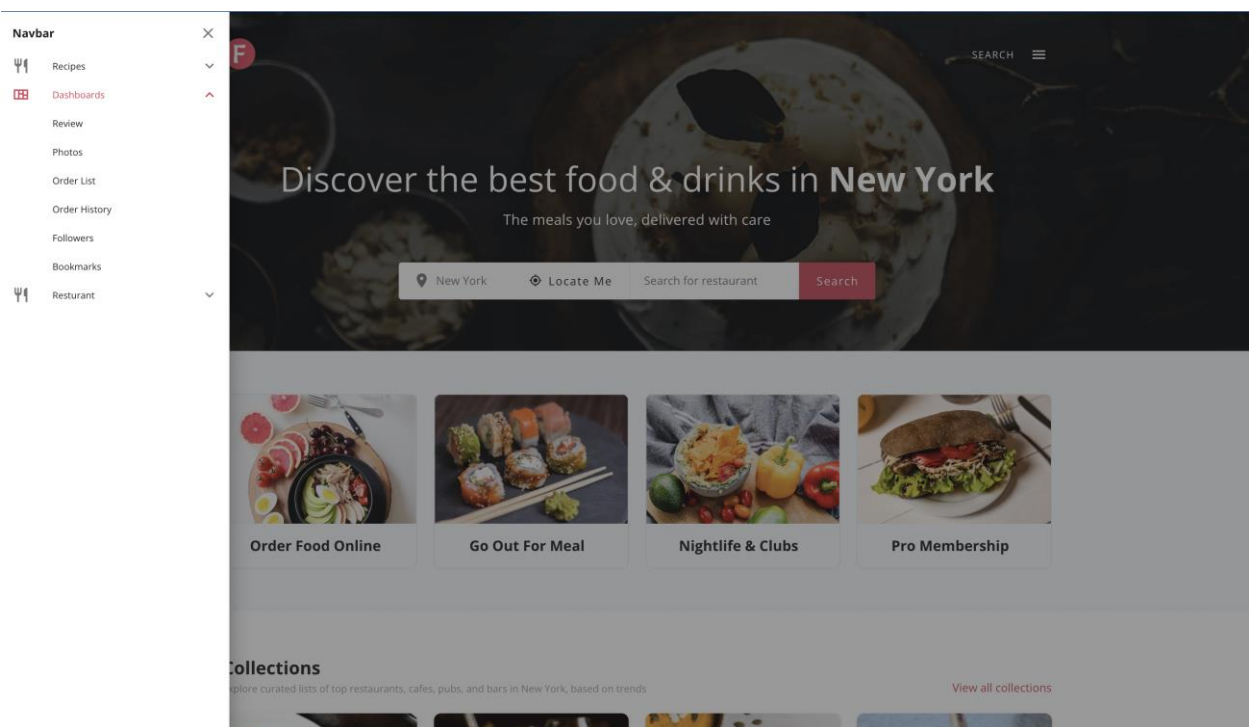
Εικόνα 53: Αποτελέσματα φίλτρων

Αν στη συνέχεια προσθέσουμε κατηγορία, max θερμίδων και χρόνο εκτέλεσης, τα αποτελέσματα περιορίζονται ακόμα περισσότερο.

