

内部設計書

香美市特化型配達サービス

Stellar Delivery

第 2 版

2025 年 12 月 21 日

StellarWorks

目次

1	概要	3
2	動作環境	3
3	開発環境	3
3.1	Flutter	3
3.2	FastAPI	3
4	コーディング規約	4
4.1	ケース（表記形式）	4
4.2	Dart の命名規約	4
4.3	コーディングスタイル	5
4.3.1	基本スタイル	5
4.3.2	Widget（Flutter）記述ルール	5
4.3.3	制御構文	5
4.3.4	コメント	5
4.3.5	import / export 規約	6
4.3.6	書式	6
4.4	ツール設定	6
5	モジュール設計	6
5.1	overlay	9
5.1.1	共通機能	9
5.1.2	配達側	14
5.2	provider	14
5.3	page	16
5.3.1	共通機能	16
5.3.2	依頼側	22
5.3.3	店舗側	36
5.3.4	配達側	52
5.4	component	52
6	API 設計	56
6.1	認証 (Auth)	56
6.2	商品 (Products)	57
6.3	注文 (Orders)	57
6.4	配達 (Delivery)	57
7	データベース設計	58
7.1	テーブル一覧	58

7.2	主要テーブル詳細	58
-----	----------------	----

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。

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

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 _ (または複数の _) を使用する。

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。

5 モジュール設計

以下に本アプリケーションのモジュール設計を示す。

- `overlay` は、アプリケーション上で一番上のレイヤーに表示されるウィジェットをまとめたモジュールについて記述している。
- `provider` は、アプリケーション全体で使用する共通の変数をまとめたモジュールについて記述

している。

- page は、アプリケーションの画面をまとめたモジュールについて記述している。
- component は、page で用いられるパーツをまとめたモジュールについて記述している。

overlay モジュール	
モジュール名	概要
<共通機能>	
logout.dart	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
withdraw.dart	退会を確定するかどうかを確認するオーバーレイ
cancel_inf_change.dart	情報の変更をキャンセルするかどうかを確認するオーバーレイ
confirm_inf_change.dart	情報の変更を確定するかどうかを確認するオーバーレイ
rule_screen.dart	利用規約を表示するオーバーレイ
<依頼側>	
<配達側>	
d_payslip_change_date.dart	配達員の給与明細の表示される期間を変更するモジュール
<店舗側>	

表 5 overlay モジュールの構成

provider モジュール	
モジュール名	概要
change_user_role.dart	アプリ全体で使用するプロバイダモジュール
app_router.gr.dart	Router を Router に変換するモジュール

表 6 provider モジュールの構成

page モジュール	
モジュール名	概要
<共通機能>	
change_root_page.dart new_member.dart mypage.dart login_page.dart logout_withdraw_page.dart	アプリ全体を包括するモジュール 新規会員登録画面を生成するモジュール マイページ画面を表示するモジュール ログイン画面を生成するモジュール ログアウト・退会画面を生成するモジュール
<依頼側>	
c_root_page.dart c_info.dart c_info_change.dart c_address.dart c_address_change.dart c_address_add.dart c_banking_info.dart c_banking_info_change.dart c_banking_info_check.dart c_order_history.dart c_order_history_detail.dart c_order_arrival_report.dart c_home.dart c_payment.dart c_product_list.dart c_order_check.dart c_order_confirm.dart c_cart.dart c_map_search.dart	「ホーム」、「カート」、「マップ」、「マイページ」、「通知」を包括するモジュール 依頼者の会員情報画面を表示するモジュール 依頼者の会員情報変更画面を表示するモジュール 配達先住所確認画面を表示するモジュール 配達先住所編集画面を表示するモジュール 配達先住所追加画面を表示するモジュール 口座情報管理画面を表示するモジュール 口座情報変更画面を表示するモジュール 口座情報変更を確認する画面を表示するモジュール 注文履歴一覧画面を表示するモジュール 注文履歴詳細画面を表示するモジュール 商品到着報告画面を表示するモジュール 依頼者のホーム画面を表示するモジュール 支払い明細画面を表示するモジュール 商品一覧画面を表示するモジュール 注文確認画面を表示するモジュール 注文確定画面を表示するモジュール カート情報を表示するモジュール マップでの店舗検索を行う画面を表示するモジュール
<配達側>	
d_home.dart d_notice.dart d_notice_detail.dart d_job_select.dart d_job_detail.dart d_map.dart d_banking_information.dart d_resume_detail.dart d_resume_change.dart d_payslip.dart d_delivery_history.dart d_delivery_history_detail.dart d_banking_information_change.dart d_banking_information_check.dart d_job_map_select.dart d_root_page.dart	ホーム画面を表示するモジュール 通知一覧画面を表示するモジュール 通知詳細画面を表示するモジュール 求人検索一覧画面を表示するモジュール 求人情報の詳細画面を表示するモジュール 求人情報より配達先ルートを示したマップを表示するモジュール 口座情報を示した画面を表示するモジュール 履歴書情報を示した画面を表示するモジュール 履歴書情報を変更する画面を表示するモジュール 給与明細を示した画面を表示するモジュール 配達履歴の一覧を示した画面を表示するモジュール 配達履歴詳細を示した画面を表示するモジュール 口座情報変更を行う画面を表示するモジュール 口座情報の変更を確認する画面を表示するモジュール マップにて配達員先住所(求人情報)を検索する画面を表示するモジュール 「ホーム」、「求人検索」、「マップ」、「受注」、「マイページ」を包括するモジュール
<店舗側>	
s_root_page.dart s_info.dart s_info_change.dart s_inventory_status.dart s_menu_regist.dart s_menu_change.dart s_menu_delete.dart s_menu_edit.dart s_order_list.dart s_work.dart s_home.dart s_delivery_complete.dart s_banking_info.dart	「ホーム」、「業務」、「注文管理」、「売上」、「マイページ」を包括するモジュール 店舗の会員情報画面を表示するモジュール 店舗の会員情報変更画面を表示するモジュール 店舗の在庫ステータス画面を表示するモジュール メニュー情報の新規登録画面を表示するモジュール メニュー情報の変更画面を表示するモジュール メニュー情報の削除画面を表示するモジュール メニュー情報設定画面を表示するモジュール 受注一覧画面を表示するモジュール 業務画面を表示するモジュール 店舗ホーム画面を表示するモジュール 受け渡し完了確認画面を表示するモジュール 口座情報管理画面を表示するモジュール

表 7 page モジュールの構成

component モジュール	
モジュール名	概要
<共通機能>	
finish_screen.dart	完了画面を生成するコンポーネント
over_screen_controller.dart	オーバーレイの表示、非表示の操作を行うコンポーネント
<依頼側>	
c_number_count.dart	注文数のカウントを増加(減少)するボタンを生成するレイヤー
general_form.dart	一般ユーザが入力するフォームを出力するコンポーネント
password_input.dart	パスワードを入力するフォームを作成するコンポーネント
<配達側>	
d_notice_change.dart	通知画面にて配達業務と運営メールの切り替えにて使用するモジュール
d_job.dart	求人情報を表示するモジュール(求人検索での求人を選択する際に使用)
button_set.dart	任意のテキストを記載したボタンを生成するレイヤー
<店舗側>	

表 8 component モジュールの構成

5.1 overlay

5.1.1 共通機能

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	Material
	他クラスとの関係	
	withdraw.dart の Withdraw().show(context) を呼び出すとオーバーレイ表示 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ーレイを閉じる 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ーレイを閉じる	

