Monthly Budget

Software Requirements Specification

Version 1.0

Table of Contents

[Introduction 3](#_Toc328746642)

[Purpose 3](#_Toc328746643)

[Scope 3](#_Toc328746644)

[Definitions, Acronyms and Abbreviations 3](#_Toc328746645)

[References 3](#_Toc328746646)

[Technologies to be used 3](#_Toc328746647)

[Overall Description 5](#_Toc328746648)

[Product Perspective 5](#_Toc328746649)

[Software Interface 5](#_Toc328746650)

[Hardware Interface 5](#_Toc328746651)

[Client Side 5](#_Toc328746652)

[Server Side 5](#_Toc328746653)

[Product Function 6](#_Toc328746654)

[User Characteristics 6](#_Toc328746655)

[Constraints 6](#_Toc328746656)

[Architecture Diagram 7](#_Toc328746657)

[Use-Case Model Description 7](#_Toc328746658)

[Package Diagram 8](#_Toc328746659)

[Database Design 9](#_Toc328746660)

[EER Diagram 9](#_Toc328746661)

[Assumptions and Dependencies 9](#_Toc328746662)

[Specific Requirements 10](#_Toc328746663)

[Functional Requirements 10](#_Toc328746664)

[Non-Functional Requirements 11](#_Toc328746665)

[Use case Reports 11](#_Toc328746666)

# Introduction

## Purpose

To build an application that help in planning monthly budget and to update related users by mail. The application will allow administrator to plan the budget for the month and update the user with the mail about the budget allocation. It also allow user to update the expenditure under the supervision of administrator.

## Scope

* Create different user role with different privileges.
* Manage all the account details of users from one central location.
* Track all the users and their transaction details.
* View all the details of all the user interactions.
* User will authenticate to avail the services, the service can be third party authentication or own implementation.
* Maintain logs of all the activities performed in the system by the users for auditing.
* Providing help for basic understanding of facilities.
* Maintain a centralized database.

## Definitions, Acronyms and Abbreviations

* Income item: where the amount flows in to the system.
* Expense item: form where the amount goes out of the system.
* User: the users associated with the system.
* Role: the role the user is playing in the system.
* Budget: is the money planning for the particular month.
* Transaction: is the entry in the system containing the details for the event.
* Item: can be anything that can change the state of the system.

## References

* IEEE SRS Format

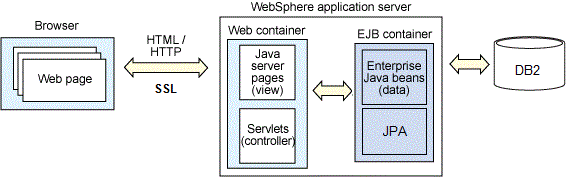
## Technologies to be used

* Database: MySQL.
* Language: J2EE.
* Framework: Spring, Hibernate.
* Front end: HTML, JSP, JQuery.

# Overall Description

Describe the general factors that affect the product and its requirements.

## Product Perspective



## Software Interface

* **Client on Internet:** Web Browser, Operating System (any)
* **Web Server:** WAS, Operating System (any)
* **Data Base Server:** MySQL, Operating System (any)
* **Development End:** Spring Tool Suite (J2EE, Java, Java Bean, Servlets, HTML), DB2, OS (Windows), Web Server.

## Hardware Interface

### Client Side

* **Processor:** Pentium II at 500 MHz or higher
* **RAM :** 64 MB or greater
* **Disk Space**: 1 GB or greater
* **Browser**: Internet Explorer 7 or later

### Server Side

* **Processor:** Pentium Dual Core at 1GHz or higher
* **RAM :** 2 GB or greater
* **Disk Space**: 14 GB or greater.

## Product Function

* **Budget**
  + Allow to plan currency allocation for the expense item
  + Display currency left for the budget item.
* **Accounts**
  + Allow to create new account and assign user to it.
  + Feature to include account in monthly budget.
* **Logbook**
  + Will display all the transaction done till the date in descending order.
* **New entry**
  + Allow to create new transaction.
  + Transaction can be expense, income and transfer.
* **Option**
  + Allow to change currency.

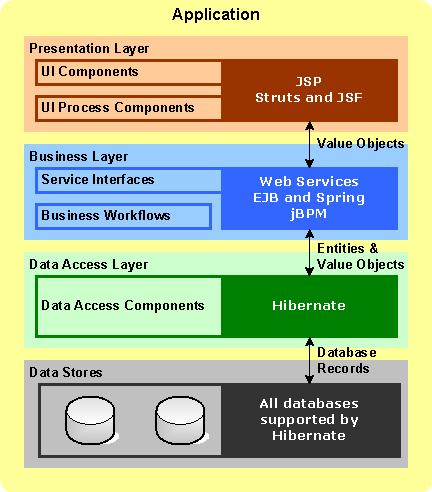
## User Characteristics

* Every user should be comfortable of working with computer and browsing.

