外部設計書

香美市特化型配達サービス

Stellar Delivery

第 1 版

2025 年 11 月 12 日

StellarWorks

目次

1 はじめに …… 3
2 システム概要 …… 3
3 機能設計 …… 3
3.1 機能概要図 …… 4
3.1.1 ログイン関連機能 …… 4
3.2 共通機能 …… 4
3.2.1 新規会員登録機能 …… 4
3.2.2 ログイン …… 5
3.2.3 ログアウト …… 6
3.2.4 退会 …… 7
3.2.5 パスワード再発行 …… 8
3.2.6 会員情報確認・変更 …… 9
3.2.7 会員情報確認・変更 …… 10
3.3 依頼側 …… 11
3.4 配達側 …… 12
3.5 店舗側 …… 12
3.6 管理側 …… 12
4 UI 設計 …… 12
4.1 依頼側 …… 12
4.2 配達側 …… 12
4.3 店舗側 …… 12
4.4 管理者側 …… 12
5 運用保守設計 …… 12
5.1 運用 …… 12
5.2 保守 …… 13
5.3 セキュリティ対策 …… 13
5.3.1 ユーザー認証 …… 13
5.3.2 パスワードのハッシュ化 …… 13
5.3.3 SQL インジェクション対策 …… 13
5.3.4 SSL 通信 …… 13
5.4 障害対策 …… 13
5.4.1 ハードウェア障害 …… 13
5.4.2 ソフトウェア障害 …… 13
5.4.3 アプリケーション開発及びテストの障害対策 …… 14

6 データベース設計 14
7 ネットワーク設計 17

1　はじめに

本書では, 弊社がシステム提案書で提案した地域密着型アルバイト求人アプリ「Stellar Delivery」の機能詳細について示す．まず，本システムの概要について示す．なお，主張の根拠はシステム提案書を参照されたい．次に本システムの画像遷移図とユーザーインターフェースを示す． その後, 運用・保守について示し, 最後に本システムを構築するデータベース設計とネットワーク設計について示す.

2　システム概要

本アプリケーション「Stellar Delivery」は、香美市特化型の配達サービスである。各々のユーザーは最初にそれぞれの利用目的（店舗側または、依頼側、配達側）に沿った利用規約に同意し、ユーザー登録を行うことで本アプリケーションを利用することができる。

図 1 は、商品の依頼から配達までのサービスの流れを表している。主な役割は、「店舗側」、「依頼側」、「配達側」、「管理側」、「Stellar Delivery」、「銀行」そして「オンライン決済システム」に分かれている。オンライン決済システムは、ユーザーがアプリ上でクレジットカード情報または kamica などの情報を登録し、決済を行う。

