**Medical Billing App**

**Problem:**

Managing the operations of a pharmacy can be complex and time-consuming. Manual billing leads to errors, takes time, and can lead to discrepancies in sales records. Slow billing can also result in customer dissatisfaction and revenue loss. Pharmacy staff struggles to know exactly when to restock or remove items, resulting in inventory inefficiencies. The pharmacy owner is unable to keep records of daily and monthly sales. This app will help manage the pharmacy and reduce human error.

**Objectives:**

The purpose of building this project is to provide a simple billing process which will help generate quick and accurate bills. It will help users maintain stock for medicine and other pharmacy products and generate daily sales reports. It will also help maintain financial statements for the pharmacy and provide alerts when products are low in stock.

**Goals:**

1. Develop a quality desktop software application that is efficient and accurate.
2. Provide a user-friendly interface.
3. Ensure better record management.
4. Reduce the time required to generate bills.
5. Improve stock management for medicines and other pharmacy products.
6. Provide stock updates and notifications for low stock.
7. Ensure high security and optimization in both functional and non-functional aspects.
8. Deliver enhanced performance.

**Requirements:**

1. **Login Page**:
   * User authentication via specific username and password.
2. Forget Password Page:

* Forget password using a specific 6-digit code received on company email.

1. **Signup Page**:
   * Authentication of new users via username, password, and a specific passkey.
2. **Billing Management Page**:
   * Allows the user to select the desired quantity of required products (if available) and generate a bill according to respective prices.
3. User Management Page:

Allows the user to manage previous users

1. **Stock Management Page**:
   * Allows the user to view available stock and add new products to the stock.
   * Supports categorization of products by type, brand, or medical category for easier management.
2. **Low Stock Notifications**:
   * Notify users when a product falls below the defined stock threshold.
   * Notifications can be shown in-app (as banners or popups) and via external means like email or SMS.
3. **Sales Report**:
   * Displays the sales report for the overall day.
   * Allows users to generate weekly and monthly sales reports for comprehensive insights.
4. **Advanced Search Functionality**:
   * Includes a search bar with filters to quickly locate products by name, category, or stock level.
5. **Export and Import Features**:
   * Enables exporting sales reports or inventory lists to formats like CSV or PDF.
   * Supports importing stock details from external files for easier data management.

**Non-Functional Requirements:**

1. **Performance**:
   * Quick load time (under 2 seconds).
   * Fast operations (under 3 seconds for billing and stock management).
2. **Scalability**:
   * Supports growing users
3. **Security**:
   * Secure authentication (password hashing, 2FA if needed).
   * Encrypt sensitive data (in transit and at rest).
   * Role-based access control for different user types.
4. **Availability**:
   * 99.9% uptime, with backups and redundancy in place.
5. **Reliability**:
   * Bug-free with logging for quick issue resolution.
   * Handles edge cases like out-of-stock or incorrect input.
6. **Usability**:
   * Intuitive UI, accessible design, and helpful tooltips.
7. **Maintainability**:
   * Modular, easy-to-update code with clear documentation.
8. **Compliance**:
   * Adherence to data privacy laws (GDPR, HIPAA).
   * Audit trails for financial and stock management actions.

**Stakeholders:**

1. Owner of the Pharmacy
2. Cashier of the Pharmacy
3. Stock Manager

**Development Tools & Technologies:**

1. **Frontend**: Flutter (for desktop applications)
2. **Backend**: Express.js/Node.js
3. **Database**: SQLite or Firebase
4. **Deployment**: Local setup for desktops, scalable to AWS/Azure/Google Cloud if needed.
5. **Notifications**:
   * In-app: Flutter notifications (e.g., awesome\_notifications).
   * External: Email (via SMTP services like SendGrid or Nodemailer) and SMS (via APIs like Twilio).

**Prototypes Needed:**

1. **Login Page**
2. **Signup Page**
3. **Forget Password page**
4. **Owner Dashboard Page**
5. **User Management page**
6. **Cashier Dashboard Page**
7. **Billing Management Page** (for all roles)
8. **Stock Management Page** (for Owner and Manager)
9. **Low Stock Notifications Page** (for Stock Manager and Owner)
10. **Sales Report Page (Owner)**

**Users:**

1. **Owner**: The pharmacy owner who has access to all functionalities in the app. They are responsible for managing users, stock, sales reports, and settings.
2. **Cashier**: A user who handles billing and sales-related activities. They can view stock, generate sales reports, and perform billing, but they do not have access to advanced features like stock management or user management.
3. **Stock Manager** (if added in the future): A user who is specifically responsible for managing stock (viewing and updating products, handling low stock alerts). They may have permissions that are different from the **Owner** or **Cashier** but would fall under similar role-based categories.