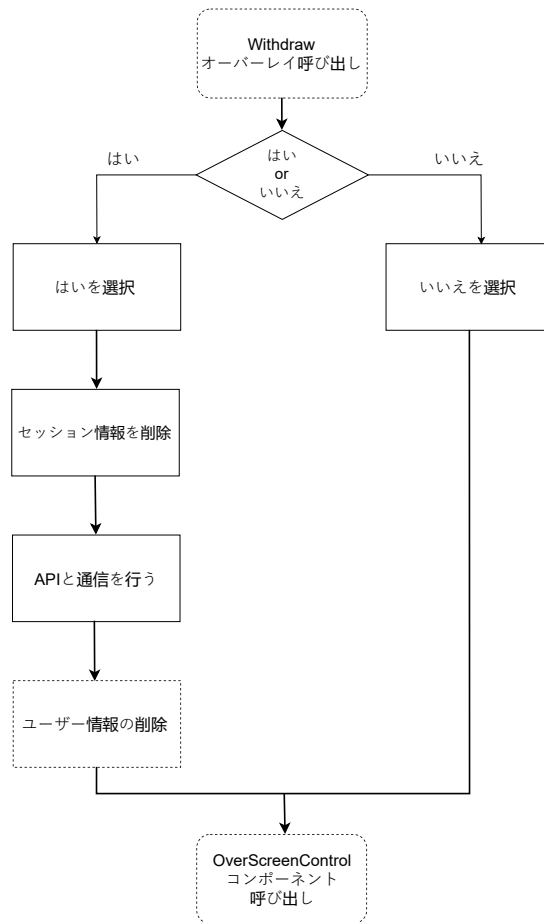


図 1 withdraw.dart の状態遷移図

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ーレイを閉じる 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	

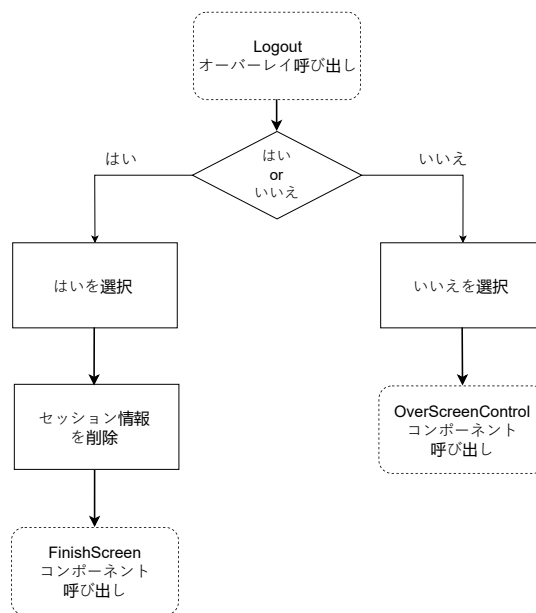


図 2 logout.dart の状態遷移図

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	Material
	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ーレイを閉じる	

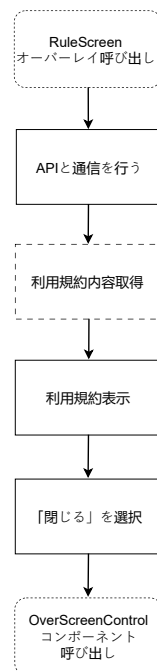


図 3 rule_screen.dart の状態遷移図

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	Material
	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ーレイ表示 いいえを押すと CancelInfChange().hide() により controlle.close が呼び出されオーバーレイを閉じる 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ーレイを閉じる	

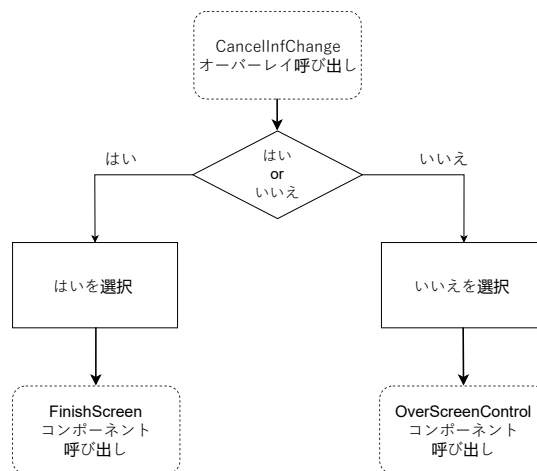


図 4 cancel_inf_change.dart の状態遷移図

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	Material
	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ーレイ表示 いいえを押すと ConfirmInfChange().hide() により controlle.close が呼び出されオーバーレイを閉じる 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ーレイを閉じる	

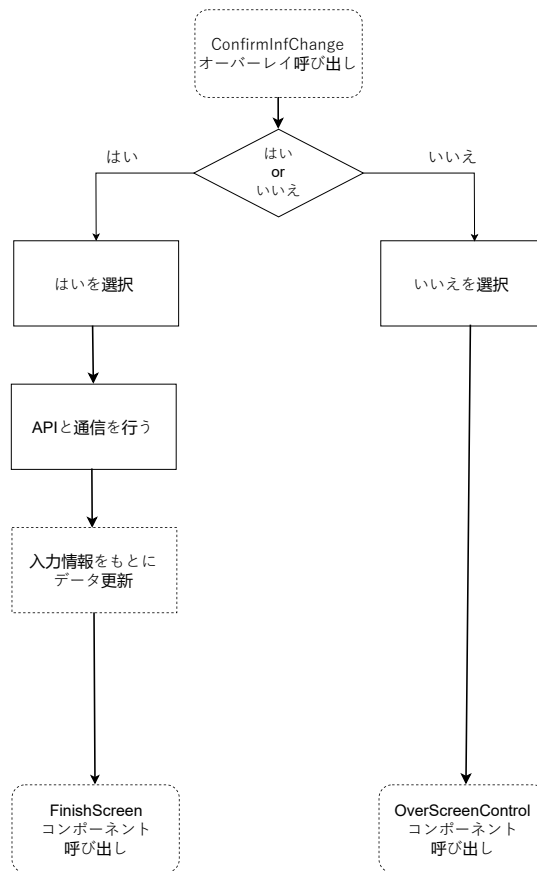


図 5 confirm_inf_change.dart の状態遷移図

5.1.2 配達側

5.2 provider

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	app_router.gr.dart	
モジュール概要	Router を Router に変換するモジュール	
責任者	岡林磨目	
構成要素	クラス名	_\$AppRouter
	クラス種類	extends RootStackRouter
	処理概要	Map に Router を入れこむクラス
	入力	なし
	出力	Map
	他クラスとの関係	
	app_router.dart を組み込む	
構成要素	クラス名	DHomeRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	DJobRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	DMapRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	DHistoryRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	CHomeRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	CCartRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし
	他クラスとの関係	
	なし	
構成要素	クラス名	CMapRouterRoute
	クラス種類	extends PageRouteInfo (void)
	処理概要	Route の名前定義
	入力	なし
	出力	なし

モジュール名	app_router.dart	
モジュール概要	Router のパス位置を定義するモジュール	
責任者	岡林磨目	
構成要素	クラス名	AppRouter
	クラス種類	extends _\$AppRouter
	処理概要	Router のパス位置を定義するクラス
	入力	なし
	出力	なし
	他クラスとの関係	
	root_page.dart を組み込む	
	c_home.dart を組み込む	
	c_cart.dart を組み込む	
	c_map_search.dart を組み込む	
構成要素	d_home.dart を組み込む	
	d_job_select.dart を組み込む	
	d_job_map_select.dart を組み込む	
	d_deliverly_history.dart を組み込む	
	s_home.dart を組み込む	
	s_work.dart を組み込む	
	s_order_list.dart を組み込む	
	s_sales.dart を組み込む	
	mypage.dart を組み込む	
	change_root_page.dart を組み込む	
構成要素	login_page.dart を組み込む	
	app_router.gr.dart を組み込む	