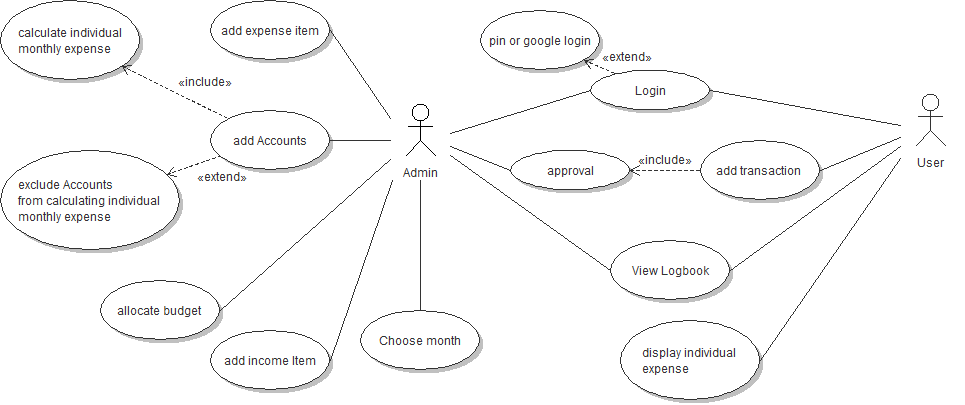
## Constraints

* GUI is only in English.
* User must have Google account for login.
* This project is implemented on cloud infrastructure.
* There is no maintainability of back up.
* Limited to HTTP/HTTPS.

## Architecture Diagram

****

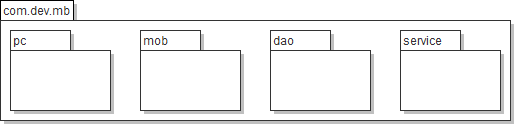
## Use-Case Model Description



**Administrator:** Administrator is responsible for the adding the account, expense item, Income Item, Allocate budget for expense item and approving the user transaction. Administrator has access to add, view and edit the logbook.

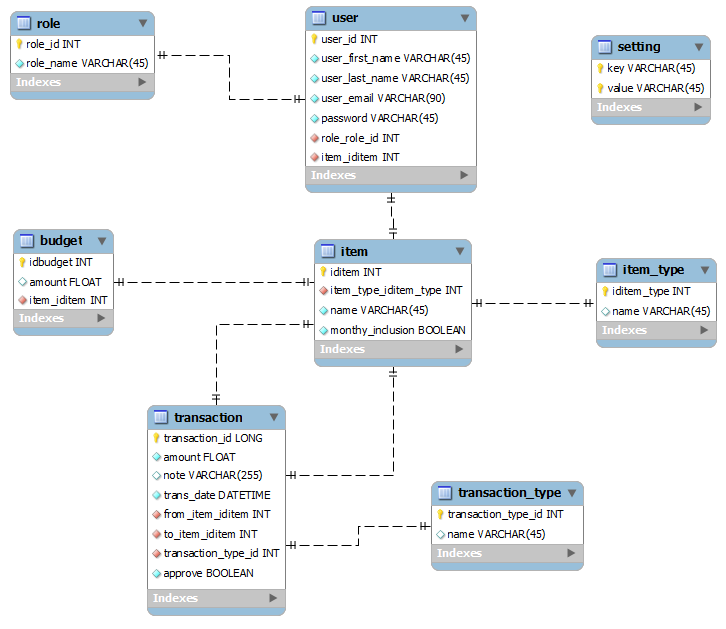
**User:** Users is responsible for adding the transaction, View logbook, view individual expenditure.

## Package Diagram



## Database Design

### EER Diagram



## Assumptions and Dependencies

* User will pay his/her individual budget timely and in any means outside this system.
* Any transfer or payment is done conceptually.
* Login is subjected to Google service availability.

# Specific Requirements

## Functional Requirements

* **Accounts**
  + Ability to add Accounts
    - Example: Hemant, Nilesh, Nishant & Monthly cash
  + Ability to add expense item
    - Example: Add Rent as Expense item
  + Ability to add income Item
    - Example: Add Accounts as Income Item
    - If added as income item, it should calculate for individual monthly expense by default
  + Ability to exclude Accounts from calculating individual monthly expense.
  + It should show individual expense for each individual (Account).
  + Income account should be mapped to the users account. Cash account is always mapped to administrator role.
* **New entry**
  + Ability to add transaction for expense, income and transfer
    - Example: Expense (only option available for users)
      * Amount, date, Account, Category, Note
    - Example: Income
      * Amount, date, Account, Category, Note
    - Example: Transfer
      * Amount, date, Account, To Account, Note
  + Users can add transactions but need approval from administrator
* **Budget**
  + Each expense item can have budget for each month.
  + Ability to allocate budget for any expense category
    - Example: Category, Monthly Budget
  + Cary out savings of this month to next month.
  + Savings should be income for next month.
* **Setting**
  + Ability to change administrator password.
  + Ability to change administrator username.
  + Ability to change currency symbol.
* **Login/Logout**
  + Google API can be implemented.
* **Logbook**
  + Should be able to view all records of transaction by month
  + User should be able to see individual monthly expense

## Non-Functional Requirements

* Secure access of confidential data (user’s details). SSL can be used.
* 365 X 24 X 7 availability.
* Better component design to get better performance at peak time.
* Flexible service based architecture will be highly desirable for future extension.
* Customizing skins
* Mobile Interface

## Use case Reports