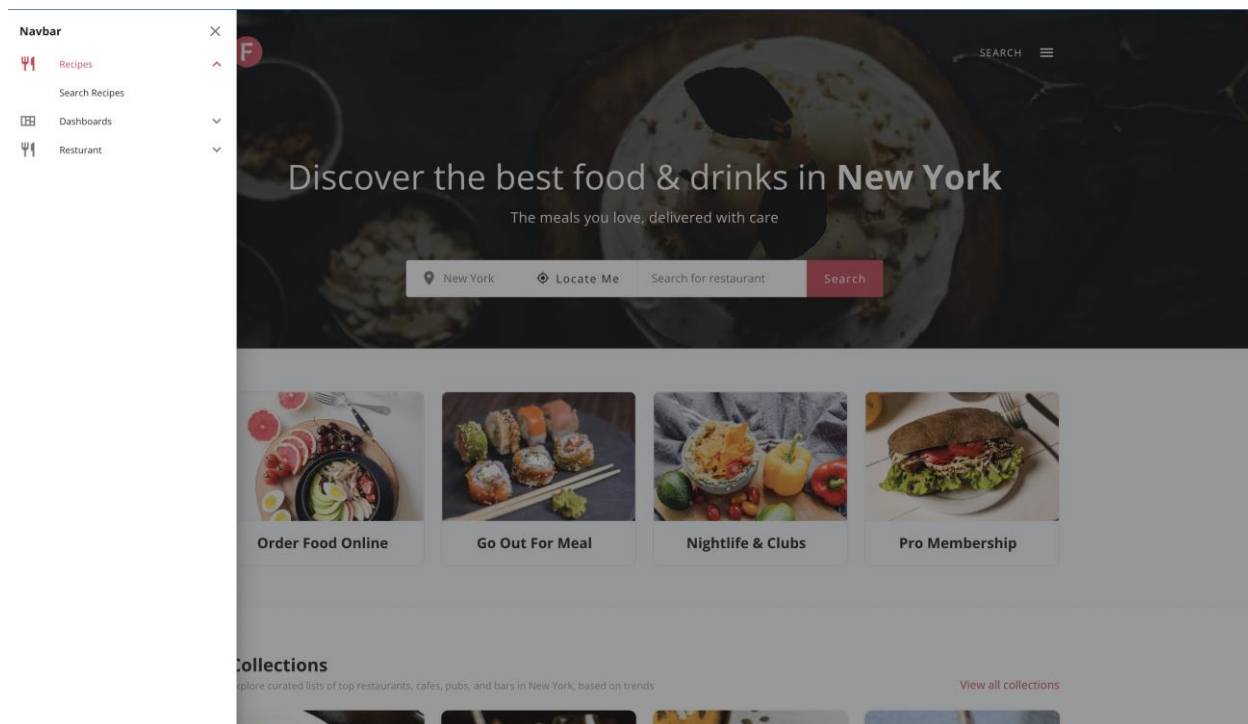


Εικόνα 54: Αποτελέσματα μετά από περισσότερα φίλτρα

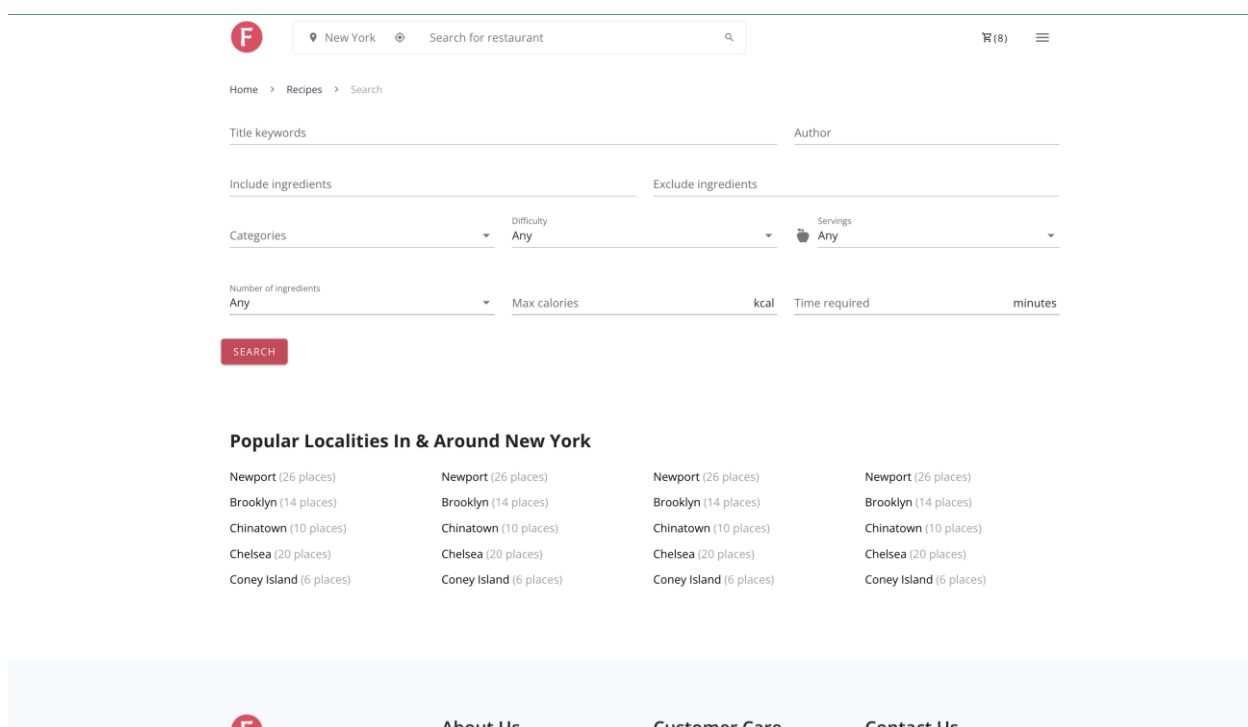
Ο χρήστης μπορεί να καταλήξει στην αναζήτηση είτε από το “Search” πάνω δεξιά στην αρχική οθόνη, είτε ανοίγοντας το tab με τις επιλογές και επιλέγοντας “Recipes” και μετά “Search Recipes”