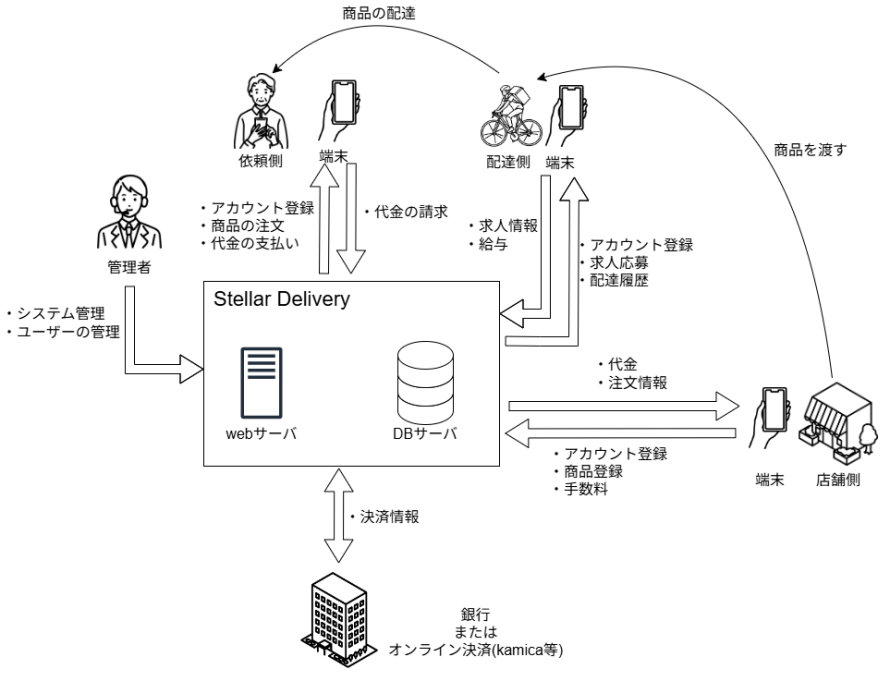

図 1　システム概要図

3　機能設計

本アプリの依頼側、配達側、店舗側そして管理側の各機能設計について記述する。

3.1　機能概要図

以下に示す図は本アプリの機能概要である。

3.1.1　ログイン関連機能

図 2 は、ログイン関連機能の概要図である。

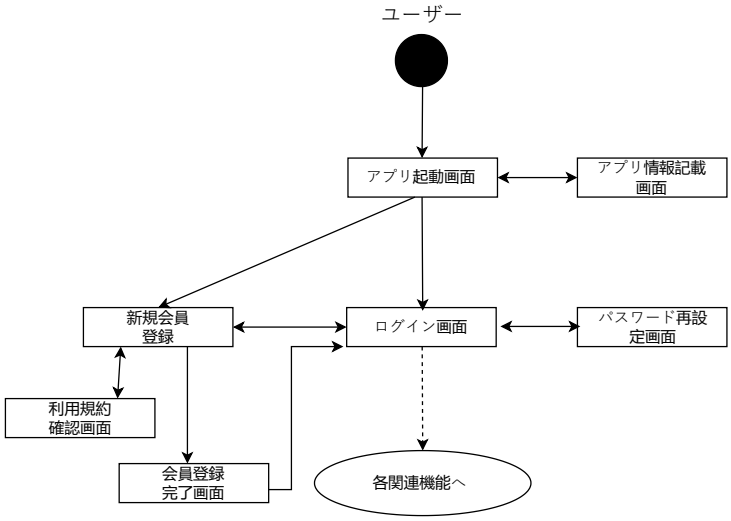

図 2　ログイン関連機能

3.2　共通機能

本アプリの依頼側、配達側、店舗側そして管理側の共通の機能設計を以下に記述する。

3.2.1　新規会員登録機能

依頼側と配達側、店舗側が利用規約に同意し、情報を入力し、その情報に不備がなく、送信されたメールで本人確認を行うことで、新規会員登録ができる機能である。また、入力情報として、配達側と依頼側は名前、性別、生年月日、住所、メールアドレス、電話番号、パスワード、パスワード（確認）を、店舗側は店舗所在地、飲食店営業許可書のファイルおよび写真、メールアドレスまたは電話番号、パスワード、パスワード（確認）を入力する。

