Frontend

Frontend Documentation: Mobile Receipt Capture App

Overview

This document outlines the frontend design and features for the mobile receipt capture app, built specifically for iOS devices (iPhone 15 Pro Max) using SwiftUI. The app focuses on a simple and streamlined workflow for capturing receipts and syncing them with a designated "Receipts" folder in iCloud.

Features

1. Landing Page for Confirmation

- **Trigger**: Opens upon activation by the **Action Button** (lower button on the right side of the phone).
- **Purpose**:
  - Prevent accidental actions (e.g., "butt calls").
  - Provide a clear option to confirm that you want to enter a receipt.
- **Design**:
  - A single, prominent **"Enter Receipt"** button to confirm intent and proceed to the camera for receipt capture.
  - No additional features like receipt history or settings are included. These are reserved for the desktop app.

2. Camera Workflow

- **Activation**: Initiated by tapping the "Enter Receipt" button on the landing page.
- **Features**:
  - **Preview Mode**:
    - After capturing a photo, a preview is displayed.
    - Options:
      - **"Use the Photo"**: Saves the photo to the designated "Receipts" folder in iCloud.
      - **"Retake Photo"**: Allows retaking the photo if it's unclear or junk.
  - Ensures only high-quality photos are saved.

3. Exit Workflow with Double-Tap

- **Trigger**: A **double-tap** on the **Action Button**.
- **Actions**:
  - Signals the app that receipt capturing is complete.
  - Automatically closes the camera and returns to the landing page.
  - Finalizes and syncs the "Receipts" folder with iCloud.

4. Post-Capture Handling

- **Storage**:
  - Captured receipt photos are saved in the "Receipts" folder in iCloud.
  - The folder will be accessed later by the desktop app for processing and data extraction.
- **Mobile App Simplicity**:
  - No additional processing or categorization occurs on the mobile app.

Backend

Backend Documentation: Desktop Receipt Processing App

Overview

This document outlines the backend design and features for the desktop app responsible for processing receipt images, extracting data, and managing a local SQLite database.

Features

1. Receipt Processing

- Displays the receipt image for review.
- Uses OCR (Tesseract) for data extraction.
- Automatically validates data and saves it into SQLite.

2. Folder Management

- Syncs with iCloud folder for new receipts.
- Archives processed receipts and sets up a new folder for future uploads.

3. Data Export

- Allows data export to CSV with filtering options for date ranges, categories, etc.

4. Database Management

- SQLite-based database for local storage of receipt and item details.

Third_party_libraries

Third-Party Libraries Documentation

Libraries and Tools

1. Tesseract OCR

- **Purpose**: Extract text from receipt images.

- **Why Chosen**: Open-source and reliable.

2. SQLite

- **Purpose**: Local database storage for receipts.

3. PyQt5

- **Purpose**: User interface for the desktop app.

Testing_plan

Testing Plan Documentation

Overview

This document outlines the testing strategies for ensuring the functionality, performance, and reliability of the desktop receipt processing app.

---

1. Unit Testing

- Test individual functions like `process_receipts` and `extract_data`.

2. Integration Testing

- Test seamless interaction between modules (e.g., UI, database, and OCR).

3. UI/UX Testing

- Validate receipt image display and data entry flows.

4. Error Handling Testing

- Simulate OCR and database errors and verify user notifications.

5. Performance Testing

- Ensure stability under large data loads.