Εικόνα 55: Τρόπος αναζήτησης από το μενού στα αριστερά

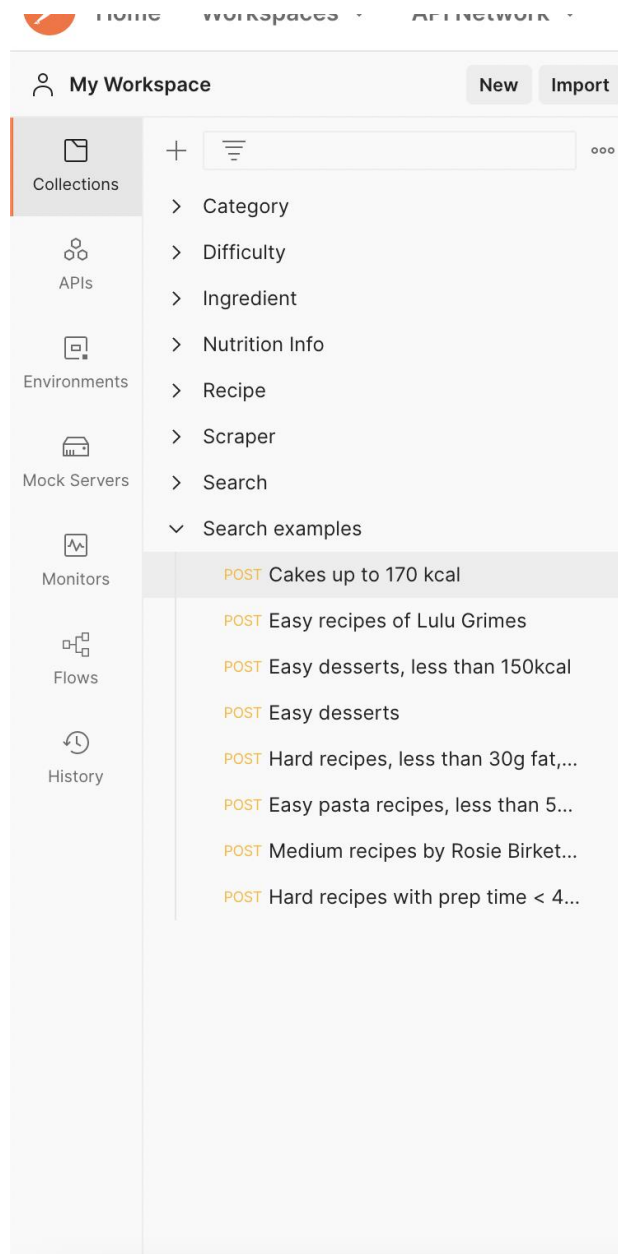


Εικόνα 56: Κλικ στο “recipes” και “search recipes”



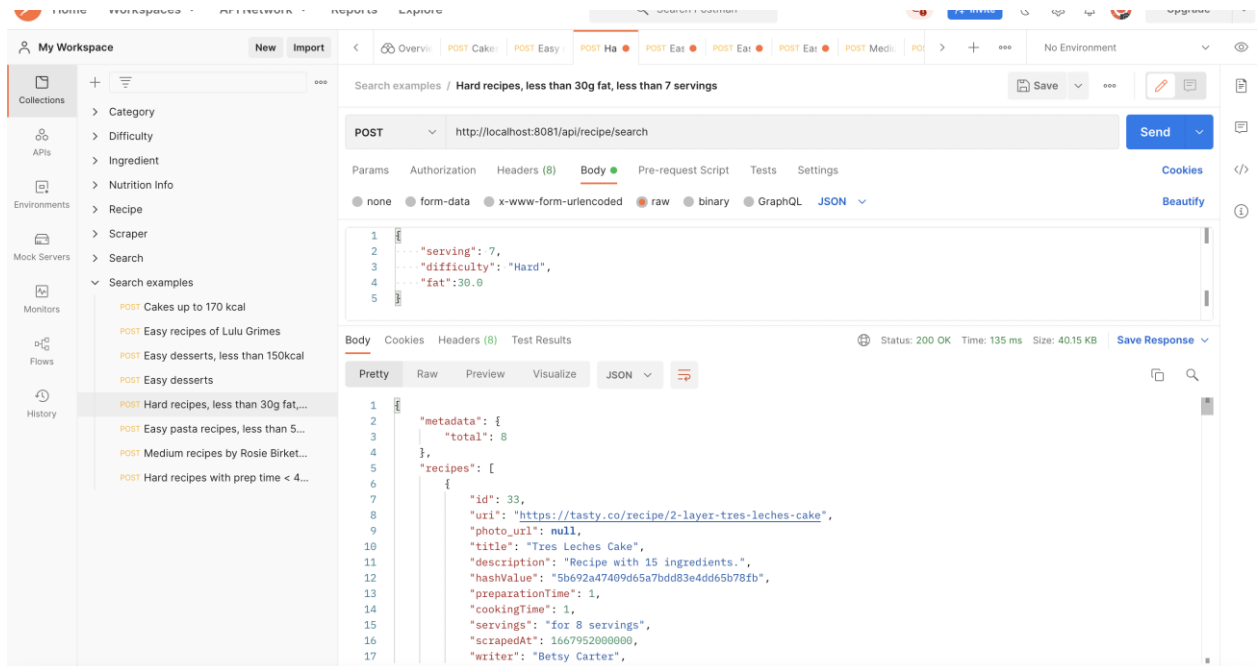
Εικόνα 57: Τα πεδία των φίλτρων

Τα αποτελέσματα μας έχουν τεστταριστεί και στο postman. Εκεί θα βλέπαμε τα αποτελέσματα άλλωστε αν δεν υπήρχε το user interface. Μερικά παραδείγματα αναζητήσεων στο postman είναι τα παρακάτω:



Εικόνα 58: Search examples στο postman

Βλέπουμε εδώ ας πούμε τα αποτελέσματα της αναζήτησης για συνταγές δυσκολίας “Hard”, που έχουν λιγότερα από 30 γραμμάρια λίπους και βγάζουν λιγότερες από 7 μερίδες.