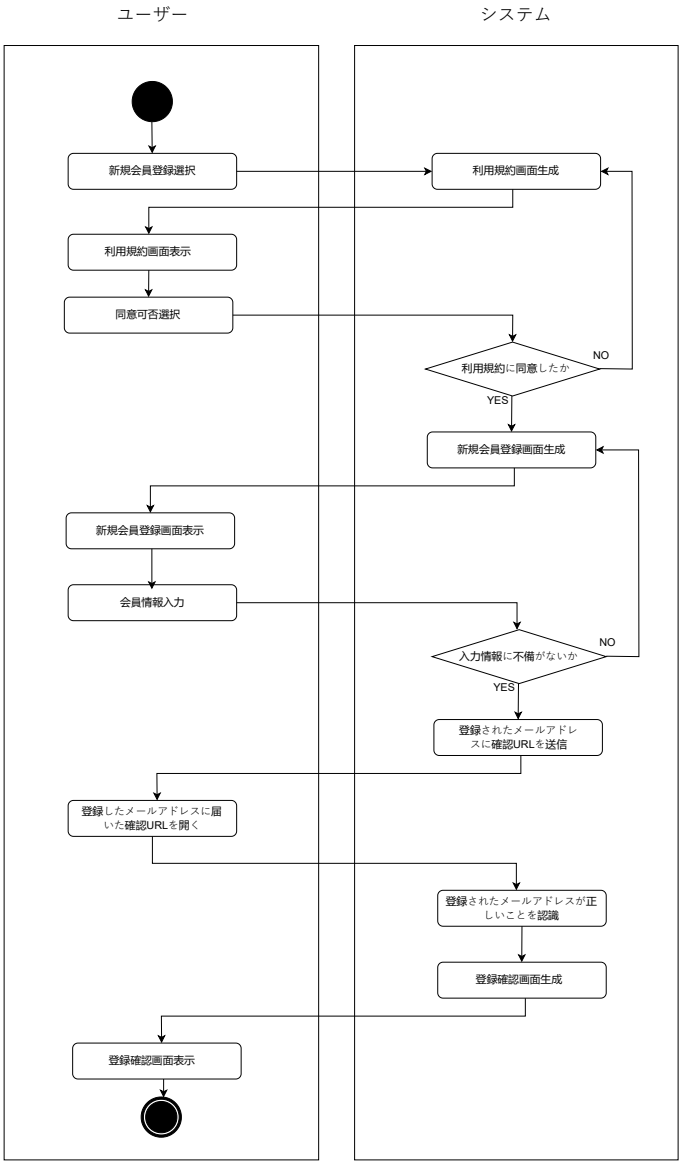

図 3　新規会員登録機能

3.2.2　ログイン

アプリ起動時に会員登録済みの利用者（依頼側、配達側、店舗側そして管理側）が、自身のログイン情報を入力することで、アプリのサービスを利用できるようにする機能である。依頼側、配達側、店舗側はメールアドレスとパスワードを、管理側は管理者 ID とパスワードを入力して、ログインする。