**Use Cases:**

**1. User Authentication (Login/Signup/Forgot Password)**

* **Actors**: Owner, Cashier and Stock Manager
* **Preconditions**: User has registered an account.
* **Basic Flow**:
  1. User navigates to the login page.
  2. User enters valid credentials (username and password).
  3. If the login is successful, the user is redirected to the dashboard based on their role.
  4. If the user forgets their password, they can use the "Forgot Password" page to reset it.
  5. If the user is new, they can sign up with their credentials and passkey.
* **Exceptions**: Invalid credentials, unregistered email.

**2. Dashboard Access**

* **Actors**: Owner, Cashier
* **Preconditions**: User is logged in.
* **Basic Flow**:
  1. Owner and Cashier are redirected to different dashboards based on their role.
  2. The **Owner** sees all available features, including stock management, sales reports, and user management.
  3. The **Cashier** only sees billing and sales report pages.
* **Exceptions**: None, as it’s role-based.

**3. Billing Management**

* **Actors**: Owner, Cashier
* **Preconditions**: User is logged in and has permission to generate bills.
* **Basic Flow**:
  1. User selects products to purchase from the available stock.
  2. User specifies quantities and generates a bill.
  3. User confirms the payment and generates a receipt.
* **Exceptions**: Insufficient stock, invalid quantities.

**4. Stock Management**

* **Actors**: Owner
* **Preconditions**: User is logged in and is authorized to manage stock.
* **Basic Flow**:
  1. Owner navigates to the stock management page.
  2. Owner can view current stock, add new products, or edit existing stock information.
  3. Owner can categorize products (by type, brand, medical category).
  4. Owner can see low stock levels and reorder items.
* **Exceptions**: Invalid product information, exceeding stock limits.

**5. Low Stock Notifications**

* **Actors**: Owner, Cashier
* **Preconditions**: Stock levels are set with thresholds.
* **Basic Flow**:
  1. When stock reaches a predefined low level, the system sends a notification to the user (via in-app pop-up, email, or SMS).
  2. The user receives alerts and can take necessary actions.
* **Exceptions**: Incorrect low stock threshold settings.

**6. Sales Report Generation**

* **Actors**: Owner, Cashier
* **Preconditions**: User is logged in and has permission to access sales data.
* **Basic Flow**:
  1. User navigates to the sales report page.
  2. The **Owner** can generate daily, weekly, or monthly sales reports with detailed insights and export them.
  3. The **Cashier** can generate basic sales reports (e.g., daily) and view summary data.
* **Exceptions**: No sales data available for the selected period.

**7. Exporting Reports or Inventory**

* **Actors**: Owner, Cashier
* **Preconditions**: Reports or inventory are available to be exported.
* **Basic Flow**:
  1. User selects the report (e.g., sales or inventory) and chooses an export format (CSV, PDF).
  2. User downloads or shares the exported file.
* **Exceptions**: Data export failure, unsupported formats.

**8. User Management (Owner only)**

* **Actors**: Owner
* **Preconditions**: Owner is logged in and has user management permissions.
* **Basic Flow**:
  1. Owner navigates to the user management page.
  2. Owner can create, update, or delete users (cashiers, stock managers).
  3. Owner assigns roles (e.g., cashier, stock manager) to users.
* **Exceptions**: Invalid user data, failed role assignment.

**Summary of Use Cases:**

1. **User Authentication** (Login/Signup/Forgot Password)
2. **Dashboard Access** (Owner Dashboard, Cashier Dashboard)
3. **Billing Management** (Owner, Cashier)
4. **Stock Management** (Owner, Stock Manager)
5. **Low Stock Notifications** (Owner, Stock Manager)
6. **Sales Report Generation** (Owner, Cashier and Stock Manager)
7. **Exporting Reports/Inventory** (Owner, Cashier)
8. **User Management** (Owner)