Εικόνα 59: Παράδειγμα αναζήτησης με περιορισμούς στο postman

Όσον αφορά τον κώδικα:

Αρχικά, έχουμε τους 5 crawlers υλοποιημένους σε python με χρήση του scrapy. Τα αρχεία των crawlers είναι τα παρακάτω:

- scrape_allrecipes.sh
- scrape_bbcgoodfood.sh
- scrape_cookingclassy.sh
- scrape_simplyrecipes.sh
- scrape_tasty.sh

Εικόνα 60: Τα αρχεία των crawlers

Τα αρχεία ακολουθούν όλα την παρακάτω μορφή με κάποιες διαφοροποιήσεις/ προσαρμογές ανάλογα το site, ώστε να καταγράφονται μόνο οι σύνδεσμοι που μας ενδιαφέρουν.

Για το AllRecipesCrawler.py:

```
from scrapy import Item, Field
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor

class AllRecipesCrawler(CrawlSpider):
    name = 'allrecipes'
    allowed_domains = ['www.allrecipes.com']
    start_urls = ['https://www.allrecipes.com']
    rules = (Rule(LinkExtractor(), callback='parse_item', follow=True),)
    filename = 'allrecipes.txt'
    accept_url = "https://www.allrecipes.com/recipe/"
    reject_urls = ["printview", "page"]

    def parse_item(self, response):
        if response.status == 200:
            url = response.url

            item = {}
            item['url'] = response.url
            item['referer'] = response.request.headers.get('Referer')
            item['status'] = response.status

            if url.startswith(self.accept_url):
                matches = True
                for prefix in self.reject_urls:
                    if prefix in url:
                        matches = False

            if matches:
                with open(self.filename, 'at') as f:
                    f.write(url + "\n")

        return item
```

Εικόνα 61: Το αρχείο του crawler για το allrecipes.com

Για τον CookingClassyCrawler.py:

```

from scrapy import Item, Field
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor

class CookingClassyCrawler(CrawlSpider):
    name = 'cookingclassy'
    allowed_domains = ['www.cookingclassy.com']
    start_urls = ['https://www.cookingclassy.com/recipes']
    rules = (Rule(LinkExtractor(), callback='parse_item', follow=True),)
    filename = 'cookingclassy.txt'
    accept_url = "https://www.cookingclassy.com/"
    reject_urls = ["https://www.cookingclassy.com/tag/", "https://www.cookingclassy.com/"]

    def parse_item(self, response):
        if response.status == 200:
            url = response.url

            item = { }
            item['url'] = response.url
            item['referer'] = response.request.headers.get('Referer')
            item['status'] = response.status

            if url.startswith(self.accept_url) and not "comment" in url:
                matches = True
                for prefix in self.reject_urls:
                    if url.startswith(prefix):
                        matches = False

                if matches:
                    with open(self.filename, 'at') as f:
                        f.write(url + "\n")

            return item

```

Εικόνα 62: Το αρχείο του crawler για το cookingclassy.com

Η main των scrapers είναι όπως φαίνεται παρακάτω:

```

1  import crawler.Importer;
2  import database.Database;
3  import model.Recipe;
4  import repository.RecipeRepository;
5  import scraper.*;
6
7  import java.util.List;
8  import java.util.Map;
9  import java.util.TreeMap;
10
11 public class Main {
12     @
13     private static boolean validate(Recipe r) {
14         if (r.nutritionInfo.kcal <= 0 || r.nutritionInfo.sugars<=0 || r.nutritionInfo.saturated_fat<=0 || r.nutritionInfo.protein<=0 || r.nutritionInfo.carbohydrates<
15             return false;
16         }
17         if (r.preparation_time <= 0 || r.cooking_time<=0) {
18             return false;
19         }
20         if (r.number_of_ingredients<=0) {
21             return false;
22         }
23     }
24     return true;
25 }
26
27 public static void main(String [] args) {
28     ScraperInterface allRecipesScraper = new AllRecipesScraper();
29     ScraperInterface bbcGoodFoodScraper = new BbcGoodFoodScraper();
30     ScraperInterface cookingClassyScraper = new CookingClassyScraper();
31     ScraperInterface simplyRecipesScraper = new SimplyRecipesScraper();
32     ScraperInterface tastyScraper = new TastyScraper();
33
34

```

Εικόνα 63: Η main των scrapers

```

35     Importer importer = new Importer();
36     Database db = new Database( autocommit: false, verbose: false);
37     RecipeRepository repository = new RecipeRepository(db);
38
39     TreeMap<String, ScraperInterface> jobs = new TreeMap<>();
40
41     try {
42         List<String> allRecipeUrls = importer.loadFromResource("allrecipes.txt");
43         List<String> bbcgoodfoodUrls = importer.loadFromResource("bbcgoodfood.txt");
44         List<String> cookingClassyUrls = importer.loadFromResource("cookingclassy.txt");
45         List<String> simplyRecipesUrls = importer.loadFromResource("simplyrecipes.txt");
46         List<String> tastyUrls = importer.loadFromResource("tasty.txt");
47
48         for (String url : allRecipeUrls) {
49             jobs.put(url, allRecipesScraper);
50         }
51         for (String url : bbcgoodfoodUrls) {
52             jobs.put(url, bbcGoodFoodScraper);
53         }
54         for (String url : cookingClassyUrls) {
55             jobs.put(url, cookingClassyScraper);
56         }
57         for (String url : tastyUrls) {
58             jobs.put(url, tastyScraper);
59         }
60         for (String url : simplyRecipesUrls) {
61             jobs.put(url, simplyRecipesScraper);
62         }
63
64         int counter = 0;
65         int failed = 0;
66
67         for (Map.Entry<String, ScraperInterface> job : jobs.entrySet()) {
68             String url = job.getKey();

