Τεχνική Αναφορά: Ανάπτυξη Διαδικτυακής Υπηρεσίας "StreetFoodGo"

Ομάδα Ανάπτυξης:

- Αναστασία Κανιστρά(2023019)
- Μάριος Ταφάνι(2023146)
- Αμβράζη Αλεξάνδρα Άννα(2023002)

Ημερομηνία: Ιανουάριος 2026

---

1. Περιγραφή Προβλήματος και Ρόλοι Χρηστών

1.1 Σκοπός και Περιγραφή Προβλήματος

Η πλατφόρμα **StreetFoodGo** είναι μια διαδικτυακή υπηρεσία που στοχεύει στην ψηφιοποίηση της εμπειρίας παραγγελίας από μικρά καταστήματα εστίασης και food trucks. Το κύριο πρόβλημα που επιλύει είναι η αδυναμία των πελατών να εντοπίσουν εύκολα ποια μικρά καταστήματα είναι ανοιχτά στην περιοχή τους και να πραγματοποιήσουν παραγγελίες απομακρυσμένα, αποφεύγοντας την αναμονή σε ουρές.

Το σύστημα λειτουργεί ως ένας κεντρικός κόμβος (hub) που συνδέει τους πεινασμένους πελάτες με τους ιδιοκτήτες καταστημάτων, παρέχοντας δυνατότητες αναζήτησης μενού, διαχείρισης καλαθιού και ολοκλήρωσης παραγγελιών σε πραγματικό χρόνο.

