Final Documentation Topics

1. **Project Overview**
    - Project objectives and purpose
    - Key features and functionality overview
    - Technologies used (e.g., .NET Core, C#, SQLite/CSV/JSON for storage, etc.)
    - System requirements and dependencies (software/hardware)
2. **Installation and Setup Guide**
    - Installation and setup
    - Database setup (if applicable, e.g., SQLite configuration or CSV/JSON file structure)
3. **User Guide**
    - **Login and Account Setup**: How to create a new user with username, password, and currency type.
    - **Managing Transactions**:
        - Adding cash inflows (credit, gain, budget)
        - Adding cash outflows (debit, spending, expenses) and enforcing balance check
        - Adding and tracking debts (source, due date, and clearance)
        - Adding custom notes and categories to transactions
        - Tagging transactions (Default and custom tags)
    - **Dashboard Features**:
        - Viewing statistics (total inflows, outflows, debts, cleared debts)
        - Sorting and filtering transactions by various criteria (tags, transaction type, etc.)
        - Viewing pending debts and filtering by date range
    - **Search and Filtering**:
        - Search functionality (by title, type, date range, tags)
    - **Import and Export**:
        - Steps for importing data (CSV/JSON/Excel)
        - Exporting data to different formats (CSV/JSON/Excel)
    - **Handling Errors and Troubleshooting**: Common error messages and their solutions.
4. **System Design and Architecture**
    - **Database Schema**: Sqlite or JSON
    - **ERD Diagrams**: ERD Diagram
    - **Class and Code Structure**:
    - **API and External Interfaces**: If the application interacts with any external services (e.g., currency conversion APIs)
5. **Security and Privacy**
    - User data handling, password security (e.g., encryption)
6. **Maintenance and Future Enhancements**
    - How to update the application
    - Known limitations and potential future features (e.g., multi-currency support, cloud sync)

7. **Appendix**
    - Glossary of terms (tags, transaction types, etc.)
    - Detailed list of default tags and date ranges
    - Example files (CSV/JSON) for importing/exporting data

For Test Case: Do simple Test and put screenshot in document(proof of work).

- **User Account Creation**:
    - Test case to ensure a new user can create an account with a username, password, and currency.
    - Test case to verify username uniqueness and password validation.
- **Login and Authentication**:
    - Test case to verify successful login with valid credentials.
    - Test case for unsuccessful login with invalid credentials.
    - Test case to ensure password is encrypted and not stored in plain text.
- **Transaction Entry**:
    - Add valid cash inflow (credit/gain/budget) and verify correct processing.
    - Add valid cash outflow (debit/spending/expenses) and ensure balance check works.
    - Add a debt and ensure correct tracking (source, due date).
    - Test with notes and tags and ensure they're correctly associated with transactions.
- **Debt Management**:
    - Test case to ensure debts are cleared correctly only from cash inflows.
    - Test the system's handling of pending debts (highlighting them correctly).
- **Tagging Transactions**:
    - Add and edit custom tags for transactions and verify proper filtering/sorting.
- **Filtering and Sorting**:
    - Test sorting by transaction type (credit, debit, debt).
    - Test filtering by date range, tags, transaction type, etc.
- **Dashboard Statistics**:
    - Verify that the statistics (total inflows, outflows, cleared debts, etc.) are accurate.
    - Ensure top 5 highest/lowest transactions are displayed correctly.
    - Verify that pending debts and other data are correctly listed.
- **Import/Export Data**:
    - Test importing data in various formats (CSV, JSON, Excel). **Loading the json file from system.**
    - Test exporting data to CSV/JSON/Excel. **Downloading the transaction details in excel file**
- **Currency Handling**:
    - Test the application with different currencies (e.g., USD, NPR) to ensure correct display and calculations.

2. UI/UX Test Cases

3. Performance Test Cases

- **Load Testing**:
- **Speed Testing**:

4. Usability Test Cases

- Test to ensure the application is intuitive and easy to use for a first-time user.

5. Compatibility Test Cases

- Verify the application works on different versions of Windows.

4. Security Test Cases

- **Data Protection**: Ensure passwords are stored securely (e.g., hashed with salt).