**Money Scam Detector Project**

**Requirement Specification Document**

**20-01-2025**

Revision List

|  |  |  |  |
| --- | --- | --- | --- |
| Ver.Rev | Date | Authors | Description |
| 1.0 |  | Noumya R R | Initial Document |

TABLE OF CONTENT

1. PROJECT/PRODUCT OVERVIEW------------------------------------ 2
2. USER STORIES----------------------------------------------------------- 3
3. HARDWARE AND SOFTWARE REQUIREMENTS---------------6
   1. BACKEND--------------------------------------------------------------6
   2. FRONTEND------------------------------------------------------------7
   3. HOSTING AND DEPLOYMENT (OPTIONAL)--------------------7
4. PROJECT/PRODUCT OVERVIEW

The **Money Scam Detector** is a rule-based fraud detection system that identifies potential fraudulent activities and scams in online financial transactions and communications. The system is built using the **Flask** framework for the backend, **SQL** for data storage, and provides real-time alerts for suspicious activities, such as transactions or emails that exhibit scam-like patterns.

1. **USER STORIES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **User Story** | Priority | **Status** | **Acceptance Criteria** |
| US001 | As a user, I want to register an account so that I can start using the scam detection system. | High | To Do | - User can create an account by providing email and password. |
| - The system stores user details securely. |
| US002 | As a user, I want to log in to my account so that I can access my transaction and email history. | High | To Do | - User can log in with email and password. |
| - A JWT token is returned upon successful authentication. |
| US003 | As a user, I want to submit a transaction for analysis so that I can check if it's fraudulent. | High | To Do | - User can submit transaction details (amount, timestamp, etc.) for analysis. |
| - The system provides a fraud score based on predefined rules. |
| US004 | As a user, I want to view my transaction history so that I can keep track of all previous transactions. | Medium | To Do | - User can retrieve a list of transactions associated with their account. |
| - Each transaction shows details like amount, date, and status. |
| US005 | As a user, I want to receive real-time alerts about suspicious transactions or emails.z | High | To Do | - User receives an alert when a transaction or email is flagged as suspicious. |
| - Alerts are delivered via WebSocket, email, or SMS. |
| US006 | As a user, I want to submit an email for scam analysis to detect fraudulent email content. | High | To Do | - User can submit an email by providing content for analysis. |
| - The system identifies scam keywords, suspicious links, or phishing phrases. |
| US007 | As a user, I want to view a list of flagged emails so that I can review and take action. | Medium | To Do | - User can view all flagged emails in their dashboard. |
| - Each flagged email displays its status (e.g., "flagged"). |
| US008 | As an admin, I want to view all flagged transactions so that I can investigate potential fraud. | High | To Do | - Admin can access a list of flagged transactions. |
| - Admin can see transaction details, including scam score and status. |
| US009 | As an admin, I want to resolve flagged transactions by marking them as legitimate or fraudulent. | High | To Do | - Admin can mark flagged transactions as "legitimate" or "fraudulent." |
| US010 | As an admin, I want to generate reports to track the performance of the fraud detection system. | Medium | To Do | - Admin can generate reports summarizing fraud detection results. |
| - Reports should include false positives, false negatives, and detection accuracy. |
| US011 | As a user, I want to view my alerts so that I can check the status of suspicious activities. | Medium | To Do | - User can access a list of all alerts. |
| - Each alert displays the associated transaction/email and status. |
| US012 | As a user, I want to receive email notifications about flagged transactions and emails so that I am notified about potential scams. | High | To Do | - User receives an email notification for every flagged transaction or email. |
| US013 | As a user, I want to be notified via SMS for critical alerts about potential scams. | Low | To Do | - User can opt-in for SMS notifications. |
| - Critical alerts (e.g., large transactions or phishing emails) trigger SMS alerts. |
| US014 | As a user, I want to update my account information so that my contact details are current. | Low | To Do | - User can edit their email or password. |
| - The system updates the database with the new information. |
| US015 | As a user, I want to check for unusual transaction patterns based on predefined rules so that I can identify potential fraud. | High | To Do | - The system analyzes transactions based on predefined rules and flags suspicious patterns (e.g., large amounts, frequent transfers). |
| US016 | As a user, I want to check email content for common scam keywords so that I can identify phishing attempts. | High | To Do | - The system flags emails containing scam-related keywords (e.g., "urgent," "verify your account"). |
| US017 | As an admin, I want to modify detection rules to improve fraud detection capabilities as new scam patterns are discovered. | Medium | To Do | - Admin can add, modify, or delete detection rules. |
| - Changes to rules are applied in real-time. |
| US018 | As a user, I want to log out of my account securely so that my session is terminated. | Low | To Do | - User can log out from the system, which invalidates the session token. |
| US019 | As a user, I want to see the status of flagged transactions or emails so that I can act on them promptly. | Medium | To Do | - Flagged items should show their status ("resolved," "unresolved") on the user’s dashboard. |
| US020 | As an admin, I want to review user activity logs for auditing purposes. | Low | To Do | - Admin can access logs of user actions and system events for audit and troubleshooting. |

1. **HARDWARE AND SOFTWARE REQUIREMENTS**

**3.1 Backend**

**Programming Language**:

* + **Python** (for developing the backend application using the Flask framework).

**Framework**:

* + **Flask** (Python web framework for building the application server).

**Real-Time Notifications**:

* + **Flask-SocketIO** (for handling WebSocket-based real-time notifications).

**Database**:

* + **SQL Database** (either PostgreSQL or MySQL).
  + **SQLAlchemy** (for interacting with the SQL database from Python).

**Authentication & Security**:

* + **Flask-Login** (for user authentication).
  + **Flask-Bcrypt** (for password hashing).
  + **JWT (JSON Web Tokens)** (for secure, stateless authentication).

**3.2 Frontend**

**Frontend Framework**:

* + **React.js** or **Vue.js** (for building the user interface and dashboards).

**HTML/CSS/JavaScript**:

* + HTML5, CSS3, and JavaScript for standard web application frontend development.

**UI Libraries**:

* + **Material-UI** or **Bootstrap** (for creating a responsive and user-friendly interface).

**Real-Time Web Interface**:

* + **WebSocket API** (for real-time updates/alerts to the user interface).

**3.3 Hosting and Deployment (Optional)**

**Cloud Service**:

* + **Heroku**, **AWS**, **Google Cloud**, or **DigitalOcean** for hosting the backend server and database.

**Web Server**:

* + **Nginx** or **Apache** (for deploying Flask-based applications).