

内部設計書

香美市特化型配達サービス

Stellar Delivery

第 1 版

2025 年 12 月 2 日

StellarWorks

目次

| | | |
|-----|---------------|---|
| 1 | 概要 | 2 |
| 2 | 動作環境 | 2 |
| 3 | 開発環境 | 2 |
| 3.1 | Flutter | 2 |
| 3.2 | FastAPI | 2 |
| 4 | コーディング規約 | 3 |
| 5 | モジュール設計 | 3 |
| 6 | API 設計 | 3 |
| 7 | API 設計 | 3 |
| 7.1 | 認証 (Auth) | 3 |
| 7.2 | 商品 (Products) | 4 |
| 7.3 | 注文 (Orders) | 4 |
| 7.4 | 配達 (Delivery) | 4 |
| 8 | データベース設計 | 6 |
| 8.1 | 主要テーブル詳細 | 6 |

1 概要

この文書は、Stellar Delivery アプリケーションの内部設計について記述したものである。本アプリケーションは、フロントエンドに Flutter、バックエンドに FastAPI を用いて開発されている。5 章 モジュール設計では、Flutter アプリケーションで用いられるクラスの設計について記述する。6 章 API 設計では、FastAPI で実装された API の入出力について記述する。

2 動作環境

本アプリケーション Stellar Delivery は以下の環境で動作することを想定している。

- iOS
- Android
- Web (管理者・店舗向け機能の一部)

3 開発環境

3.1 Flutter

Flutter の開発には、表 1 の環境を用いる。

表 1 Flutter 開発環境

| 項目 | 内容 | 説明 |
|-------------|--------------------|---------------|
| 使用するフレームワーク | Flutter | |
| プログラミング言語 | Dart | |
| バージョン管理 | Git | |
| 使用する OS | Windows, Mac | |
| 使用するエディタ | Visual Studio Code | |
| 拡張機能 | Flutter | Flutter 開発に必要 |

3.2 FastAPI

FastAPI の開発には、表 2 の環境を用いる。

表 2 FastAPI 開発環境

| 項目 | 内容 | 説明 |
|--------------|-----------------------|--------------------------|
| 使用するフレームワーク | FastAPI | |
| プログラミング言語 | Python | |
| バージョン管理 | Git | |
| コンテナプラットフォーム | Docker | |
| 使用する OS | Windows, Mac | |
| 使用するエディタ | Visual Studio Code | |
| 拡張機能 | Python Extension Pack | Python で必要な拡張機能をまとめたもの |
| | Black | Python のフォーマッタ |
| | Flake8 | コーディング規約を強制するもの |
| | isort | import を自動的に並べ替えしてくれる |
| | Dev Containers | コンテナ上で VSCode を実行するために必要 |

4 コーディング規約

5 モジュール設計

6 API 設計

7 API 設計

バックエンド（FastAPI）で実装する主な API エンドポイントを記述する。

7.1 認証 (Auth)

ログイン

POST /auth/login

- **Request:** email, password
- **Response:** access_token, token_type, user_role

新規登録

POST /auth/register

- **Request:** email, password, user_type, profile_data(name, address, etc.)
- **Response:** user_id, message

7.2 商品 (Products)

店舗別商品一覧取得

GET /stores/{store_id}/products

- **Request:** store_id (path)
- **Response:** list[product_object]

商品登録 (店舗のみ)

POST /products

- **Request:** token, name, price, description, image, stock_status
- **Response:** product_id, message

7.3 注文 (Orders)

注文作成

POST /orders

- **Request:** token, store_id, items[product_id, quantity], total_price
- **Response:** order_id, message

注文ステータス更新

PUT /orders/{order_id}/status

- **Request:** token, status (prepared, delivering, delivered)
- **Response:** message

7.4 配達 (Delivery)

配達可能案件一覧取得

GET /delivery/jobs

- **Request:** token, current_location
- **Response:** list[order_object]

案件受注

POST /delivery/jobs/{order_id}/accept

- **Request:** token, order_id

- **Response:** message

8 データベース設計

以下に、本アプリケーションにおける主要なデータベース設計を示す。

表 3 テーブル一覧

| テーブル名 | 説明 |
|--------------------|------------------------------|
| users | 全ユーザ（依頼者・配達員・店舗・管理者）の基底情報 |
| requester_profiles | 依頼者の詳細情報（住所、クレジットカード情報等） |
| deliverer_profiles | 配達員の詳細情報（履歴書、稼働状況、口座情報） |
| store_profiles | 店舗の詳細情報（営業許可証、営業時間、口座情報） |
| products | 商品情報 |
| orders | 注文情報のヘッダー（合計金額、ステータス、依頼者 ID） |
| order_details | 注文明細（商品 ID、数量） |
| deliveries | 配達情報（担当配達員 ID、位置情報ログ） |
| notifications | 通知情報 |

8.1 主要テーブル詳細

users テーブル

| カラム名 | 制約 | 型 | 説明 |
|------------|--------|----------|-------------------------------------|
| id | PK | int | ユーザー ID |
| email | NN, UQ | varchar | メールアドレス |
| password | NN | varchar | ハッシュ化されたパスワード |
| role | NN | char | 役割（requester/deliverer/store/admin） |
| created_at | | datetime | 作成日時 |

orders テーブル

| カラム名 | 制約 | 型 | 説明 |
|--------------|----|----------|---------------|
| id | PK | int | 注文 ID |
| requester_id | FK | int | 依頼者 ID |
| store_id | FK | int | 店舗 ID |
| deliverer_id | FK | int | 配達員 ID（受注後設定） |
| status | NN | varchar | 注文ステータス |
| total_price | NN | int | 合計金額 |
| ordered_at | | datetime | 注文日時 |

deliveries テーブル

| カラム名 | 制約 | 型 | 説明 |
|---------------|----|----------|----------|
| id | PK | int | 配達 ID |
| order_id | FK | int | 注文 ID |
| pickup_time | | datetime | 商品受け取り時間 |
| delivery_time | | datetime | 配達完了時間 |
| status | NN | varchar | 配達状況 |