```

Εικόνα 64: Συνέχεια της main

```

68     String url = job.getKey();
69     ScraperInterface scraper = job.getValue();
70
71     List<Recipe> results = scraper.scrape(url);
72
73     for (Recipe r : results) {
74         System.out.println("=====");
75
76         r.print();
77
78         if (validate(r) == true) {
79             db.newTransaction();
80
81             try {
82                 repository.insert(r);
83
84                 db.commit();
85                 counter++;
86             } catch (Exception ex) {
87                 db.rollback();
88                 System.out.println("[ *** Warning *** ] Recipe could not be inserted: " + ex.getMessage());
89                 failed++;
90             }
91         } else {
92             System.out.println("[ *** Warning *** ] Recipe: " + r.uri + " was skipped. Validation failed.");
93         }
94     }
95
96     System.out.println("Imported: " + counter);
97     System.out.println("Failed : " + failed);
98 } catch (Exception ex) {
99     ex.printStackTrace();
100 }
101 }

```

Εικόνα 65: Συνέχεια της main

Ο κώδικας του κάθε scraper είναι διαφορετικός.
 Η “main” του backend είναι η “RecipesApplication”, που φαίνεται παρακάτω:

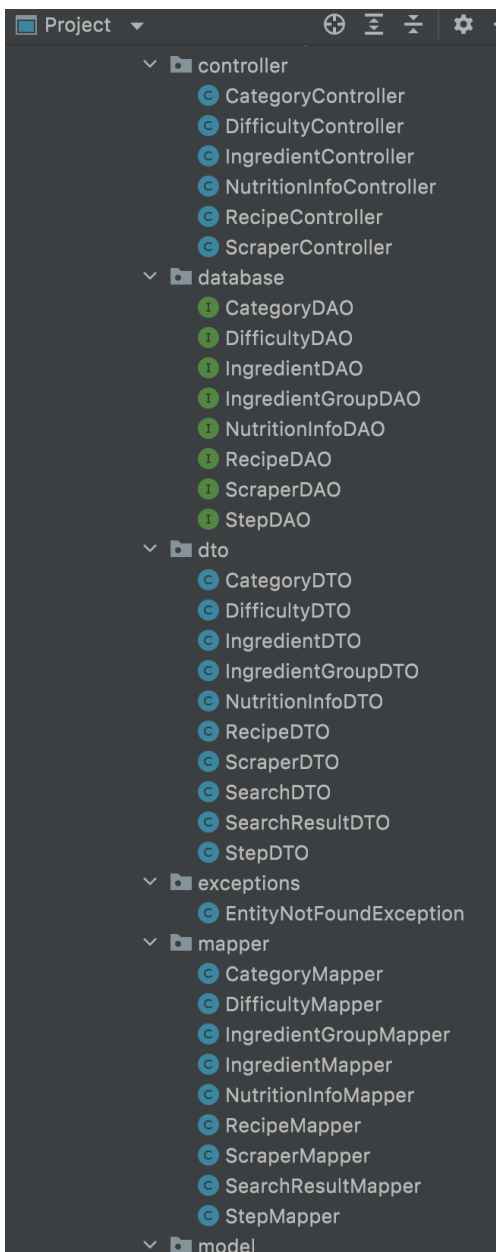
```

RecipesApplication
RecipesApplication.java
1 package com.anna.recipes;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5 import org.springframework.context.annotation.ComponentScan;
6 import org.springframework.web.servlet.config.annotation.CorsRegistry;
7 import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
8
9 @SpringBootApplication
10 public class RecipesApplication {
11
12     public static void main(String[] args) {
13         SpringApplication.run(RecipesApplication.class, args);
14     }
15 }
16

```

Εικόνα 66: Backend - Main

Στο πλαίσιο του backend έχουμε χρησιμοποιήσει DAOs, DTOs, Mappers, Controllers, για κάθε οντότητα της βάσης όπως φαίνεται παρακάτω:



Εικόνα 67: Αρχεία του backend

Ενδεικτικά ακολουθεί ο κώδικας καθενός αρχείου για την οντότητα “category” :