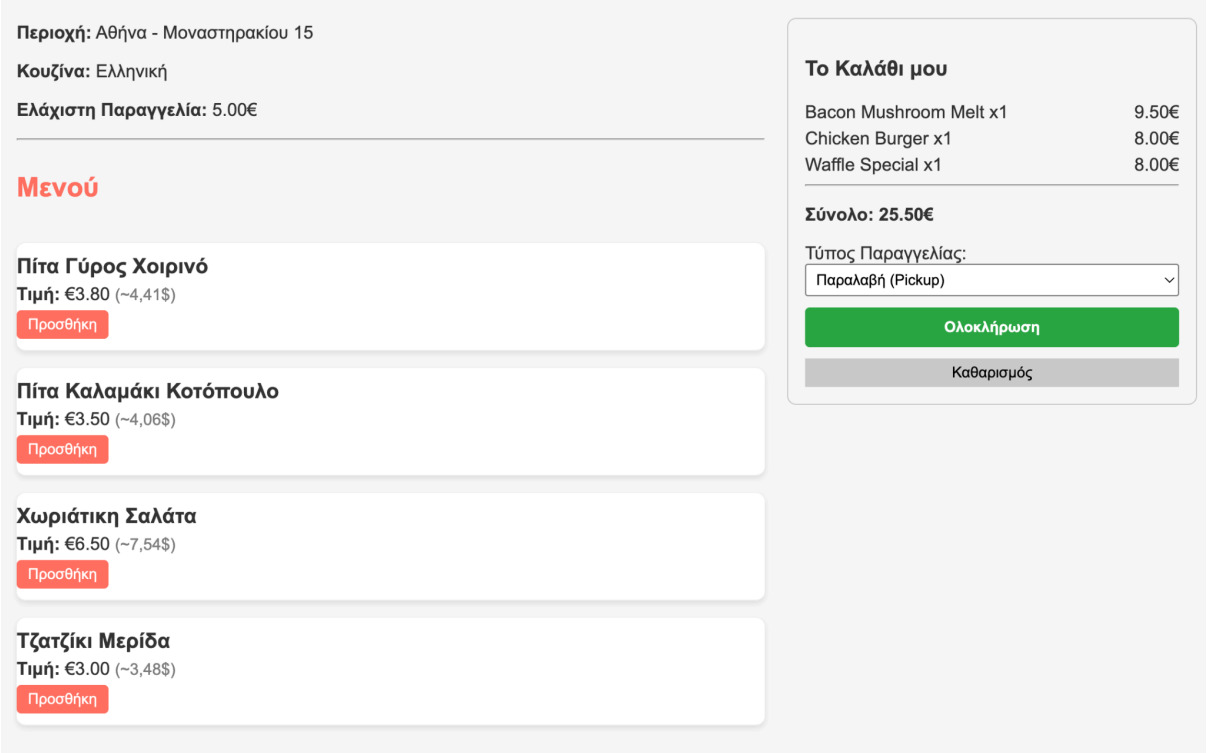
1.2 Υποστηριζόμενοι Ρόλοι Χρηστών

Το σύστημα έχει σχεδιαστεί για να υποστηρίζει τρεις διακριτούς ρόλους χρηστών, με διαφορετικά επίπεδα πρόσβασης και λειτουργικότητας:

1. **Επισκέπτης (Guest):**
    - Μπορεί να πλοηγηθεί στην αρχική σελίδα και να δει τη λίστα με τα διαθέσιμα καταστήματα.
    - Έχει πρόσβαση στα μενού των καταστημάτων για να δει προϊόντα και τιμές.
    - Δεν μπορεί να πραγματοποιήσει παραγγελία αν δεν εγγραφεί/συνδεθεί.
    - Βλέπει τις τρέχουσες συναλλαγματικές ισοτιμίες (USD/EUR) στη σελίδα του καταστήματος.
2. **Πελάτης (Customer):**
    - Διαθέτει όλες τις δυνατότητες του επισκέπτη.
    - Μπορεί να πραγματοποιήσει είσοδο (Login) στο σύστημα.

- Διαχειρίζεται το προφίλ του και τις αποθηκευμένες διευθύνσεις παράδοσης.
- Μπορεί να προσθέσει προϊόντα στο καλάθι και να ολοκληρώσει παραγγελίες (Delivery ή Take Away).
- Έχει πρόσβαση στο ιστορικό των παραγγελιών του (`My Orders`).

3. **Ιδιοκτήτης Καταστήματος (Store Owner):**
   - Έχει πρόσβαση σε διαχειριστικό περιβάλλον (Dashboard).
   - Μπορεί να αλλάξει την κατάσταση του καταστήματος (Ανοιχτό/Κλειστό) σε πραγματικό χρόνο.
   - Επιβλέπει τις εισερχόμενες παραγγελίες.
   - *Σημείωση:* Το σύστημα διασφαλίζει ότι μόνο ο ιδιοκτήτης του συγκεκριμένου καταστήματος μπορεί να αλλάξει τις ρυθμίσεις του.



Εικόνα 1: Η σελίδα μενού του καταστήματος με εμφάνιση της τρέχουσας ισοτιμίας συναλλάγματος.

---

2. Αρχιτεκτονική Συστήματος

2.1 Γενική Επισκόπηση

Η εφαρμογή αναπτύχθηκε χρησιμοποιώντας το πλαίσιο **Spring Boot** και ακολουθεί μια πολυεπίπεδη αρχιτεκτονική (Layered Architecture). Επιπλέον, έχουν ενσωματωθεί στοιχεία **Hexagonal Architecture (Ports & Adapters)** για την επικοινωνία με εξωτερικές υπηρεσίες, διασφαλίζοντας την αποσύζευξη του πυρήνα της εφαρμογής από εξωτερικές εξαρτήσεις.

2.2 Layers (Επίπεδα) Εφαρμογής

Η δομή του κώδικα χωρίζεται στα εξής επίπεδα:

- **Presentation Layer (Controllers):**
  - `Web Controllers` (π.χ. `StoreController`, `OrderController`): Διαχειρίζονται τα αιτήματα HTTP και επιστρέφουν HTML σελίδες μέσω της μηχανής προτύπων **Thymeleaf**.
  - `REST Controllers` (π.χ. `OrderRestController`): Εκθέτουν endpoints που επιστρέφουν δεδομένα σε μορφή **JSON**, επιτρέποντας την κατανάλωση της υπηρεσίας από τρίτα συστήματα (Mobile Apps, SPAs).
- **Service Layer (Business Logic):**
  - Περιέχει την επιχειρησιακή λογική (π.χ. `OrderService`, `StoreService`, `UserService`).
  - Εδώ γίνονται οι έλεγχοι εγκυρότητας (Validation), όπως ο έλεγχος αν το κατάστημα είναι ανοιχτό ή αν καλύπτεται το ελάχιστο ποσό παραγγελίας.
  - Χρήση Interfaces για την επίτευξη χαλαρής σύζευξης (Loose Coupling).
- **Data Access Layer (Repositories):**
  - Υλοποιείται μέσω **Spring Data JPA**.
  - Τα Repositories (π.χ. `UserRepository`, `OrderRepository`) παρέχουν μεθόδους για την επικοινωνία με τη βάση δεδομένων χωρίς την ανάγκη συγγραφής SQL queries.
- **Domain Model (Entities):**
  - Οι κλάσεις που αντιπροσωπεύουν τον πίνακα της βάσης δεδομένων (π.χ. `User`, `Order`, `Product`, `Store`) με τις μεταξύ τους συσχετίσεις (OneToMany, ManyToOne).

2.3 Τεχνολογίες και Components

- **Backend:** Java, Spring Boot 3.
- **Database:** H2 Database (In-memory/File-based) για την αποθήκευση δεδομένων κατά την ανάπτυξη.
- **Security:** Spring Security με υποστήριξη JWT και Form Login.
- **Build Tool:** Maven.
- **Utilities:** Lombok (για μείωση boilerplate κώδικα).

---

3. Περιγραφή Εξωτερικών Υπηρεσιών

Στο πλαίσιο των απαιτήσεων για κατανεμημένα συστήματα, η εφαρμογή StreetFoodGo καταναλώνει μια εξωτερική υπηρεσία συναλλάγματος.

3.1 Υπηρεσία Συναλλάγματος (Exchange Rate API)

Η εφαρμογή παρέχει στους χρήστες την πληροφορία της τρέχουσας ισοτιμίας Ευρώ προς Δολαρίου, ώστε να γνωρίζουν την αξία των προϊόντων σε διεθνές νόμισμα.

- **Πάροχος:** `exchangerate-api.com`
- **Συμβόλαιο (Contract):** Η υπηρεσία καλείται μέσω HTTP GET αιτήματος στο endpoint `https://api.exchangerate-api.com/v4/latest/EUR`. Επιστρέφει ένα JSON αντικείμενο που περιέχει πεδία όπως `base` (EUR), `date`, και ένα map `rates` με τις ισοτιμίες.
- **Τρόπος Ενσωμάτωσης (Ports & Adapters):**
  - Ορίστηκε ένα Interface `CurrencyService` (Port) που περιέχει τη μέθοδο `getDollarRate()`.
  - Υλοποιήθηκε η κλάση `ExternalCurrencyService` (Adapter) η οποία χρησιμοποιεί το `RestTemplate` του Spring για να κάνει το HTTP call.
  - Αυτό επιτρέπει την εύκολη αντικατάσταση της εξωτερικής υπηρεσίας στο μέλλον χωρίς να αλλάξει ο κώδικας των Controllers.
- **Χειρισμός Σφαλμάτων:** Η κλήση προς το API βρίσκεται εντός `try-catch` block. Σε περίπτωση αποτυχίας (π.χ. timeout, network error), η υπηρεσία δεν "ρίχνει" την εφαρμογή, αλλά καταγράφει το σφάλμα και επιστρέφει `null` ή μια default τιμή, διασφαλίζοντας την αδιάλειπτη λειτουργία του UI.

---

4. Μηχανισμοί Ασφάλειας και Τεκμηρίωση API

4.1 Authentication & Authorization

Η ασφάλεια υλοποιείται μέσω του **Spring Security** και ακολουθεί μια υβριδική προσέγγιση για να εξυπηρετήσει τόσο τους χρήστες του Browser όσο και τους REST Clients.