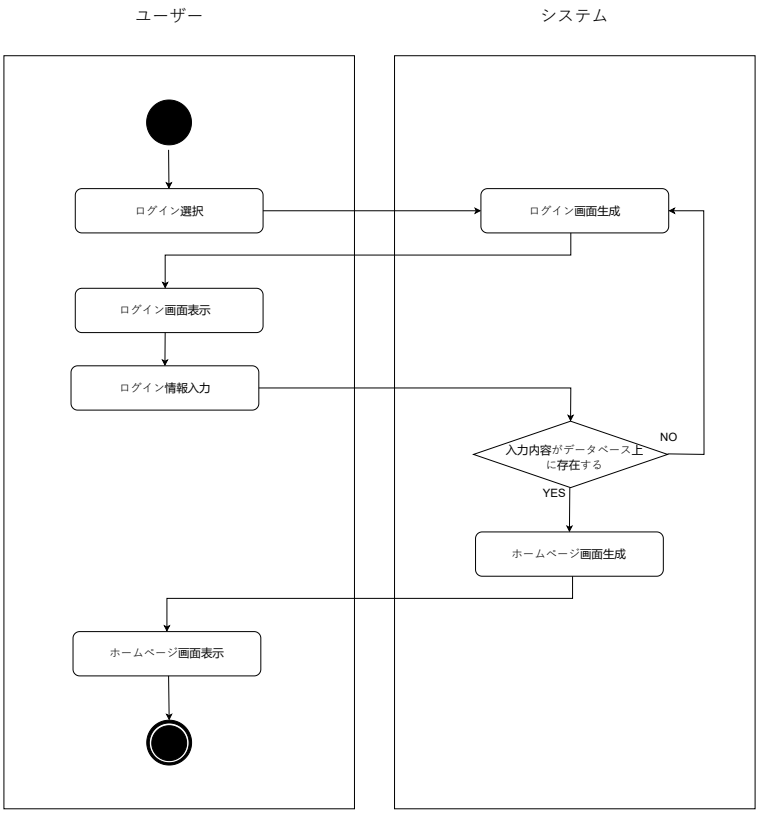

図 4　ログイン機能

3.2.3　ログアウト

ログアウトを選択することで、ログイン前の状態に戻す機能で、ログイン画面に遷移する。

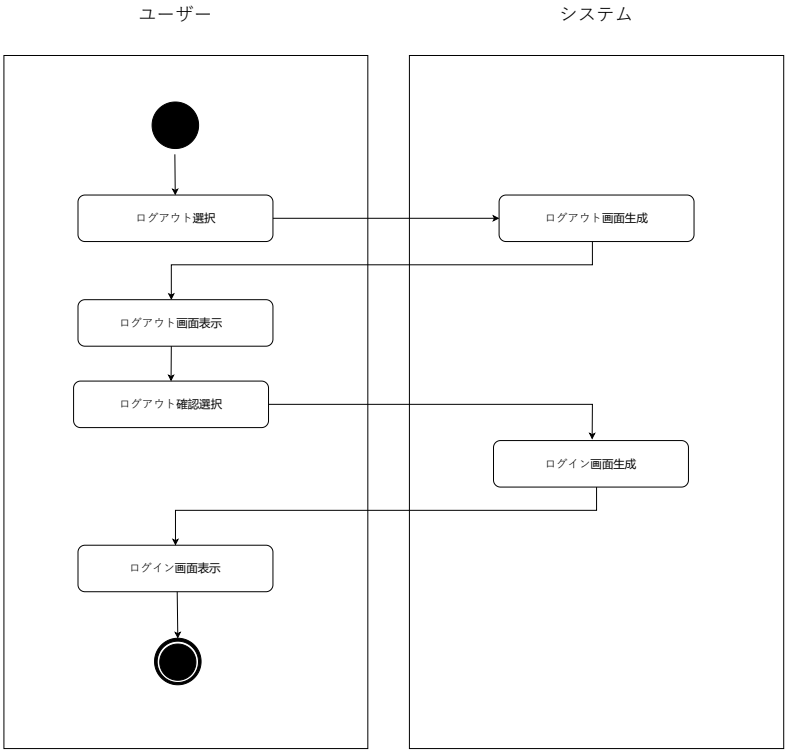

図 5　ログアウト機能

3.2.4　退会

　会員が任意のタイミングでアプリ会員から退会する機能。会員情報をデータベースから完全に削除する。

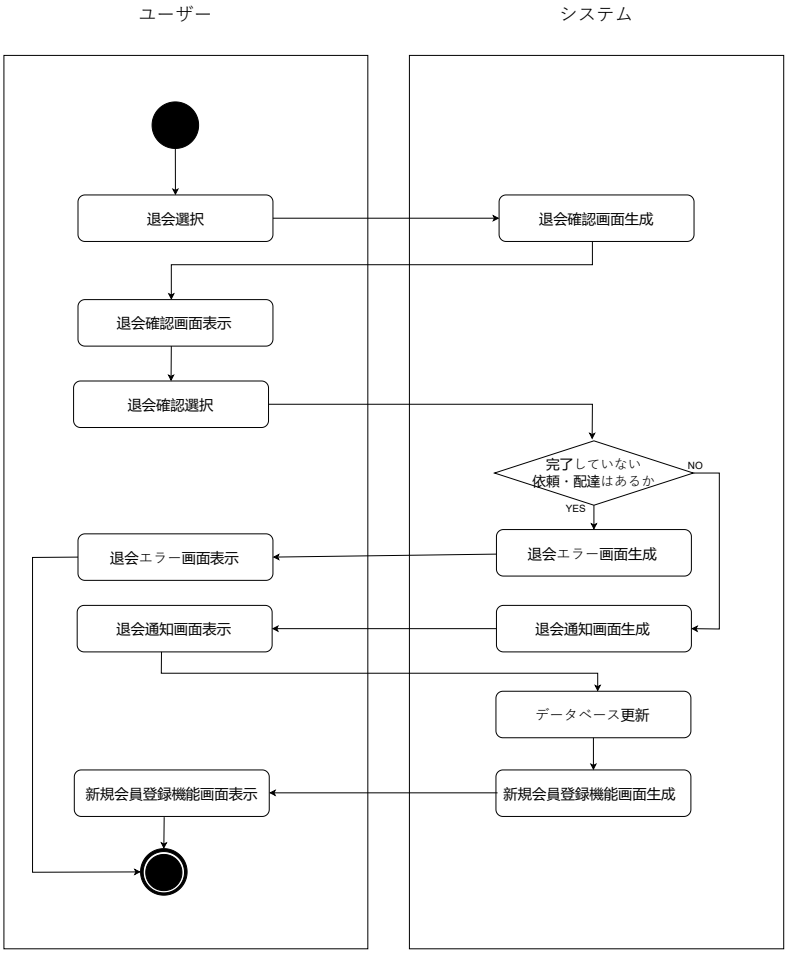

図 6　退会機能

3.2.5　パスワード再発行

ユーザーはパスワードを忘れた場合、メールアドレスをフォームに入力することで再発行を行う。間違っている場合はエラーメッセージを出力し、合っている場合はパスワードを生成してデータベースを更新し、メールアドレス宛にそのパスワードを送信し、通知画面を表示する。

