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 11 日

StellarWorks

目次

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

本章では、Flutter の開発における Dart の命名規約およびコーディングスタイルを示す。

4.1　ケース（表記形式）

表 3 に命令時に用いるケースの定義を示す。

表 3　命名時に用いるケースの定義

ケース	説明	例
lowerCamelCase	先頭小文字, 単語区切り大文字	userName
UpperCamelCase	先頭大文字, 単語区切り大文字	UserProfile
snake_case	小文字, アンダースコアつなぎ	user_profile
UPPER_SNAKE_CASE	大文字, アンダースコアつなぎ	MAX_COUNT

4.2　Dart の命名規約

表 4 に Dart における対象ごとの命名規約を示す。
以下の点に注意する。

- private でない識別子をアンダースコア（_）から始めないこと。
- ハンガリアン記法（mUser, kUser など）は使用しない。
- 未使用コールバック引数には _（または複数個の _ ）を使用する。

表 4　Dart の命名規約

対象	ケース	例
モジュール名/パッケージ名	snake_case	user_profile, my_package
ファイル名/ディレクトリ名	snake_case	user_detail_page.dart
クラス名	UpperCamelCase	UserModel
メソッド/関数名	lowerCamelCase	getUserData()
変数名	lowerCamelCase	userCount
定数名	lowerCamelCase	maxRetryCount

4.3　コーディングスタイル

Dart のコーディングスタイルは Effective Dart に従う。以下に主な規約を記載する。

4.3.1　基本スタイル

- インデントは 2 スペースとする。
- 空行にはインデントを入れない（スペースを入れない）。
- オブジェクト生成時に new は記述しない。const は必要な場合のみ使用する。
- 再代入しない変数は final として宣言する。

4.3.2　Widget（Flutter）記述ルール

- 引数が 2 つ以上ある Widget は複数行で書く。
- Widget の引数が 1 つでも、その子 Widget が複数行の場合は親も改行する。
- Widget リストは末尾カンマ（trailing comma）をつけ、自動整形に従う。
- EdgeInsets は複数引数でも 1 行でよい。

4.3.3　制御構文

- if, for, while などの制御構文は中括弧 { } を省略しない。
- ただし、if と else がいずれも 1 行で完結する場合のみ、省略を許容する。

4.3.4　コメント

- // の後には半角スペースを 1 つ置く。
- 意図が明確なコードに対して不要なコメントを記述しない。

4.3.5 import / export 規約

Dart における import 文は、読みやすさと依存関係の明確化のため、以下の順序で記述するものとする。

1. **dart:** 標準ライブラリ
2. **package:** 外部パッケージ
3. **package:（自プロジェクト）** アプリ内 lib/ を package import で参照
4. **相対パス:** ../ や ./ を用いた参照

その他の注意点：

- import は辞書順で並べる。
- export は import とは別のグループとして最後にまとめる。
- 可能な限り相対 import ではなく package import を使用する。

4.3.6 書式

- 原則として 1 行 80 文字以内とする。
- dart の自動 formatter に従う。

4.4 ツール設定

以下のツールを導入し、自動で規約を適用できるようにする。

- VSCode の ”Format on Save” を有効にする。
- `dart fix --apply` により自動修正を行う。
- `flutter analyze` でコーディング規約違反を確認する。

モジュール名	withdraw.dart	
モジュール概要	退会を確定するかどうかを確認するオーバーレイ	
責任者	梶本和希	
構成要素	クラス名	Withdraw
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Maaterial
	他クラスとの関係	
	withdraw.dart の Withdraw().show(context) を呼び出すとオーバレイ表示 キャンセルが押されると Withdraw().hide() により controlle.close が呼び出されオーバレイを閉じる 退会を押すと退会処理後、Withdraw().hide() により controlle.close が呼び出されオーバレイを閉じる	