Controller:

```
CategoryController.java
1 package com.anna.recipes.controller;
2
3 import com.anna.recipes.dto.CategoryDTO;
4 import com.anna.recipes.exceptions.EntityNotFoundException;
5 import com.anna.recipes.mapper.CategoryMapper;
6 import com.anna.recipes.service.CategoryService;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.web.bind.annotation.GetMapping;
9 import org.springframework.web.bind.annotation.PathVariable;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.RestController;
12
13 import java.util.List;
14
15 @RestController
16 @RequestMapping("/api")
17 public class CategoryController {
18     @Autowired
19     CategoryService service;
20
21     @Autowired
22     CategoryMapper mapper;
23
24     @GetMapping(value="/category/{id}")
25     public CategoryDTO getById(@PathVariable("id") int id){
26         return mapper.toDTO(service.findById(id).orElseThrow(()->new EntityNotFoundException("Category")));
27     }
28
29     @GetMapping(value="/category/all")
30     public List<CategoryDTO> getAll(){
31         return mapper.toDTO(service.list());
32     }
33
34 }
35
```

Εικόνα 68: Κώδικας controller για category

DAO(Data Access Object):

```
CategoryDAO.java x
1 package com.anna.recipes.database;
2
3 import com.anna.recipes.model.Category;
4 import org.springframework.data.jpa.repository.JpaRepository;
5 import org.springframework.stereotype.Repository;
6
7 import java.util.Optional;
8
9 @Repository
10 public interface CategoryDAO extends JpaRepository<Category, Long> {
11     Optional<Category> findById(int id);
12
13 }
```

Εικόνα 69: Κώδικας DAO για category

Ακολουθεί επίσης το DAO του “Recipe” το οποίο κάνει παραπάνω queries(δεν κάνει απλά εύρεση με βάση Id όπως τα υπόλοιπα). Τα φέρνει σε σειρά με βάση τη χρονική στιγμή που έγιναν scraped και κάνει και τα απαραίτητα «ταιριάσματα» στον τίτλο και την περιγραφή.

```
RecipeDAO.java
1 package com.anna.recipes.database;
2
3 import com.anna.recipes.model.Recipe;
4 import org.springframework.data.jpa.repository.JpaRepository;
5 import org.springframework.data.jpa.repository.Query;
6 import org.springframework.stereotype.Repository;
7 import java.util.List;
8 import java.util.Optional;
9
10 @Repository
11 public interface RecipeDAO extends JpaRepository<Recipe, Long> {
12     Optional<Recipe> findById(int id);
13
14     @Query(nativeQuery = true, value = "SELECT * FROM recipe order by scraped_at desc limit 10")
15     List<Recipe> latest();
16
17
18     @Query(nativeQuery = true, value = "select * from recipe where MATCH(title,description) against((select title
19     List<Recipe> findSimilar(int id);
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2
```



```

1 package com.anna.recipes.mapper;
2
3 import com.anna.recipes.dto.CategoryDTO;
4 import com.anna.recipes.model.Category;
5 import org.springframework.stereotype.Service;
6
7 import java.util.ArrayList;
8 import java.util.List;
9
10 @Service
11 public class CategoryMapper {
12     @Transactional
13     public CategoryDTO toDTO(Category entity) {
14         CategoryDTO dto = new CategoryDTO();
15         dto.setId(entity.getId());
16         dto.setDescription(entity.getDescription());
17         return dto;
18     }
19
20     @Transactional
21     public List<CategoryDTO> toDTO(List<Category> entities) {
22         List<CategoryDTO> dtos = new ArrayList<>();
23
24         for (Category c : entities) {
25             dtos.add(toDTO(c));
26         }
27         return dtos;
28     }
29 }

```

Εικόνα 71: Κώδικας mapper για category

4.2 Πίνακες με χρήσιμα στατιστικά και πληροφορίες

Πίνακας 1: Links που εισήχθησαν στη βάση δεδομένων

Allrecipes.com	Bbcgoodfood.com	Tasty.co	Cookingclassy.com	Simplyrecipes.com
9750	8307	286	1064	1439

Πίνακας 2: Μνήμη που απαιτήθηκε για κάθε αρχείο txt που περιλαμβάνει τα links με τις συνταγές του κάθε σάιτ.

Simplyrecipes.txt	Tasty.txt	Allrecipes.txt	Bbcgoodfood.txt	Cookingclassy.txt
177.012 bytes	154.282 bytes	925.037 bytes	947.762 bytes	106.697 bytes

Πίνακας 3: Μνήμη που απαιτήθηκε για τη βάση δεδομένων

Recipe_db	MySql
77.3437500 MB	2.5937500 MB

Πίνακας 4: Μέσος χρόνος scraping και crawling

Scraping	Crawling(fetching links of each site)
> 2 recipes / 1 sec	>2500 /1 hour

5. ΣΥΜΠΕΡΑΣΜΑΤΑ

5.1 Γενικά Συμπεράσματα

Οι τεχνολογίες έχουν εξελιχθεί πολύ. Ένα απλό site πια δεν είναι απλά μια html σελίδα, αλλά είναι πολλές τεχνολογίες που αλληλοεπιδρούν μεταξύ τους με τους τρόπους που είδαμε.

Ουσιαστικά, σκοπός του project είναι να διευκολύνει το χρήστη προσφέροντάς του πρόσβαση σε συνταγές από πολλές διαφορετικές πηγές, καθώς και ευέλικτους και βολικούς τρόπους αναζήτησης που θα τον διευκολύνουν πολύ καλύτερα από το να έψαχνε σε κάθε site ξεχωριστά. Αντί να αναζητά δηλαδή σε κάθε μηχανή αναζήτησης και έπειτα σε κάθε διαφορετικό ιστότοπο, έχουμε συγκεντρώσει την πληροφορία από πολλές διαφορετικές πηγές, ώστε να επιταχύνουμε και να διευκολύνουμε τον χρήστη.