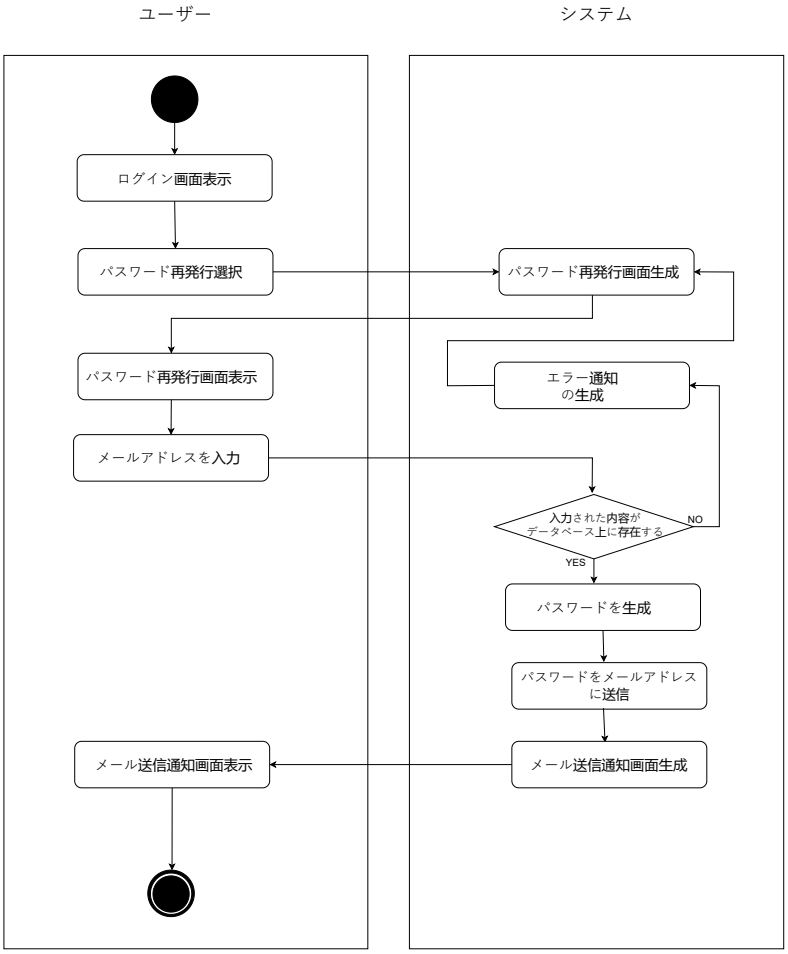

図 7　パスワード再発行機能

3.2.6　会員情報確認・変更

会員がアプリ内のホームページ画面から自身の会員情報を確認・変更するための機能である。

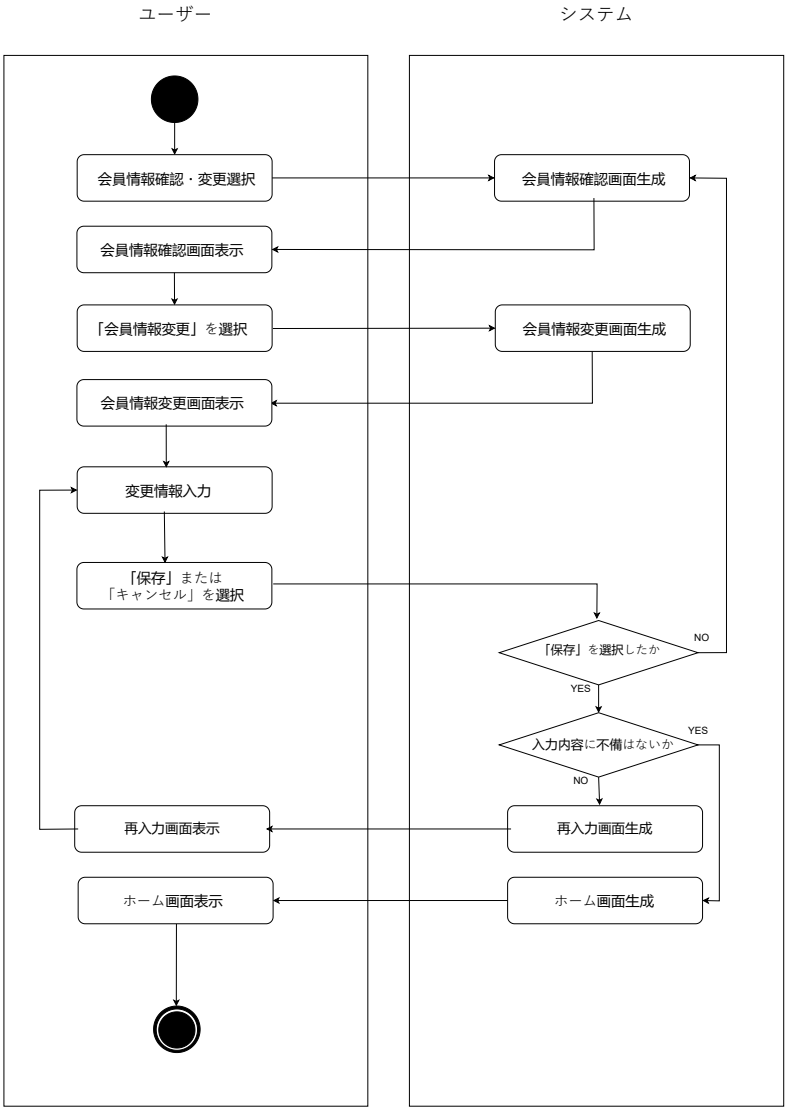

図 8　会員情報確認・変更

3.2.7　会員情報確認・変更

