Darkstore API

Software Requirements Specification

Contents

Introduction

Purpose

The purpose of this project is to create a REST API service that manages couriers, orders, and their distribution for a darkstore. The service aims to optimize the courier workflow, manage orders, and streamline the allocation process, ensuring smooth operations and timely deliveries within the darkstore ecosystem.

System restrictions

- The system will not provide real-time tracking of courier locations.
- The system will not support chat or notification features.
- The service will not support multiple languages; all data and communication will be in the same language.

Intended audience and project scope

1. **Administrators / Logistics Managers:**
   - *Function:* Responsible for managing and organizing courier operations, such as assigning orders to couriers, monitoring delivery progress, and maintaining courier schedules.
   - *Access:* Full system access, including order assignment, tracking, and data management.

2. **Couriers:**
   - *Function:* Responsible for executing the deliveries, viewing assigned orders, updating order status upon delivery, and managing their availability schedules.
   - *Access:* Restricted access limited to the orders assigned and their delivery status updates.

Potential improvements

- Real-time order tracking
- Machine learning for demand prediction
- Registration of new couriers via app

Non-Functional Requirements

- Performance requirements

  - The application will use a stable database system that can handle higher current load

  - The application will take into account the growth of users and the growing volume of stored data.

- Security requirements

  - Login credentials will be encrypted and secured

  - The system will use an external payment gateway to transfer funds

- Quality requirements

  - The design of the application will be intuitive, modern and minimalistic

  - The application will be usable without limitations on all common operating systems and browsers

References

https://gitlab.fel.cvut.cz/B231_B6B36EAR/morenmat

Class diagram

Only the highlighted (blue) lines in the class diagram will be implemented.

WorkingHour
- startTime : LocalTime
- endTime : LocalTime

«enumeration» Role
ROLE_COURIER, ROLE_MANAGER

«use»

User
- email : String
- password : String
- role : Role

Extends

«enumeration» CourierType
FOOT, BIKE, AUTO

«use»

0..*

works in interval

1

Courier
- type : CourierType
- orders : List<Order>
- regions : List<CourierRegion>
- workingHours : List<TimeInterval>

+ findMaxOrdersCount() : Integer
+ findMaxOrdersWeight()
+ getMultiplierForRating()
+ getMultiplierForEarnings()

Manager

+ addOrder() : void
+ assignOrder() : void
+ createCouriers()

0..*

manages

0..*

CourierRegion
- region : Integer

0..*

1

GroupOrder
- deliveryDate : LocalDate
- limit : Limit
- courier : Courier
- orders : List<Order>

0..1

1

1..*

«enumeration» PaymentStatus
PENDING, DECLINED, APPROVED, NOT_PAID

«use»

PaymentInfo
- date: LocalDateTime
- status: PaymentStatus

Order
- weight : Float
- region : Region
- cost : Integer
- completedTime : LocalDateTime
- status : OrderStatus
- deliveryHours : List<DeliveryHour>
- courier : Courier
- groupOrder : GroupOrder

+ completeOrder()

0..*

«use»

«enumeration» OrderStatus
NEW, ASSIGNED, FINISHED

BankGateway

1

0..*

DeliveryHour
- start : LocalTime
- finish : LocalTime