5.3 page

5.3.1 共通機能

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	

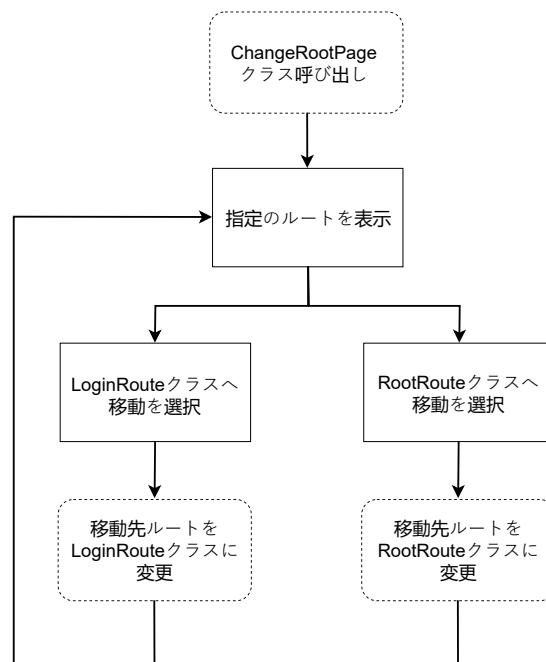


図 6 change_root_page.dart の状態遷移図

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	

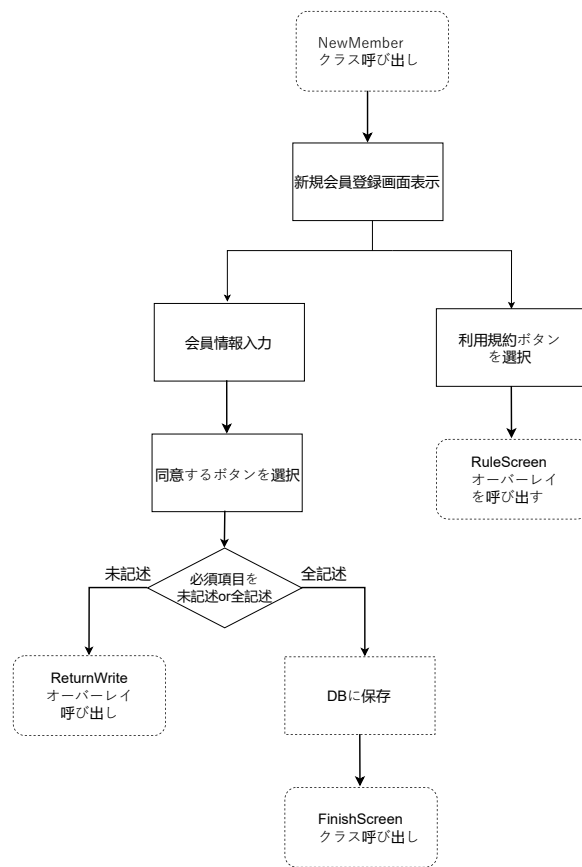


図 7 new_member.dart の状態遷移図

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	MyPageRoutePage
	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	

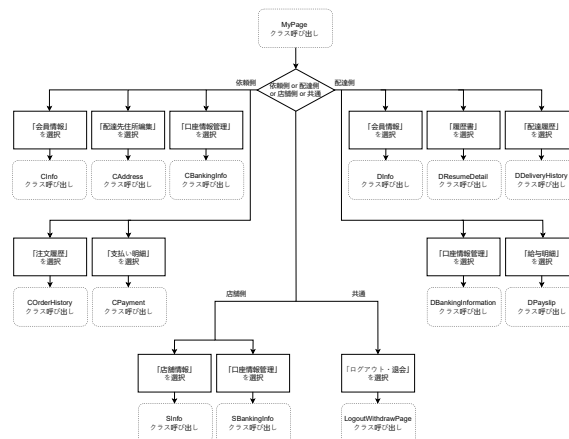


図 8 mypage.dart の状態遷移図

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, implements PreferredSizeWidget
	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	

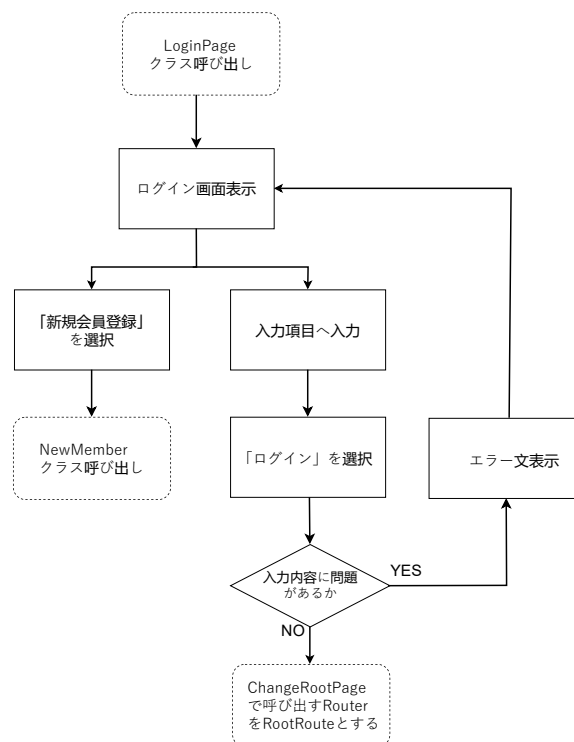


図 9 login_page.dart の状態遷移図

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	

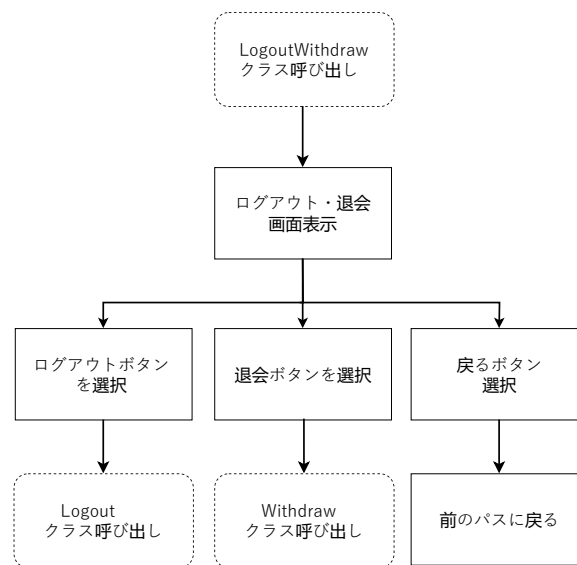


図 10 logout_withdraw_page.dart の状態遷移図

5.3.2 依頼側

モジュール名	c_map_search.dart	
モジュール概要	マップでの店舗検索を行う画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	MapSearch
	クラス種類	extends StatelessWidget
	処理概要	マップと店舗検索バーを表示する
	入力	なし
	出力	Widget
	他クラスとの関係	GoogleMap Widget を表示し、検索結果に応じてピンを配置する