5.2 Δυνατότητες Επέκτασης

Η εφαρμογή αυτή είναι δυνατό να επεκταθεί στο μέλλον και να προστεθούν κι άλλα features. Για παράδειγμα, οι scrapers μπορούν να σαρώνουν τις εικόνες των συνταγών, να τις περνάμε με links στη ΒΔ και στη συνέχεια να τις στέλνουμε κι αυτές στο frontend ώστε να παρουσιάζονται πιο όμορφα τα αποτελέσματα στο χρήστη. Επιπλέον, θα μπορούσαν να προστεθούν και άλλες περιπτώσεις αναζητήσεων, να επεκταθεί ο σχεδιασμός της διεπαφής ώστε να παρέχει περισσότερες δυνατότητες στο χρήστη ή και να φτιαχτούν και άλλοι scrapers για άλλα sites, έτσι ώστε να υπάρχει ακόμη μεγαλύτερος όγκος συνταγών στη βάση μας.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	Ελληνικός Όρος
Web Crawling	Ανίχνευση Ιστού
Web Scraping	Σάρωση Ιστού
Database	Βάση Δεδομένων
Site	Ιστότοπος
User Interface	Διεπαφή Χρήστη
Link	Σύνδεσμος
Feature	Δυνατότητα/ Χαρακτηριστικό
Input	Εισαγόμενα δεδομένα
Crawler	Ανιχνευτής
Development	Ανάπτυξη
Scraper	Σαρωτής
Project	Έργο
Design pattern	Μοτίβο Σχεδίασης
Deployment	Ανάπτυξη
Components	Συστατικά
Framework	Δομή
Template	Πρότυπο
Data	Δεδομένα
Extract data	Απόσπαση δεδομένων
End-to-end	Από άκρη σε άκρη
Servers	Διακομιστές
Import	Εισαγωγές
Portable	Φορητή
Configurations	Διαμορφώσεις
Settings	Ρυθμίσεις
Documentation	Τεκμηρίωση
Validation	Επικύρωση
Run	Εκτέλεση
Entity	Οντότητα
Request	Αίτημα
Browser	Προγράμματα περιήγησης
Query	Ερώτημα
Continue	Συνέχεια
Search	Αναζήτηση
Close	Κλείσιμο
Terminal	Τερματικό

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

UI	User Interface
Txt	Text file
DAO	Data Access Object
DTO	Data Transaction Object
URL	Uniform Resource Locator
URI	Uniform Resource Identifier
Json	JavaScript Object Notation
PC	Personal computer
BCNF	Boyce–Codd Normal Form
MVC	Model view controller
MTV	Model template views
API	Application programming interface
Rest	Representational state transfer
ΕΚΠΑ	Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