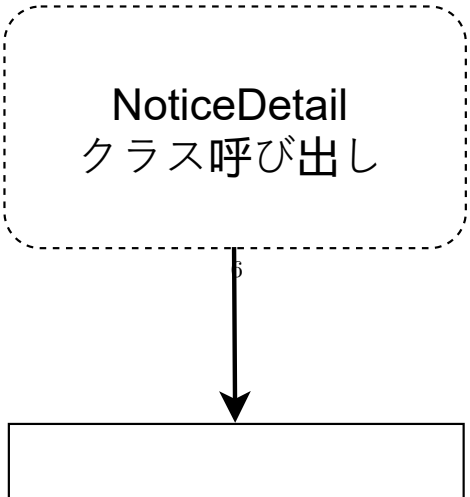
モジュール名	logout.dart	
モジュール概要	ログアウトを確定するかどうかを確認するオーバーレイ	
責任者	岡林磨目	
構成要素	クラス名	Logout
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Material
	他クラスとの関係	
	logout.dart の Logout().show(context) を呼び出すとオーバーレイ表示 キャンセルを押すと Logout().hide() により controlle.close が呼び出されオーバレイを閉じる ログアウトを押すことで login_page.dart の LoginPage クラスを呼び出すことで、ログイン画面を表示	

5 モジュール設計

5.1 overlay

5.1.1 共通機能

5.1.2 配達員



モジュール名	rule_screen.dart	
モジュール概要	利用規約を表示するオーバーレイ	
責任者	岡林磨目	
構成要素	クラス名	RuleScreen
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Maaterial
	他クラスとの関係	
	rule_screen.dart の RuleScreen().show(context) を呼び出すとオーバーレイ表示 閉じるを押すと RuleScreen().hide() により controlle.close が呼び出されオーバレイを閉じる	

モジュール名	cancel_inf_change.dart	
モジュール概要	情報の変更をキャンセルするかを確認するオーバーレイ	
責任者	岡林磨目	
構成要素	クラス名	CancelInfChange
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Maaterial
	他クラスとの関係	
	cancel_inf_change.dart の CancelInfChange().show(context) を呼び出すとオーバレイ表示 いいえを押すと CancelInfChange().hide() により controlle.close が呼び出されオーバレイを閉じる はいを押すと変更キャンセル処理後、CancelInfChange().hide() により controlle.close が呼び出されオーバレイを閉じる	

5.2 provider

5.3 provider

5.4 page

5.4.1 共通

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。

モジュール名	confirm_inf_change.dart	
モジュール概要	情報の変更を確定するかを確認するオーバーレイ	
責任者	岡林磨目	
構成要素	クラス名	ConfirmInfChange
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Maaterial
	他クラスとの関係	
	confirm_inf_change.dart の ConfirmInfChange().show(context) を呼び出すとオーバレイ表示 いいえを押すと ConfirmInfChange().hide() により controlle.close が呼び出されオーバレイを閉じる はいを押すと変更確定処理後、ConfirmInfChange().hide() により controlle.close が呼び出されオーバレイを閉じる	

モジュール名	rule_screen.dart	
モジュール概要	利用規約を表示するオーバーレイ	
責任者	岡林磨目	
構成要素	クラス名	RuleScreen
	クラス種類	OverEntry
	処理概要	Overlay の内容を返す
	入力	Buildcontext context
	出力	Maaterial
	他クラスとの関係	
	rule_screen.dart の RuleScreen().show(context) を呼び出すとオーバーレイ表示 閉じるを押すと RuleScreen().hide() により controlle.close が呼び出されオーバレイを閉じる	

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

モジュール名	change_user_role.dart	
モジュール概要	アプリ全体で使用するプロバイダモジュール	
責任者	岡林磨目	
構成要素	クラス名	ChangeUserRole
	クラス種類	ChangeNotifier
	処理概要	アプリ全体で使用する依頼側・配達側・店舗側の判断を行う int 型変数を定義している。また、値が変更されると、それに対応した色 (UI) へ変更が行われる。
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	