会員がアプリの操作方法などの問い合わせを管理者に対して行うための機能である。問い合わせへの回答は会員登録したメールアドレスに送信される。

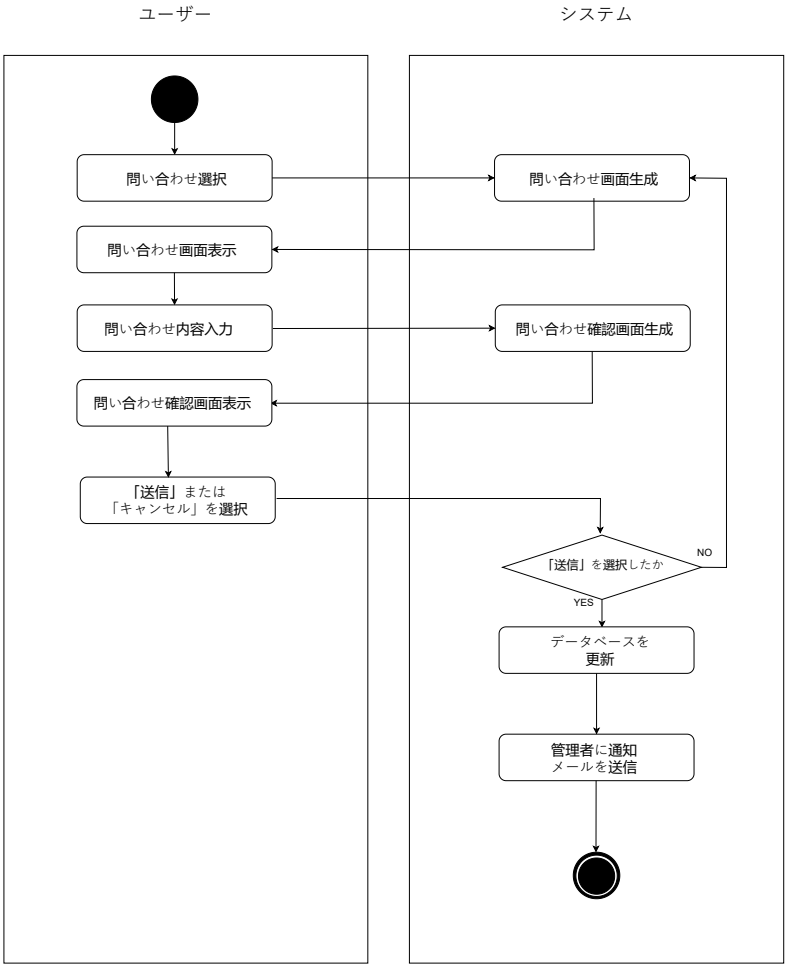
ユーザー
システム
問い合わせ選択
問い合わせ画面生成
問い合わせ画面表示
問い合わせ内容入力
問い合わせ確認画面生成
問い合わせ確認画面表示
「送信」または
「キャンセル」を選択
「送信」を選択したか
NO
YES
データベースを
更新
管理者に通知
メールを送信