- **Authentication (Ταυτοποίηση):**
  - **Web UI:** Χρησιμοποιείται κλασικό Session-based authentication (Cookies) με Form Login. Οι χρήστες εγγράφονται και τα συνθηματικά τους αποθηκεύονται κρυπτογραφημένα με **BCrypt**.
  - **REST API:** Για τα endpoints κάτω από το `/api/**`, η εφαρμογή υποστηρίζει **Stateless Authentication** με χρήση **JWT (JSON Web Tokens)**. Έχει υλοποιηθεί το `JwtAuthenticationFilter` που ελέγχει την επικεφαλίδα `Authorization` σε κάθε αίτημα.
- **Authorization (Εξουσιοδότηση):**
  - Χρήση Role-Based Access Control (RBAC).
  - Ο ρόλος `ROLE_OWNER` απαιτείται για πρόσβαση στο `/owner/**` και για αλλαγή κατάστασης καταστήματος.
  - Ο ρόλος `ROLE_CUSTOMER` απαιτείται για την υποβολή παραγγελίας.
  - Οι επισκέπτες έχουν πρόσβαση μόνο σε δημόσια endpoints (`/login`, `/register`, `/stores`).

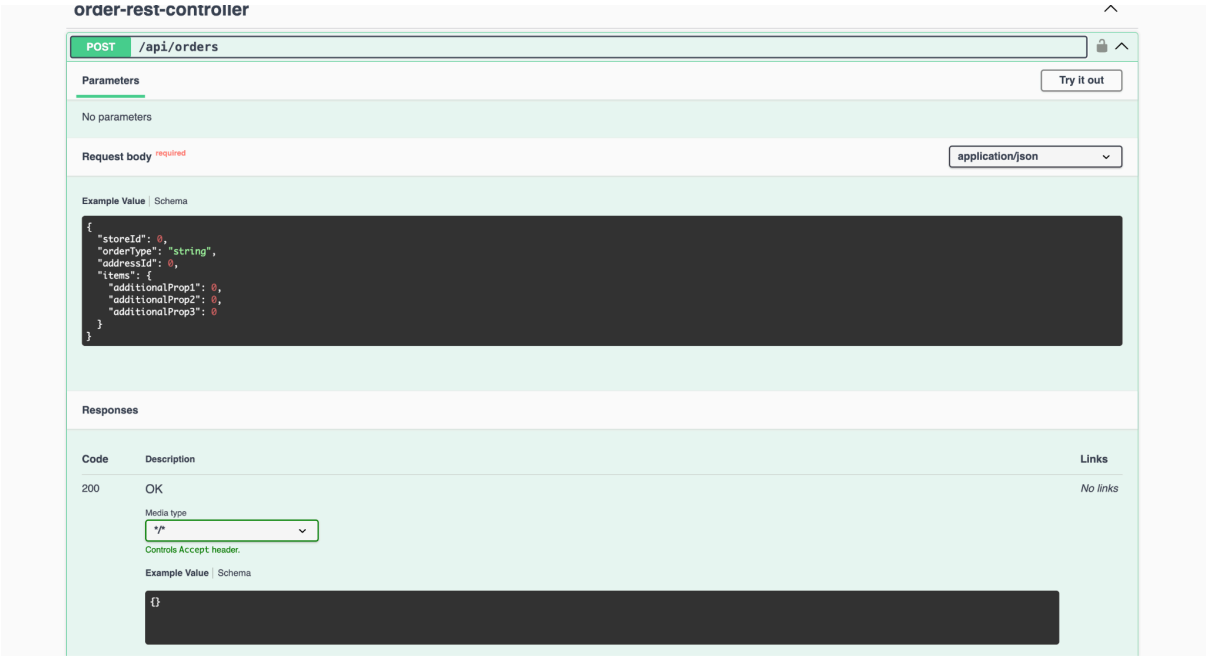
4.2 Τεκμηρίωση REST API (OpenAPI/Swagger)