ΑΝΑΦΟΡΕΣ

- [1] *Web scraping* (2023) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Web_scraping (Προσπελάστηκε: 1/11/2021).
- [2] *Web Crawler* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Web_crawler (Προσπελάστηκε: 1/11/2021).
- [3] *Representational state transfer* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Representational_state_transfer (Προσπελάστηκε: 12/2/2022).
- [4] *Frontend and backend* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Frontend_and_backend (Προσπελάστηκε: 5/3/2022).
- [5] *Relational database* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Relational_database (Προσπελάστηκε: 19/1/2022).
- [6] *NoSQL* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: <https://en.wikipedia.org/wiki/NoSQL> (Προσπελάστηκε: 19/1/2022).
- [7] *A fast and powerful scraping and web crawling framework* (no date) *Scrapy*. Διαθέσιμο στο: <https://scrapy.org/> (Προσπελάστηκε: 1/3/2022).
- [8] Committee, A.N.P.M. (no date) *Apache nutch™, Apache Nutch™*. Διαθέσιμο στο: <https://nutch.apache.org/> (Προσπελάστηκε: 21/11/2021).
- [9] *Heritrix 3 documentation* (no date) *Heritrix 3 Documentation - Heritrix 3 documentation*. Διαθέσιμο στο: <https://heritrix.readthedocs.io/en/latest/> (Προσπελάστηκε: 10/1/2022).
- [10] *Spring Boot* (no date) *Spring*. Διαθέσιμο στο: <https://spring.io/projects/spring-boot> (Προσπελάστηκε: 26/3/2022).
- [11] *The PHP framework for web artisans* (no date) *Laravel*. Διαθέσιμο στο: <https://laravel.com/> (Προσπελάστηκε: 30/3/2022).
- [12] *Django* (no date) *Django Project*. Διαθέσιμο στο: <https://www.djangoproject.com/>. (Προσπελάστηκε: 19/1/2022).
- [13] *React – a JavaScript library for building user interfaces* (no date) – *A JavaScript library for building user interfaces*. Διαθέσιμο στο: <https://reactjs.org/> (Προσπελάστηκε: 1/5/2022).
- [14] *The progressivejavascript framework* (no date) *Vue.js - The Progressive JavaScript Framework | Vue.js*. Διαθέσιμο στο: <https://vuejs.org/> (Προσπελάστηκε: 1/6/2022).
- [15] (no date) *Angular*. Διαθέσιμο στο: <https://angular.io/> (Προσπελάστηκε: 18/5/2022).
- [16] (no date) *MySQL*. Διαθέσιμο στο: <https://www.mysql.com/> (Προσπελάστηκε: 15/1/2022).
- [17] Group, P.S.Q.L.G.D. (2023) *PostgreSQL*. Διαθέσιμο στο: <https://www.postgresql.org/> (Προσπελάστηκε: 15/1/2022).
- [18] *MySQL Workbench download now "* (no date) *MySQL*. Διαθέσιμο στο: <https://www.mysql.com/products/workbench/> (Προσπελάστηκε: 17/1/2022).
- [19] *MySQL after insert trigger by practical examples* (2020) *MySQL Tutorial*. Διαθέσιμο στο: <https://www.mysqltutorial.org/mysql-triggers/mysql-after-insert-trigger/>. (Προσπελάστηκε: 26/1/2022).
- [20] *MD5* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: <https://en.wikipedia.org/wiki/MD5> (Προσπελάστηκε: 9/2/2022).
- [21] *Java HTML parser, built for html editing, cleaning, scraping, and XSS Safety* (no date) *jsoup*. Διαθέσιμο στο: <https://jsoup.org/> (Προσπελάστηκε: 17/2/2022).
- [22] *IntelliJ IDEA – the leading Java and Kotlin Ide* (no date) *JetBrains*. Διαθέσιμο στο: <https://www.jetbrains.com/idea/> (Προσπελάστηκε: 1/9/2021).
- [23] Porter, B., Zyl, J.van and Lamy, O. (no date) *Welcome to Apache Maven, Maven*. Διαθέσιμο στο: <https://maven.apache.org/> (Προσπελάστηκε: 1/9/2021).
- [24] *Data Access Object* (no date) *Design Patterns: Data Access Object*. Διαθέσιμο στο: <https://www.oracle.com/java/technologies/data-access-object.html> (Προσπελάστηκε: 16/4/2022).
- [25] *Spring - Rest Controller* (2021) *GeeksforGeeks*. Διαθέσιμο στο: <https://www.geeksforgeeks.org/spring-rest-controller/> (Προσπελάστηκε: 20/3/2022).
- [26] Seniuk, V. (2020) *Data Transfer Object Pattern in java - implementation and mapping, Stack Abuse*. Stack Abuse. Διαθέσιμο στο: <https://stackabuse.com/data-transfer-object-pattern-in-java-implementation-and-mapping/> (Προσπελάστηκε: 22/3/2022).
- [27] Lubowa, E. (2022) *Why you need to use DTO's in your rest api*, *Medium*. Medium. Διαθέσιμο στο: <https://medium.com/@enocklubowa/why-you-need-to-use-dtos-in-your-rest-api-d9d6d7be5450> (Προσπελάστηκε: 12/4/2022).
- [28] *Node.js* (no date) *Node.js*. Διαθέσιμο στο: <https://nodejs.org/en/> (Προσπελάστηκε: 7/6/2022).
- [29] *Vue Cli* (no date) *Vue CLI*. Διαθέσιμο στο: <https://cli.vuejs.org/> (Προσπελάστηκε: 8/6/2022).
- [30] Microsoft (2021) *Visual studio code - code editing, redefined, RSS*. Microsoft. Διαθέσιμο στο: <https://code.visualstudio.com/> (Προσπελάστηκε: 14/4/2022).
- [31] *The Intuitive Vue Framework* (no date) *Nuxt*. Διαθέσιμο στο: <https://nuxtjs.org/> (Προσπελάστηκε: 28/4/2022).

- [32] *FoodHub - Food Delivery Template* (no date) *Vuetify Store*. Διαθέσιμο στο: https://store.vuetifyjs.com/products/foodhub-food-delivery-template?utm_source=vuetifyjs.com&utm_medium=themes (Προσπελάστηκε: 3/5/2022).
- [33] *A material design framework for vue.js* (no date) *Vuetify*. Διαθέσιμο στο: <https://vuetifyjs.com/en/> (Προσπελάστηκε: 4/5/2022).
- [34] *Babel · The compiler for next generation javascript* (no date) *Babel*. Διαθέσιμο στο: <https://babeljs.io/> (Προσπελάστηκε: 7/5/2022).
- [35] *Boyce–Codd Normal Form* (2022) *Wikipedia*. Wikimedia Foundation. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Boyce%E2%80%93Codd_normal_form (Προσπελάστηκε: 6/1/2022).
- [36] (no date) *Postman*. Διαθέσιμο στο: <https://www.postman.com/> (Προσπελάστηκε: 1/3/2022).
- [37] *API testing client that flows with you* (no date) *HTTPie*. Διαθέσιμο στο: <https://httpie.io/> (Προσπελάστηκε: 12/5/2022).
- [38] *PyCharm: The python IDE for professional developers by jetbrains* (no date) *JetBrains*. Διαθέσιμο στο: <https://www.jetbrains.com/pycharm/> (Προσπελάστηκε: 18/12/2021).