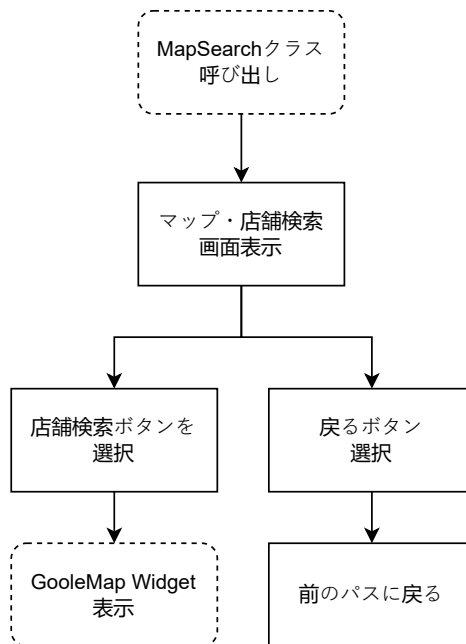


図 11 c_map_search.dart の状態遷移図

モジュール名	c_product_list.dart	
モジュール概要	商品一覧画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	ProductList
	クラス種類	extends StatelessWidget
	処理概要	選択された店舗の商品一覧を表示する
	入力	int storeId
	出力	Widget
	他クラスとの関係	
	API から商品情報を取得し、Card Widget 等でリスト表示する。商品選択でカートに追加または詳細表示を行う	

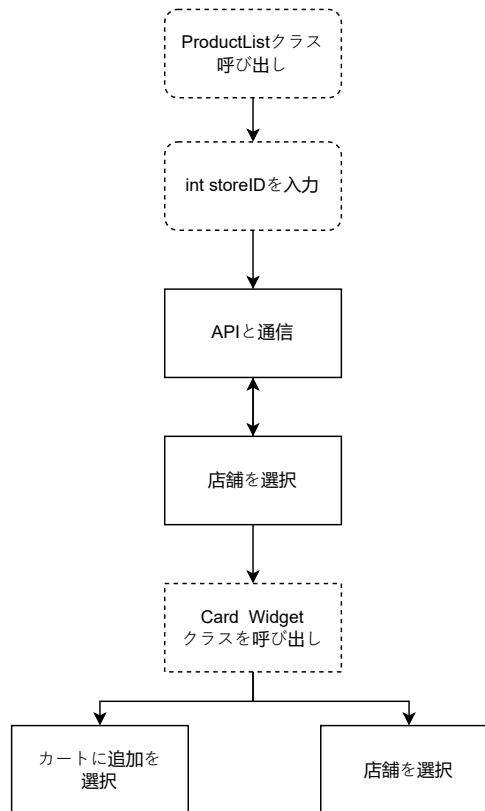
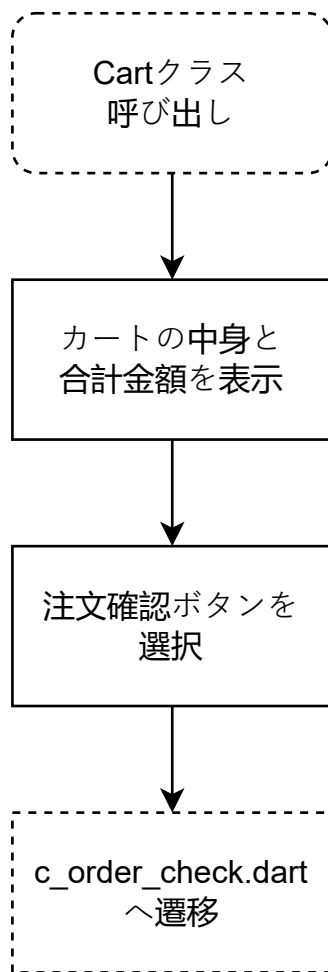


図 12 c_product_list.dart の状態遷移図

モジュール名	c_cart.dart	
モジュール概要	カート情報を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	Cart
	クラス種類	extends StatelessWidget
	処理概要	現在カートに入っている商品リストと合計金額を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	c_order_check.dart へ遷移し注文確認を行う	



モジュール名	c_order_confirm.dart	
モジュール概要	注文確定画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	OrderConfirm
	クラス種類	extends StatelessWidget
	処理概要	注文内容の最終確認と確定処理を行う
	入力	OrderData order
	出力	Widget
	他クラスとの関係	
	API へ注文データを送信し、成功時に完了画面 (finish_screen.dart) を表示する	

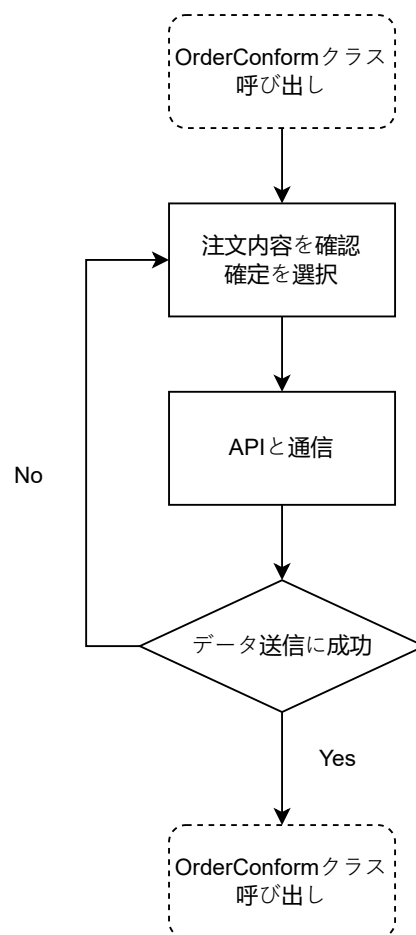


図 14 c_order_confirm.dart の状態遷移図

モジュール名	c_payment.dart	
モジュール概要	支払い明細画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	PaymentDetail
	クラス種類	extends StatelessWidget
	処理概要	注文の詳細な支払い内訳を表示する
	入力	int orderId
	出力	Widget
	他クラスとの関係	
	なし	

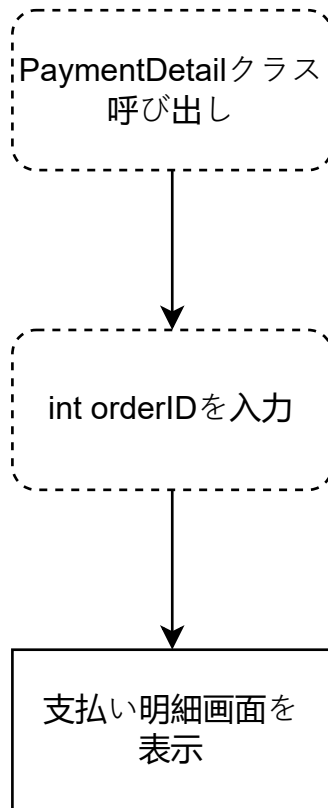
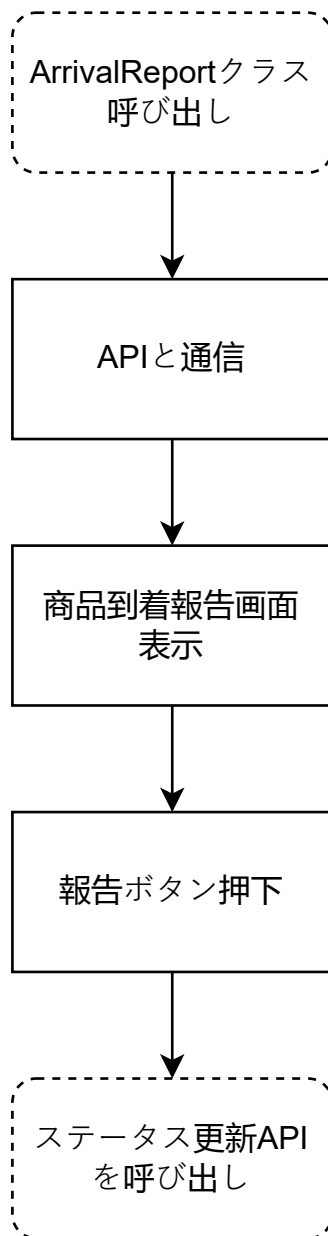
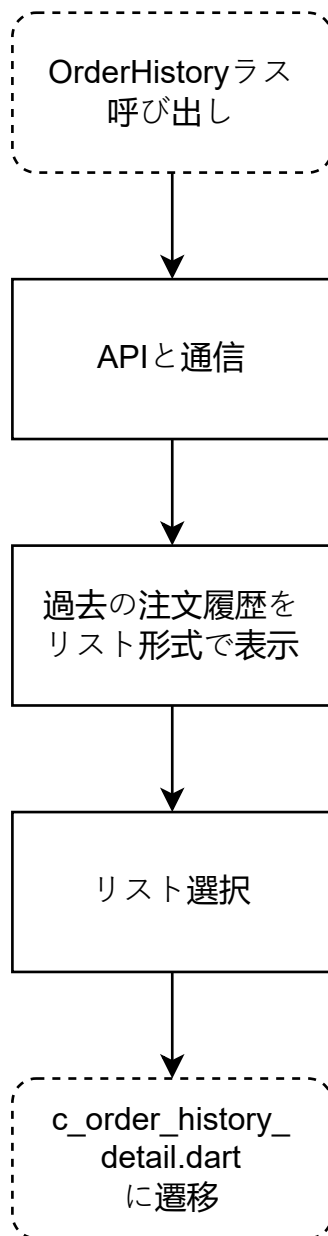