図 9　問い合わせ機能

3.3　依頼側

3.4　配達側

3.5　店舗側

3.6　管理側

4　UI 設計

本アプリの UI 設計を以下に示す.

4.1　依頼側

4.2　配達側

4.3　店舗側

4.4　管理者側

5　運用保守設計

本アプリケーションのバックエンド側は AmazonWebServices で構築する． したがって，24 時間 365 日という時間はオンライン/バッチを含みシステムが稼働している時間として定義する．本アプリに関しての運用・保守について以下に記述する.

5.1　運用

運用事項について以下の項目を行う。

- システムが正常に稼働しているかの監視
- データのバックアップ処理
- 障害対応
- セキュリティ対策
- 定期メンテナンス
- ユーザーからの問い合わせ対応

個人情報へのアクセス権は弊社社員 6 名に限定し，アクセスログを取得することで不正アクセスを

防止する．ユーザーからの問い合わせに関しては，専用の問い合わせフォームを設置し，対応状況を管理する．

5.2 保守

保守事項については以下の項目とする。

- ソフトウェアのバージョンアップ
- バグ修正
- 機能追加・改善

上記を定期的に実施し，システムの安定性とユーザー満足度を向上させる．システムの根幹にかかわるバグが発見された場合は、速やかにシステムを停止し、原因の調査と修正を行う。

5.3 セキュリティ対策

5.3.1 ユーザー認証

ユーザー登録の際、メールで本人確認を行う。また、パスワードは強力なものを要求する。

5.3.2 パスワードのハッシュ化

パスワードは、安全なハッシュ関数を用いてハッシュ化し、データベースに保存する。この操作を行うことで、万が一データベースが漏洩した場合でも、パスワードの平文が流出するリスクを軽減する。また、個人情報にアクセスできる社員もパスワードを閲覧することができないようになる。

