[Project Name]: Expense Management System

Requirement Document

1. Introduction

- The expense management system is designed to help individuals or businesses track and manage their expenses efficiently.

- The application will be implemented in Python and will utilize file storage for persisting expense data.

2. Functional Requirements

2.1. Expense Creation

- Users should be able to add new expenses by providing details such as the date, category, description, and amount.

- The application should validate user input for required fields and appropriate formats.

2.2. Expense Listing

- Users should be able to view a list of all expenses recorded in the system.

- The list should display essential information such as the date, category, description, and amount of each expense.

2.3. Expense Filtering and Sorting

- Users should have the ability to filter and sort expenses based on criteria such as date, category, or amount.

- Filtering options may include date range selection or category-based filtering.

2.4. Expense Modification and Deletion

- Users should be able to edit or delete existing expenses.

- The system should validate user input during modification and confirm deletion requests.

2.5. Expense Reporting

- The application should generate reports summarizing expenses based on different criteria, such as date range, category, or monthly spending.

- The reports can be displayed in the console or exported to a file.

3. Non-Functional Requirements

3.1. User Authentication

- The system should provide authentication mechanisms to ensure that only authorized users can access and modify expense data.

- User roles and access levels can be implemented to differentiate between regular users and administrators.

3.2. Performance

- The application should provide fast responses, even with a large number of recorded expenses.

- The response time for listing, filtering, and reporting should be minimal.

3.3. Data Persistence

- The expense data should be stored in a file-based format for persistence.

- The system should handle file I/O operations to read, write, and update expense data.

4. User Interface

- The application should have a user-friendly console-based interface for ease of use and simplicity.

- Clear instructions and prompts should be displayed to guide users through different functionalities.

5. Constraints

- The application should be implemented using Python version 3 or later.

- File storage should be utilized for persisting expense data.

- Third-party libraries should be used only if necessary and should be compatible with Python 3.

6. Future Enhancements

- The application can be expanded to include additional features, such as expense categories management, data visualization, or integration with external services for currency conversion or receipt scanning.

- A web-based or mobile version of the application could be developed for broader accessibility and user convenience.

7. Assumptions and Dependencies

- It is assumed that users have basic knowledge of operating a console-based application and are familiar with expense management principles.

- The availability and accuracy of expense data depend on the reliability of file storage and the system's data integrity.

8. Project Timeline and Deliverables

- The project will be executed in multiple phases, including planning, development, testing, and deployment.

- The team will work collaboratively to deliver the following milestones:

- Phase 1: Requirement gathering, project planning, and data storage design.

- Phase 2: Backend development, including expense creation, listing, modification, and deletion.

- Phase 3: Frontend development, console

interface design, filtering, sorting, and reporting functionalities.

- Phase 4: User authentication implementation, performance optimization, and error handling.

- Phase 5: Testing, bug fixing, documentation preparation, and final deployment.

9. Risks and Mitigation Strategies

- Potential risks include data loss, file corruption, or security vulnerabilities.

- Mitigation strategies may involve implementing regular backups, data validation mechanisms, and secure file access protocols.

10. Budget and Resources

- The project team will require access to development tools, documentation, and libraries.

- A budget should be allocated for any necessary paid libraries or external resources.

- The team will utilize version control systems, project management tools, and communication platforms for collaboration.

11. Conclusion

- The expense management system project aims to deliver a user-friendly and efficient application for tracking and managing expenses.

- By utilizing Python and file storage, the project will provide a convenient solution for individuals and businesses.

- The project team will work collaboratively to ensure the successful implementation, testing, and deployment of the application.

----------------------------------------------