図 15 `c_payment.dart` の状態遷移図

モジュール名	c_order_arrival_report.dart	
モジュール概要	商品到着報告画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	ArrivalReport
	クラス種類	extends StatelessWidget
	処理概要	商品受け取りを確認し、完了報告を行う
	入力	int orderId
	出力	Widget
	他クラスとの関係	
	報告ボタン押下でステータス更新 API を呼び出す	



モジュール名	c_order_history.dart	
モジュール概要	注文履歴一覧画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	OrderHistory
	クラス種類	extends StatelessWidget
	処理概要	過去の注文履歴をリスト形式で表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	リスト選択で c_order_history_detail.dart へ遷移する	



モジュール名	c_address.dart	
モジュール概要	配達先住所確認画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	AddressInfo
	クラス種類	extends StatelessWidget
	処理概要	登録されている配達先住所を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	編集ボタンで c_address_change.dart、追加ボタンで c_address_add.dart へ遷移する	

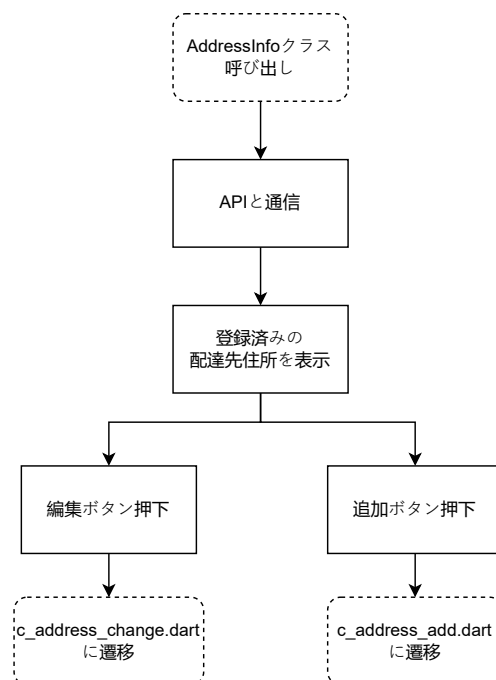
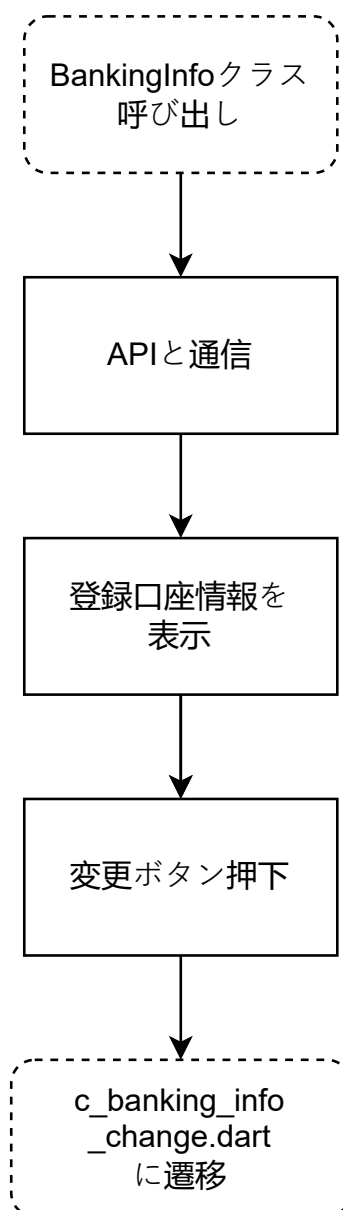


図 18 c_address.dart の状態遷移図

モジュール名	c_banking_info.dart	
モジュール概要	口座情報管理画面を表示するモジュール	
責任者	工藤 愛	
構成要素	クラス名	BankingInfo
	クラス種類	extends StatelessWidget
	処理概要	登録口座情報を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	変更ボタンで c_banking_info_change.dart へ遷移する	



5.3.3 店舗側

モジュール名	s_root_page.dart	
モジュール概要	「ホーム」、「業務」、「注文管理」、「売上」、「マイページ」を包括するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SRootPage
	クラス種類	extends StatelessWidget
	処理概要	下部ナビゲーションバーを制御し、店舗側の主要 5 画面を切り替える親クラス
	入力	なし
	出力	Widget
	他クラスとの関係	AutoTabsScaffold 等を用い、s_home、s_work、s_order_list、s_sales、mypage を子ルートとして管理する

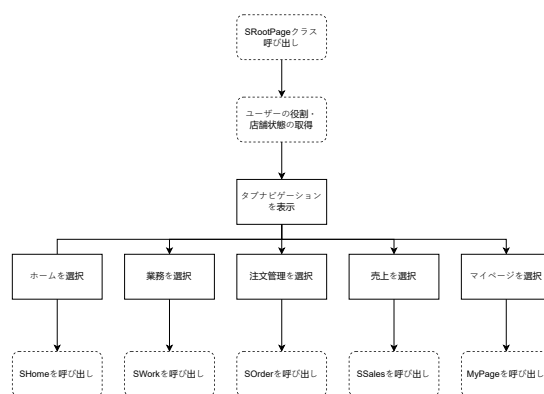
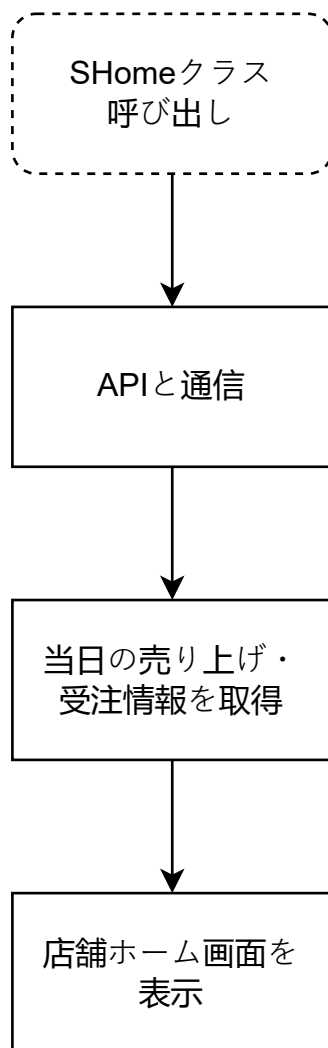