5.3.3 SQL インジェクション対策

SQL インジェクション攻撃を防ぐために、プリペアドステートメントを使用し、ユーザーからの入力を適切にエスケープする。

5.3.4 SSL 通信

SSL 通信を採用し、通信内容の盗聴や改ざんを防止する。

5.4 障害対策

5.4.1 ハードウェア障害

AWS 上にバックエンドを構築することで、ハードウェア障害に対する冗長化を実現する。AWS の障害以外では、ハードウェア障害は発生しないものとする。

5.4.2 ソフトウェア障害

ソフトウェア障害に対しては、定期的なバックアップと監視システムを導入することで、障害発生時に迅速に対応できる体制を整える。

5.4.3 アプリケーション開発及びテストの障害対策

開発者各々の環境に依存しないように、Docker を用いて開発を行う。

6 データベース設計

```
%% Mermaid ER diagram for Stellar Delivery
%% Save as .mmd and open with Mermaid live editor or VSCode Mermaid Preview

erDiagram
    USERS {
        INT user_id PK "ユーザー ID"
        VARCHAR name "氏名"
        VARCHAR email "メールアドレス (unique)"
        VARCHAR password_hash "パスワードハッシュ"
        VARCHAR phone "電話番号"
        DATE birthday "生年月日"
        ENUM role "customer/delivery/store/admin"
        DATETIME created_at "登録日時"
        DATETIME updated_at "更新日時"
    }

    CUSTOMERS {
        INT customer_id PK FK "FK -> users.user_id"
        VARCHAR address "配送先住所"
        ENUM payment_method "credit/bank/kamica/cash"
        DECIMAL balance "残高"
    }

    STORES {
        INT store_id PK FK "FK -> users.user_id"
        VARCHAR store_name "店舗名"
        VARCHAR address "店舗住所"
        VARCHAR bank_account "振込口座"
```

```
    DECIMAL fee_rate "手数料率 (%)"
}

DELIVERY_WORKERS {
    INT delivery_id PK FK "FK -> users.user_id"
    ENUM vehicle_type "bicycle/motorbike/car"
    ENUM employment_status "part-time/freelance"
    INT total_deliveries "累計配達回数"
    FLOAT rating "平均評価"
}

ADMINS {
    INT admin_id PK FK "FK -> users.user_id"
    TEXT permissions "権限 (JSON 等)"
}

PRODUCTS {
    INT product_id PK "商品 ID"
    INT store_id FK "FK -> stores.store_id"
    VARCHAR name "商品名"
    TEXT description "商品説明"
    DECIMAL price "価格"
    VARCHAR image_url "画像 URL"
    INT stock "在庫数"
}

ORDERS {
    INT order_id PK "注文 ID"
    INT customer_id FK "FK -> customers.customer_id"
    INT store_id FK "FK -> stores.store_id"
    INT delivery_id FK "FK -> delivery_workers.delivery_id (NULL 可)"
    DECIMAL total_amount "合計金額"
    ENUM status "ordered/preparing/delivering/completed/canceled"
    DATETIME order_time "注文日時"
    DATETIME delivery_time "配達完了日時"
}
```

```
ORDER_ITEMS {
    INT order_id FK "FK -> orders.order_id"
    INT product_id FK "FK -> products.product_id"
    INT quantity "数量"
    DECIMAL subtotal "小計"
    PK order_product_pk "(order_id, product_id)"
}

PAYMENTS {
    INT payment_id PK "決済 ID"
    INT order_id FK "FK -> orders.order_id"
    ENUM payment_method "credit/bank/kamica"
    DECIMAL amount "支払金額"
    ENUM status "pending/completed/failed"
    VARCHAR transaction_id "外部決済 ID"
    DATETIME created_at "実行日時"
}

DELIVERY_HISTORY {
    INT history_id PK "履歴 ID"
    INT delivery_id FK "FK -> delivery_workers.delivery_id"
    INT order_id FK "FK -> orders.order_id"
    ENUM status "accepted/picked_up/delivered/canceled"
    DATETIME timestamp "状態変更時刻"
}

JOBS {
    INT job_id PK "求人 ID"
    INT store_id FK "FK -> stores.store_id"
    VARCHAR title "タイトル"
    TEXT description "募集内容"
    DECIMAL wage "報酬額"
    DATETIME created_at "登録日時"
    DATETIME closed_at "募集締切日時"
}

JOB_APPLICATIONS {
```

```
        INT job_id FK "FK -> jobs.job_id"
        INT delivery_id FK "FK -> delivery_workers.delivery_id"
        DATETIME applied_at "応募日時"
        ENUM status "applied/accepted/rejected"
        PK job_app_pk "(job_id, delivery_id)"
    }

    %% Relationships (cardinality expressive)
    USERS ||--|| CUSTOMERS : "is a"
    USERS ||--|| STORES : "is a"
    USERS ||--|| DELIVERY_WORKERS : "is a"
    USERS ||--|| ADMINS : "is a"

    STORES ||--o{ PRODUCTS : "sells"
    CUSTOMERS ||--o{ ORDERS : "places"
    STORES ||--o{ ORDERS : "receives"
    DELIVERY_WORKERS ||--o{ ORDERS : "delivers"

    ORDERS ||--o{ ORDER_ITEMS : "contains"
    PRODUCTS ||--o{ ORDER_ITEMS : "included in"

    ORDERS ||--o{ PAYMENTS : "has"

    DELIVERY_WORKERS ||--o{ DELIVERY_HISTORY : "has history"
    ORDERS ||--o{ DELIVERY_HISTORY : "has history"

    STORES ||--o{ JOBS : "posts"
    JOBS ||--o{ JOB_APPLICATIONS : "receives"
    DELIVERY_WORKERS ||--o{ JOB_APPLICATIONS : "applies"
```

7 ネットワーク設計

test