モジュール名	change_root_page.dart	
モジュール概要	アプリ全体を包括するモジュール	
責任者	岡林磨目	
構成要素	クラス名	ChangeRootPage
	クラス種類	extends StatelessWidget
	処理概要	アプリの大まかな区別である LoginRoute と RootRoute を包括しているクラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	app_router.gr.dart の LoginRoute クラスを組み込む	

- **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

モジュール名	new_member.dart	
モジュール概要	新規会員登録画面を生成するモジュール	
責任者	梶本和希	
構成要素	クラス名	NewMember
	クラス種類	extends StatelessWidget
	処理概要	新規会員登録画面の Widget を動的に変更 するクラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	new_member.dart の NewMenberState クラスを組み込む	
構成要素	クラス名	NewMemberState
	クラス種類	extends StatelessWidget
	処理概要	新規会員登録画面を表示するクラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	rule_screen.dart の RuleScreen クラスを呼び出すことで利用規約を表示 login_page.dart の LoginPage クラスを呼び出しログイン画面を表示 戻るボタンの選択で前のモジュールに戻る	

- **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。

8.1　主要テーブル詳細

users テーブル
orders テーブル
deliveries テーブル

モジュール名	mypage.dart	
モジュール概要	マイページ画面を表示するモジュール	
責任者	岡林磨目	
構成要素	クラス名	MyPageRouterPage
	クラス種類	extends AppRouter
	処理概要	MyPage にアクセスするための Router
	入力	なし
	出力	Route
	他クラスとの関係	
	なし	
構成要素	クラス名	MyPage
	クラス種類	extends StatelessWidget
	処理概要	マイページ画面を動的に変更するクラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	MyPage.dart の MyPageState クラスを組み込む	
構成要素	クラス名	MyPageState
	クラス種類	extends StatelessWidget
	処理概要	マイページ画面の状態を返すクラス
	入力	なし
	出力	State
	他クラスとの関係	
	title_appbar.dart の TitleAppBar クラスを組み込む mypage.dart の ScrollMyPageDetail クラスを組み込む	
構成要素	クラス名	
	クラス種類	
	処理概要	
	入力	
	出力	
	他クラスとの関係	
構成要素	クラス名	
	クラス種類	
	処理概要	
	入力	
	出力	
	他クラスとの関係	

モジュール名	login_page.dart	
モジュール概要	ログイン画面を生成するモジュール	
責任者	梶本和希	
構成要素	クラス名	LoginPage
	クラス種類	extends StatelessWidget
	処理概要	ログイン画面の Widget を動的に変更する クラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	login_page.dart の LoginPageState クラスを組み込む	
構成要素	クラス名	LoginPageState
	クラス種類	extends StatelessWidget
	処理概要	ログイン画面の状態を返すクラス
	入力	なし
	出力	State
	他クラスとの関係	
	new_member.dart の NewMember クラスを呼び出すことで、新規会員登録画面を表示 login_page.dart の LoginAppBar クラスを組み込む	
構成要素	クラス名	LoginAppBar
	クラス種類	extends StatelessWidget,imprementsPref- ferdSizeWidget
	処理概要	ログイン画面用に作成を行った AppBar を 生成するコンポーネント
	入力	ChangeGeneralCorporationstore
	出力	Widget
	他クラスとの関係	
	なし	

モジュール名	logout_withdraw_page.dart	
モジュール概要	ログアウト・退会画面を生成するモジュール	
責任者	梶本和希	
構成要素	クラス名	LogoutWithdraw
	クラス種類	extends StatelessWidget
	処理概要	ログアウト・退会画面を生成するクラス
	入力	なし
	出力	Widget
	他クラスとの関係	
	withdraw.dart の Withdraw クラスの呼び出すことで退会オーバーレイを表示 logout.dart の Logout クラスの呼び出すことでログアウトオーバーレイを表示戻るボタンの選択で前のモジュールに戻る	

表 5　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

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

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

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