図 20 s_root_page.dart の状態遷移図

モジュール名	s_home.dart	
モジュール概要	店舗ホーム画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SHomePage
	クラス種類	extends StatelessWidget
	処理概要	本日の注文数、売上概要を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	PI から店舗のサマリーデータを取得して表示する



モジュール名	s_work.dart	
モジュール概要	業務画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SWorkPage
	クラス種類	extends StatelessWidget
	処理概要	メニュー情報設定と在庫ステータス設定を管理する
	入力	なし
	出力	Widget
	他クラスとの関係	
	s_inventory_status.dart、s_menu_edit.dart のリンクを持つ	

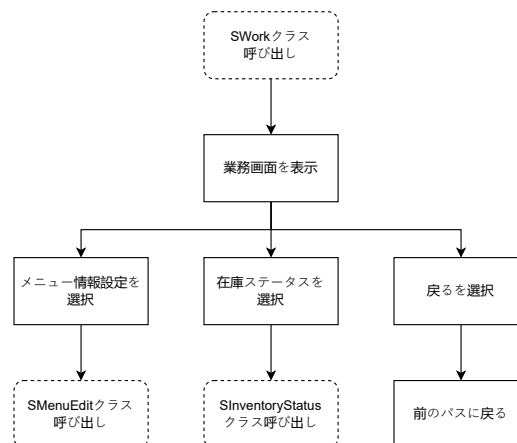
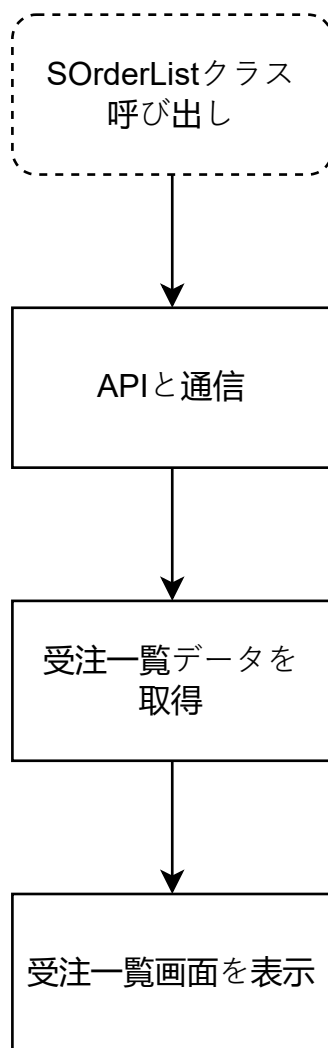
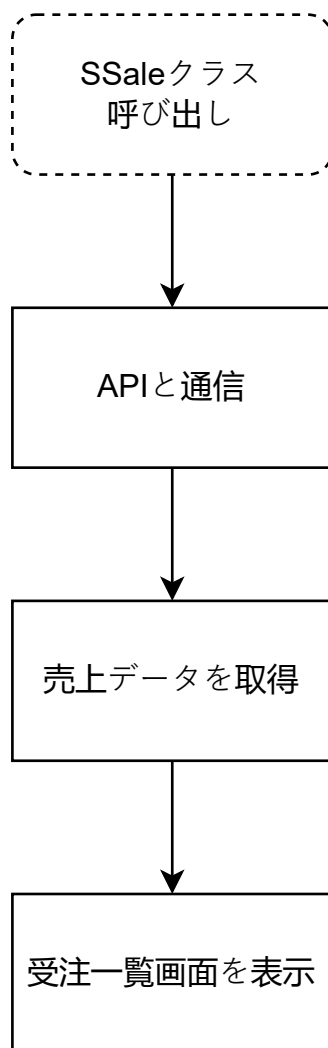


図 22 s_work.dart の状態遷移図

モジュール名	s_order_list.dart	
モジュール概要	受注一覧画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SOrderListPage
	クラス種類	extends StatelessWidget
	処理概要	注文一覧を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	なし	



モジュール名	s_sales.dart	
モジュール概要	売上確認画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SSalesPage
	クラス種類	extends StatelessWidget
	処理概要	日別の売上集計を表示する
	入力	なし
	出力	Widget
	他クラスとの関係	
	なし	



モジュール名	s_menu_edit.dart	
モジュール概要	メニュー情報設定（登録・変更・削除）画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SMenuEditPage
	クラス種類	extends StatelessWidget
	処理概要	メニューの一覧表示および編集（追加・更新・削除）の起点となる画面
	入力	なし
	出力	Widget
	他クラスとの関係	
	s_menu_regist.dart, s_menu_change.dart, s_menu_delete.dart の機能を包含または遷移する	

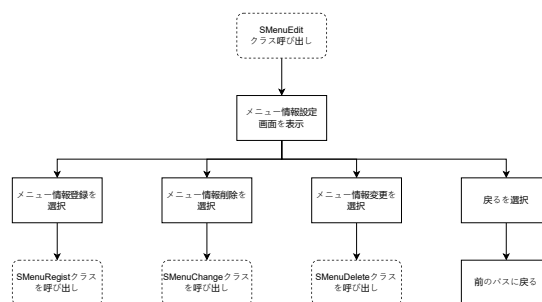
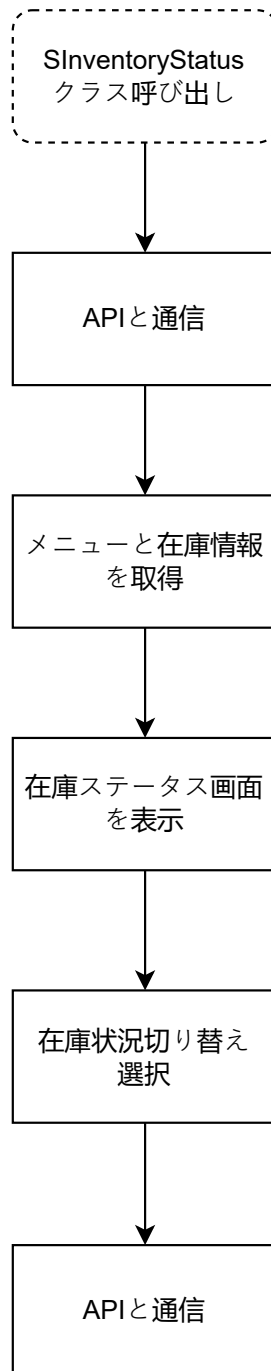


図 25 s_menu_edit.dart の状態遷移図

モジュール名	s_inventory_status.dart	
モジュール概要	店舗の在庫ステータス画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SInventoryStatus
	クラス種類	extends StatelessWidget
	処理概要	メニューごとの「在庫あり/なし」を切り替え、ユーザー画面への反映を制御する。
	入力	なし
	出力	Widget
	他クラスとの関係	
	API と通信し、商品の品切れフラグを更新する。	



モジュール名	s_menu_regist.dart	
モジュール概要	メニュー情報の新規登録画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SMenuRegist
	クラス種類	extends StatefulWidget
	処理概要	新しいメニューの名前、価格、商品説明、 画像などを入力・保存する。
	入力	なし
	出力	Widget
	他クラスとの関係	
	登録完了後、finish_screen.dart を表示し、s_menu_edit.dart に戻る。	