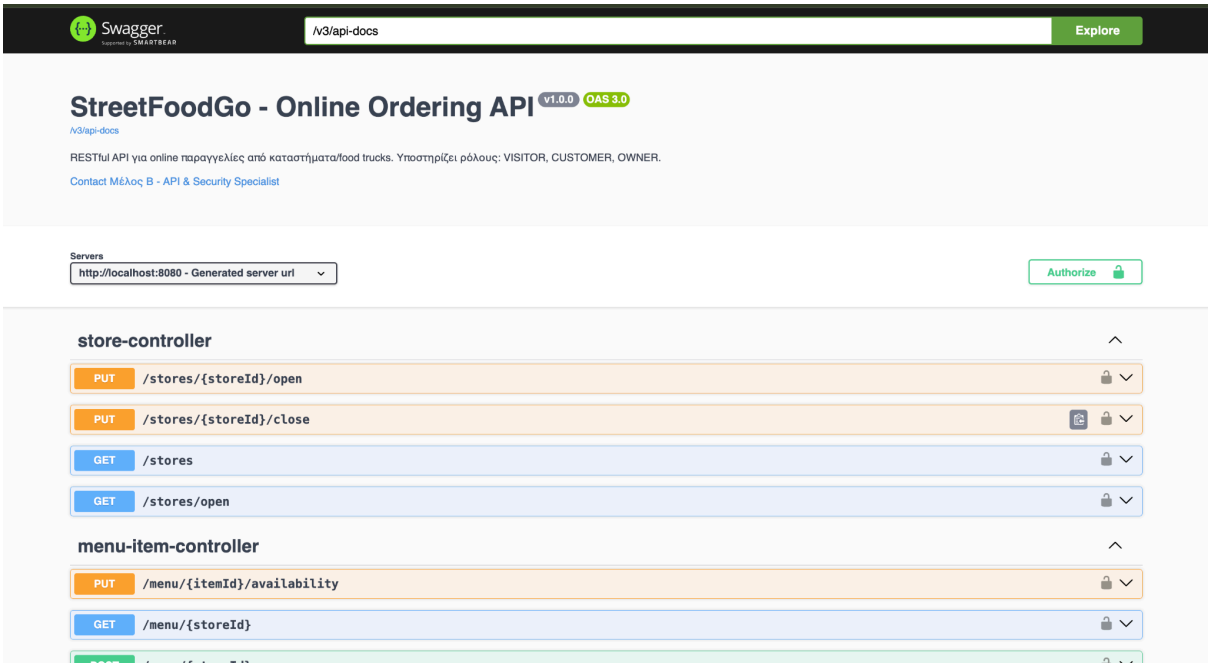
Η εφαρμογή διαθέτει πλήρως τεκμηριωμένο REST API σύμφωνα με το πρότυπο **OpenAPI 3.0**.

- Έχει ενσωματωθεί η βιβλιοθήκη `springdoc-openapi`.
- Η τεκμηρίωση είναι προσβάσιμη δυναμικά μέσω **Swagger UI** στη διεύθυνση `/swagger-ui/index.html`.
- Μέσω του Swagger UI, οι developers μπορούν να δουν όλα τα διαθέσιμα endpoints (π.χ. `POST /api/orders`, `GET /stores`), τα μοντέλα δεδομένων (DTOs) και να εκτελέσουν δοκιμαστικά αιτήματα.

Βασικά REST Endpoints που υλοποιήθηκαν:

1. `POST /api/orders`: Δημιουργία νέας παραγγελίας (δέχεται JSON `OrderRequestDto`).
2. `GET /stores`: Ανάκτηση λίστας καταστημάτων.
3. `PUT /stores/{id}/open`: Ενημέρωση κατάστασης καταστήματος.




Swagger
/v3/api-docs
Explore
StreetFoodGo - Online Ordering API
v1.0.0
OAS 3.0
/v3/api-docs
RESTful API για online παραγγελίες από καταστήματα/food trucks. Υποστηρίζει ρόλους: VISITOR, CUSTOMER, OWNER.
Contact Μέλος B - API & Security Specialist
Servers
http://localhost:8080 - Generated server url
Authorize
store-controller
PUT /stores/{storeId}/open
PUT /stores/{storeId}/close
GET /stores
GET /stores/open
menu-item-controller
PUT /menu/{itemId}/availability
GET /menu/{storeId}