

**Web Technology Final Project**

**App Name: Expenses Tracker (WebApp)**

**A work present by Mubalama Mweze Jean-Claude (id:23515)**

**Welcome to the Expense Tracker App repository! It is a simple web application which allows users to manage and track their expenses.**

**A.** Requirements Identification:

**1.User Authentication:**

**Login:** Users can log in with their credentials to access their expense records.

**Sign Up:** New users can create an account by registering with required details if they don't have an existing account.

**2.Expense Operations:**

**Adding Expenses:** Users can add new expenses with details such as amount, date, category, description, etc.

**Updating Expenses:** Ability to edit existing expenses to modify details.

**Deleting Expenses:** Capability to remove unwanted expenses from the records.

**3.Expense Summary:**

**View Summaries:** Display expenses in a table format with details such as date, amount, category, and description.

**Total Amount Spent:** Show a summary of the total amount spent within a specified timeframe.

**4.Filtering Options:**

**Filter by Year/Month:** Users can filter expenses based on specific months or years.

**Filter by Expense Type:** Capability to filter expenses by different categories or types.

**5.Pagination:**

**Paginated View:** Display expenses in manageable chunks with pagination controls for easy navigation.

Customizable Expense Categories:

**6.Add/Edit Categories:** Ability to customize expense categories to align with the user's preferences or needs.

**7.Download Functionality:**

Export as CSV: Allow users to download expense records in CSV format for external use or analysis.

**8.Responsive Design:**

Cross-device Compatibility: Ensure the application is accessible and functional across various devices (desktops, tablets, mobile phones) with a responsive design.

**B. Expense Tracker User Guide**

**1. Logging In:**

To access the Expense Tracker, use the provided login credentials (username/password) or sign up if you're a new user.

**2. Dashboard Overview:**

Upon login, you'll land on the dashboard displaying a summary of your expenses.

The dashboard includes:

Total amount spent overview.

Filters to sort expenses by year, month, or category.

Quick access to adding, editing, or deleting expenses.

**3. Adding Expenses:**

To add a new expense:

Click on the "Add Expense" or "New Transaction" button.

Fill in the required details: amount, date, category, and description.

Save the entry to record your expense.

**4. Editing/Deleting Expenses:**

To modify existing expenses:

Navigate to the specific expense within the expense table.

Click on the edit icon to make changes or delete to remove the expense.

**5. Filtering Options:**

Use the filters available to view expenses for specific time frames (year, month) or by expense type/category.

Select the desired criteria to refine your expense view.

**6. Expense Categories:**

Customize expense categories:

Navigate to settings or profile.

Access the "Categories" section to add, edit, or remove expense categories.

**7. Downloading Expenses:**

To export expense records:

Click on the "Download" or "Export" option.

Choose the CSV format to download your expense data for external use.

**8. Responsive Design:**

The application is designed to work seamlessly across devices (desktop, tablet, mobile).

Enjoy a consistent experience regardless of the device you're using.

**9. Account Management:**

Access your profile/account settings to update personal details or change passwords.

**10. Logging Out:**

To log out securely:

Use the logout option available in the profile/account settings.

**Additional Notes:**

For any further assistance or queries, refer to the help section within the application or reach out to customer support.

**C. Expense Tracker Technical Documentation**

**1. Architecture Overview:**

Frontend: HTML, CSS, and Bootstrap used for creating the user interface (UI).

Backend: Java 17.0 with Spring Boot 3 as the framework for server-side logic.

Database: MySQL utilized as the primary data storage system.

**2. Implementation Details:**

**Frontend Implementation:**

**Thymeleaf Templating:** Leveraging Thymeleaf for server-side templating to generate dynamic HTML pages.

**Bootstrap:** Employing Bootstrap for responsive and mobile-first design elements.

**Backend Implementation:**

**Spring Boot:** Utilizing Spring Boot for rapid development and auto-configuration of the application.

**Spring Data JPA:** Utilizing Spring Data JPA for data access and management, simplifying database operations.

**Spring Security:** Implementing Spring Security for authentication and authorization.

**3. Technical Libraries and Frameworks:**

**Frontend:** HTML, CSS, Bootstrap for UI design and responsiveness.

**Backend:** Java 17.0, Spring Boot 3, Spring Data JPA, and Spring Security for server-side logic, data access, authentication, and authorization.

**4. Database Connectivity:**

**MySQL:** Using MySQL as the database management system to store expense-related data.

**5. Server Configuration:**

**Server:** Deploying the application on Tomcat, the built-in server of Spring Boot.

**6. Additional Technical Details:**

**Security Measures:** Utilizing Spring Security for robust user authentication and authorization mechanisms.

**Error Handling:** Employing robust error handling mechanisms both on the frontend and backend for a seamless user experience.

**Deployment:** Deploying the application on cloud platforms or servers using Tomcat as the server.

**7. Build and Dependency Management:**

**Maven:** Using Maven for managing project dependencies and build processes.

**8. APIs and Data Handling:**

**RESTful APIs:** Designing RESTful APIs using Spring MVC for managing expenses through CRUD operations.

**Spring Data JPA:** Interacting with the MySQL database to handle data access operations.

**9. Scalability and Performance:**

Ensuring scalability through Spring Boot's modularity and ease of scaling.

Implementing caching mechanisms to optimize application performance.

**10. Documentation and Version Control:**

Maintaining comprehensive documentation using tools like Swagger for API documentation.

Employing Git for version control and collaborative development.

**D. Database Schema**