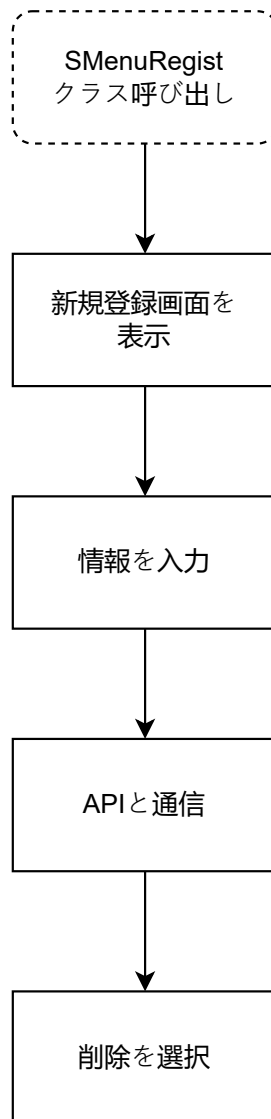


図 27 `s_menu_regist.dart` の状態遷移図

モジュール名	s_menu_change.dart	
モジュール概要	メニュー情報の変更画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SMenuChange
	クラス種類	extends StatefulWidget
	処理概要	既存メニューの情報を取得し、修正内容をデータベースに反映させる。
	入力	なし
	出力	Widget
	他クラスとの関係	
	confirm_inf_change.dart を呼び出して変更の最終確認を行う。	

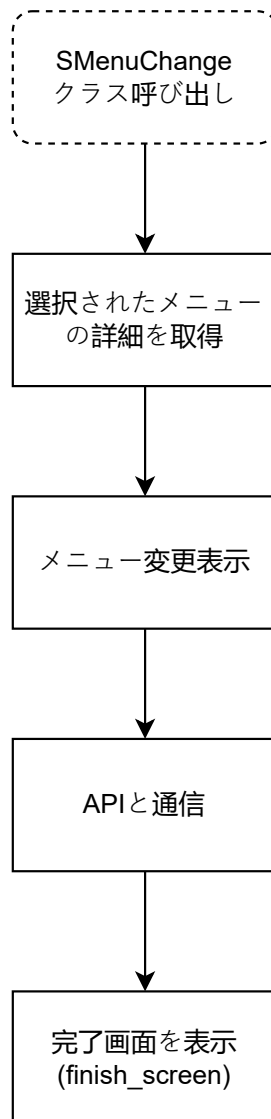


図 28 s_menu_change.dart の状態遷移図

モジュール名	s_menu_delete.dart	
モジュール概要	メニュー情報の削除画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SMenuDelete
	クラス種類	extends StatelessWidget
	処理概要	特定のメニューを削除する際の最終確認画面を表示し、削除処理を実行する。
	入力	なし
	出力	Widget
	他クラスとの関係	
	API 通信を行い、論理削除または物理削除を実行後、一覧へ戻る。	

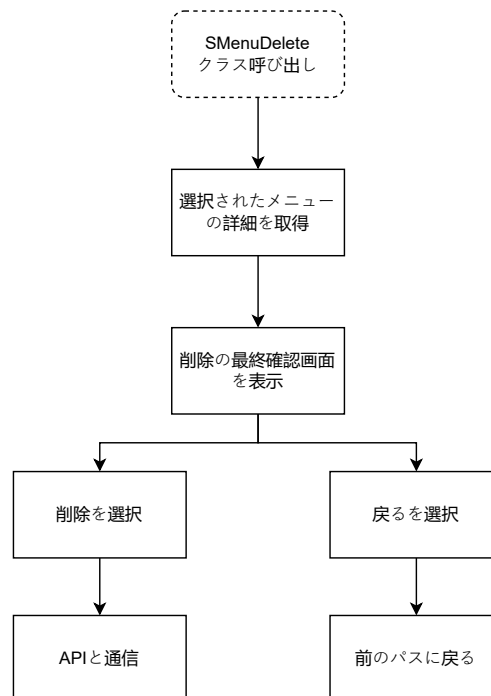


図 29 s_menu_delete.dart の状態遷移図

モジュール名	s_banking_info.dart	
モジュール概要	口座情報管理画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SBankingInfo
	クラス種類	extends StatelessWidget
	処理概要	売上の振込先となる銀行口座情報の確認および、変更画面への遷移を行う。
	入力	なし
	出力	Widget
	他クラスとの関係	
	口座変更時は、共通のバリデーションコンポーネント等を利用する。	

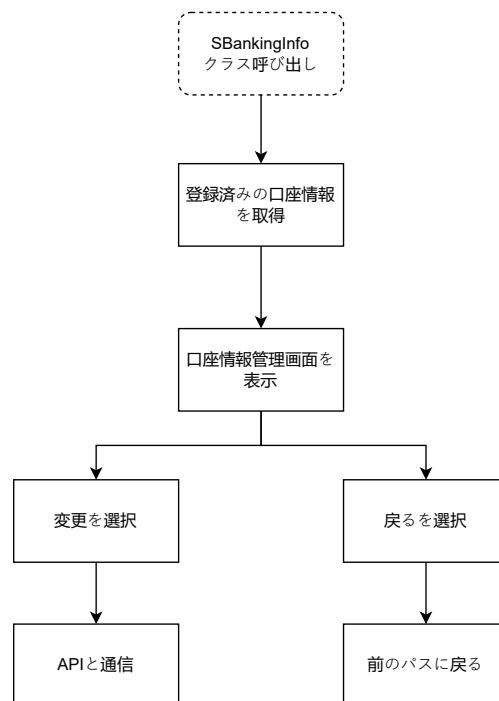


図 30 s_banking_info.dart の状態遷移図

モジュール名	s_delivery_complete.dart	
モジュール概要	受け渡し完了確認画面を表示するモジュール	
責任者	江藤 加奈	
構成要素	クラス名	SDeliveryComplete
	クラス種類	extends StatelessWidget
	処理概要	配達員または注文者への商品受け渡しが完了したことを確定する。
	入力	なし
	出力	Widget
	他クラスとの関係	
	確定後、finish_screen.dart を表示する。	

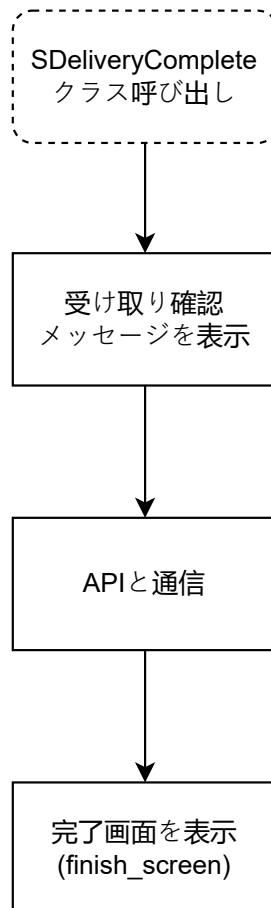


図 31 `s_delivery_complete.dart` の状態遷移図

5.3.4 配達側

5.4 component

モジュール名	overscreen_controller.dart	
モジュール概要	オーバーレイの表示、非表示の操作を行うコンポーネント	
責任者	梶本和希	
構成要素	クラス名	OverScreenController
	クラス種類	immutable
	処理概要	オーバーレイの内容を返す
	入力	なし
	出力	Widget
	他クラスとの関係	
	CloseLoadingScreen によってオーバーレイを閉じる UpdateLoadingScreen によってオーバーレイを表示	

モジュール名	finishscreen.dart	
モジュール概要	完了画面を生成するコンポーネント	
責任者	梶本和希	
構成要素	クラス名	FinishScreen
	クラス種類	extendsStatelessWidget
	処理概要	アイコン、テキストを引数で与えることによって、任意の完了画面を生成するコンポーネント
	入力	StringappbarText, IconDataappIcon,String-text,StringbuttonTextbool isAppBar
	出力	Widget
	他クラスとの関係	
	titleappbar.dart の TitleAppBar クラスを組み込む normalbottomappbar.dart の NormalBottomAppBar クラスを組み込む buttonset.dart の ButtonSet クラスを組み込む LoginRoute の最初の画面へと戻る	

