Документация на приложението BiteBunny

1. Обща информация за приложението

Приложението " **BiteBunny** " е уеб-базирана платформа, създадена да свързва клиенти, ресторанти, доставчици и администратори в единна, сигурна и интуитивна система за поръчка и доставка на храна. Основната цел на системата е да автоматизира процеса от избора на храна до доставката ѝ, като същевременно осигурява отлична потребителска и административна поддръжка.

Технологичен стек

- Back-end: Java(Spring Boot)
- Front-end: Angular
- База данни: MySQL
- Контейнери: Docker, Docker Compose

Стартиране на проекта

- Цялостно стартиране: docker-compose up --build
- Frontend локално: npm install && ng serve

2. Потребителски роли и функционалности

2.1. Клиент

Ролята на клиента е централна за приложението. Системата му предлага пълен достъп до ресторанти, менюта и поръчки.

Основни функционалности:

- Регистрация и вход (вкл. с Google/Facebook)
- Преглед на ресторанти и добавяне в любими
- Филтрация по категории и търсене по име
- Преглед на менюта и артикули (с описание, размер, цена)
- Поръчка и плащане (наложен платеж или карта)

- Проследяване на поръчка в реално време
- Профил и история на поръчки
- Оценки и ревюта на ресторанти и храни

2.2 Служител на ресторант

Потребител с тази роля управлява съдържанието и процесите на конкретен ресторант.

Функционалности:

- Създаване и редакция на ресторантски профил
- Управление на менюта и артикули
- Приемане или отказване на поръчки
- Известяване на доставчици за готови поръчки
- Достъп до справки и статистика

2.3 Доставчик

Доставчикът е отговорен за логистиката и крайната доставка на поръчките до клиентите.

Функционалности:

- Приемане на поръчки (по първи приел принцип)
- Управление на активни и завършени доставки
- Промяна на статус на поръчка (в процес / доставена)
- Изчисляване на бонуси при достигане на оборот
- Преглед на данни за поръчка (адрес, клиент, ресторант)

2.4 Администратор

Администраторът разполага с пълни права върху системата и всички модули в нея.

Функционалности:

- Управление на всички ресторанти и потребители
- Създаване и модериране на съдържание
- Промяна на статуси и права
- Генериране на справки, отчети и статистика
- Поддръжка на цялостната инфраструктура

3. Сигурност и достъп

- JWT базирана автентикация за всички чувствителни API заявки
- Ролева базирана авторизация (RBAC)
- Шифроване на пароли и защита от brute-force атаки
- Интеграция с OAuth (Google / Facebook)

4. Поддръжка и разширяемост

- Микросървисен подход чрез модулна архитектура на контролерите
- Docker-контейнери позволяват лесно мащабиране
- RESTful API с ясна структура, готова за мобилна интеграция
- Възможност за добавяне на нови роли (напр. оператор, маркетинг и т.н.)

API документация

Технологична архитектура

- Back-end: Java(Spring Boot)
- Front-end: Angular
- База данни: MySQL
- Контейнеризация: Docker (стартиране чрез docker-compose)
    - Стартиране на всичко: docker-compose up --build
    - Стартиране само на frontend: npm install && ng serve

API контролери

AdminFoodController

Базов URL: /api/admin/food

Метод	URL	Описание	Headers / Тяло	Отговор
POST	/	Създаване на храна	Authorization, CreateFoodRequest	201 Created
PUT	/{foodId}	Актуализация на наличност на храна	Authorization	200 OK
DELETE	/{foodId}	Изтриване на храна	Authorization	200 OK

AdminRestaurantController

Базов URL: /api/admin/restaurants

Метод	URL	Описание	Headers / Тяло	Отговор
POST	/	Създаване на ресторант	Authorization, CreateRestaurantRequest	201 Created
PUT	/{id}	Актуализация на ресторант	Authorization, CreateRestaurantRequest	200 OK

PUT	/{id}/status	Смяна на статус на ресторант	Authorization	200 OK
DELETE	/{id}	Изтриване на ресторант	Authorization	200 OK

AuthController

Базов URL: /auth

Метод	URL	Описание	Request Body / Headers	Отговор
POST	/register	Регистрация на нов потребител	RegisterRequest	201 Created / 400, 500
POST	/login	Вход на потребител	LoginRequest	200 OK / 401, 500
POST	/refresh	Обновяване на JWT	RefreshTokenRequest	200 OK / 401