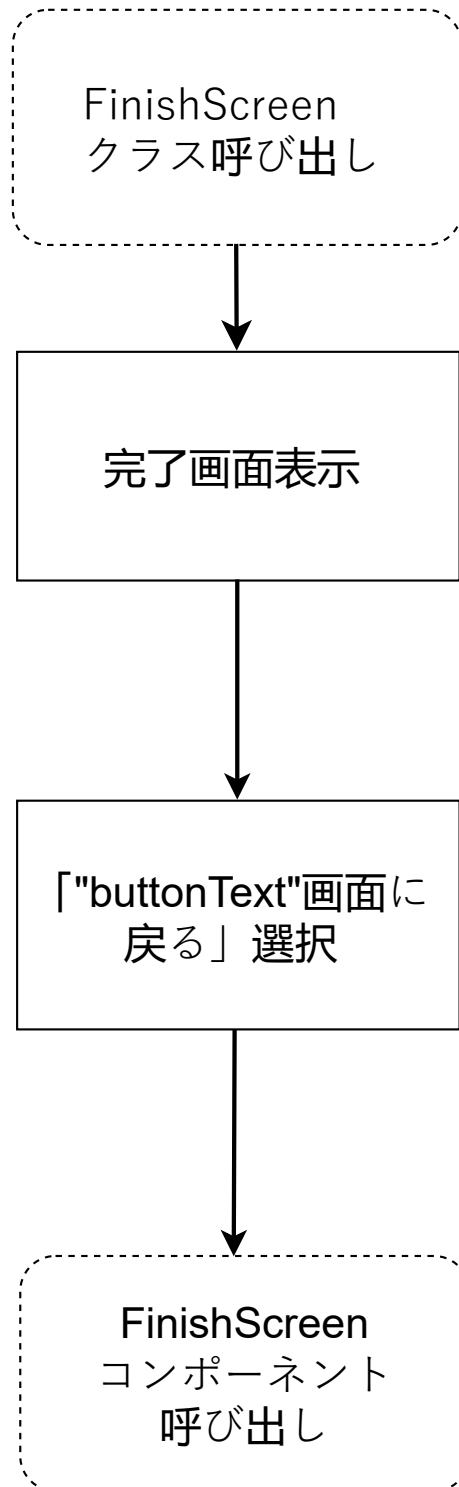


図 32 FinishScreen の状態遷移図

モジュール名	botton_set.dart	
モジュール概要	任意のテキストが書かれたボタン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	ButtonSet
	クラス種類	extendsStatelessWidget
	処理概要	body の呼び出された部分にボタンを設置するコンポーネント
	入力	String buttonName
	出力	Widget
	他クラスとの関係	
	change_user_role.dart の ChangeUserRole.dart クラスを組み込む	

モジュール名	title_appbar.dart	
モジュール概要	AppBar の下にページタイトルを表示する AppBar を作成するコンポーネント	
責任者	梶本和希	
構成要素	クラス名	TitleAppBar
	クラス種類	extendsStatelessWidget
	処理概要	AppBar の bottom にてもう一度 AppBar を作成し、そこにページのタイトルを表示させる AppBar を生成するコンポーネント
	入力	String appBarText, bool isReturn
	出力	Widget
	他クラスとの関係	
	pop を行う	

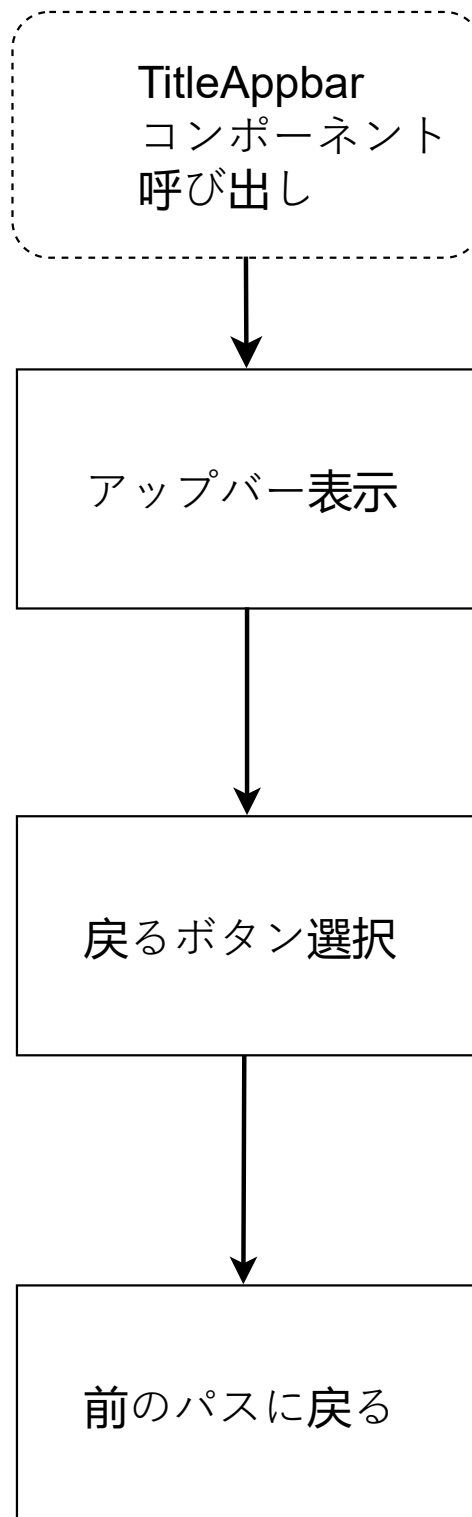


図 33 TitleAppBar の状態遷移図

モジュール名	normal_bottom_appbar.dart	
モジュール概要	BottomNavigationBar がない画面で用いる BottomAppBar を作成するコンポーネント	
責任者	梶本和希	
構成要素	クラス名	NormalBottomAppBar
	クラス種類	extends StatelessWidget
	処理概要	色がついただけの BottomAppBar の Widget を返すコンポーネント
	入力	なし
	出力	Widget
	他クラスとの関係	
	なし	

モジュール名	button_set.dart	
モジュール概要	任意のテキストが書かれたボタンを生成するモジュール	
責任者	金澤優	
構成要素	クラス名	ButtonSet
	クラス種類	extends StatelessWidget
	処理概要	body の呼び出された部分にボタンを設置するコンポーネント
	入力	String buttonName
	出力	Widget
	他クラスとの関係	
	change_general_corporation.dart の ChangeGeneralCorporation クラスを組み込む	

6 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。

6.1 認証 (Auth)

ログイン

POST /auth/login

Request	email (str), password (str)
Response	access_token (str), token_type (str), user_role (str)

新規登録

POST /auth/register

Request	email (str), password (str), user_type (str), profile_data (dict)
Response	user_id (int), message (str)

6.2 商品 (Products)

店舗別商品一覧取得

GET /stores/{store_id}/products

Request	store_id (int)
Response	list[product_object]

商品登録 (店舗のみ)

POST /products

Request	token (str), name (str), price (int), description (str), image (str), stock_status (bool)
Response	product_id (int), message (str)

6.3 注文 (Orders)

注文作成

POST /orders

Request	token (str), store_id (int), items[list{product_id (int), quantity (int)}], total_price (int)
Response	order_id (int), message (str)

注文ステータス更新

PUT /orders/{order_id}/status

6.4 配達 (Delivery)

配達可能案件一覧取得

GET /delivery/jobs

案件受注

POST /delivery/jobs/{order_id}/accept

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

Request	token (str), current_location (str)
Response	list[order_object]

7 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。

7.1 テーブル一覧

表 9 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

7.2 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

Request	token (str), order_id (int)
Response	message (str)

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