CartController

Базов URL: /api/cart

Метод	URL	Описание	Headers / Тяло	Отговор
GET	/	Връща количката на потребителя	Authorization	200 OK / 404
POST	/items	Добавяне на продукт в количката	Authorization, CartItemRequest	200 OK
PUT	/items/{itemId}	Актуализация на количество	Authorization, CartItemRequest	200 OK
DELETE	/items/{itemId}	Премахване на продукт от количката	Authorization	200 OK
DELETE	/	Изчистване на количката	Authorization	204 No Content

FoodController

Базов URL: /api/food

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/search	Търсене на храна по име	Authorization, foodName=...	200 OK
GET	/menu/{menuId}	Извличане на храни по меню и категория	Authorization, foodCategory	200 OK

HomeController

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/api/home	Данни за начална страница	—	200 OK

MenuController

Базов URL: /api/menus

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/restaurant/{restaurantId}	Взима меню на ресторант		200 OK
POST	/restaurant/{restaurantId}	Създава меню за ресторант	Menu	200 OK
PUT	/{menuId}	Обновява меню	Menu	200 OK
DELETE	/{menuId}	Изтрива меню		200 OK
POST	/{menuId}/foods/{foodId}	Добавя храна към меню		200 OK

DELETE	/{menuId}/foods/{foodId}	Премахва храна от меню		200 OK

OrderController

Базов URL: /api/orders

Метод	URL	Описание	Headers / Параметри	Отговор
POST	/	Създаване на нова поръчка	Authorization, CreateOrderRequest	201 Created или 500
GET	/user	Поръчки на текущия потребител	Authorization, page, size	200 OK или 500
GET	/{orderId}	Извличане на конкретна поръчка	Authorization	200 OK или 404
PUT	/{orderId}/status	Актуализиране на статус на поръчка	Authorization, status параметър	200 OK или 400

PaymentController

Базов URL: /api/payments

Метод	URL	Описание	Headers / Параметри	Отговор
POST	/	Обработва плащане	Authorization, PaymentRequest	201 Created или 400
GET	/{paymentId}	Извлича плащане по ID	Authorization	200 OK или 404
GET	/order/{orderId}	Извлича плащане по ID на поръчка	Authorization	200 OK или 404

POST	/refund/{paymentId}	Възстановява плащане (refund)	Authorization	200 OK или 400

RestaurantController

Базов URL: /api/restaurants

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/	Връща всички ресторанти	Authorization	200 OK
GET	/{id}	Връща ресторант по ID	Authorization	200 OK или 404
GET	/search?keyword=...	Търси ресторанти по ключова дума	Authorization, keyword	200 OK
PUT	/{id}/add-favourites	Добавя ресторант към любими	Authorization, id	200 OK

ReviewController

Базов URL: /api/reviews

Метод	URL	Описание	Headers / Параметри	Отговор
POST	/	Създава ново ревю	Authorization, JSON тяло: CreateReviewRequest	201 Created
GET	/restaurant/{restaurantId}	Взима ревюта за ресторант	Query: page, size	200 OK
GET	/food/{foodId}	Взима ревюта за храна	Query: page, size	200 OK

PUT	/{reviewId}	Актуализира ревю	Authorization, JSON тяло: CreateReviewRequest	200 OK
DELETE	/{reviewId}	Изтрива ревю	Authorization	204 No Content

TestController

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/api/test	Тества дали приложението работи	Няма	200 OK (текст: "Application is running!")

UserController

Базов URL: /api/users

Метод	URL	Описание	Headers / Параметри	Отговор
GET	/profile	Връща потребителския профил по JWT токен	Authorization: Bearer <token>	200 OK, 401 Unauthorized

UserProfileController

Базов URL: /api/user/profile

Метод	URL	Описание	Headers / Тяло	Отговор
GET	/	Извлича профил на текущия потребител по JWT	Authorization: Bearer <token>	200 OK, 404 Not Found

PUT	/	Актуализира профила на потребителя	Authorization: Bearer <token>, JSON: UpdateProfileRequest	200 OK, 400 Bad Request
PUT	/password	Промяна на парола	Authorization: Bearer <token>, JSON: UpdatePasswordRequest	200 OK, 400 Bad Request
DELETE	/	Изтрива потребителския профил	Authorization: Bearer <token>	204 No Content, 500

Security

Всеки endpoint, освен /auth/*, изисква JWT в Authorization header:

Authorization: Bearer <jwt>

JWT (JSON Web Token) се използва за удостоверяване на потребители в приложението. Той се издава при вход и се изпраща при всяка последваща заявка към защитените ресурси. Когато accessToken изтече, потребителят може да го поднови чрез:

POST /auth/refresh