Expense Manager

**MCA 6th Semester**

**MCSP-060 (Project Report)**

**Anirban Nandy**

**Enrollment No-105057536**

# Index

Contents

[**MCA 6th Semester** 1](#_Toc351476508)

[**MCSP-060 (Project Report)** 1](#_Toc351476509)

[**Anirban Nandy** 1](#_Toc351476510)

[**Enrollment No-105057536** 1](#_Toc351476511)

[Index 2](#_Toc351476512)

[Introduction: 4](#_Toc351476513)

[**Figure:** Overview of Daily notebook & Social Networking Updater 5](#_Toc351476514)

[Objective: 5](#_Toc351476515)

[System Analysis 6](#_Toc351476516)

[Identification of Need: 6](#_Toc351476517)

[Preliminary Investigation: 6](#_Toc351476518)

[Feasibility Study: 6](#_Toc351476519)

[Project Planning & Scheduling: 6](#_Toc351476520)

[Gantt chart 6](#_Toc351476521)

[Tracking Gantt 7](#_Toc351476522)

[Pert Chart 7](#_Toc351476523)

[Software requirement specifications (SRS): 8](#_Toc351476524)

[Functional Requirement 9](#_Toc351476525)

[Non-functional Requirements 11](#_Toc351476526)

[Software Engineering Paradigm applied 12](#_Toc351476527)

[Data models 12](#_Toc351476528)

[Context Diagram 12](#_Toc351476529)

[0-Level DFD 13](#_Toc351476530)

[1-Level DFD 14](#_Toc351476531)

[2-Level DFD 15](#_Toc351476532)

[Sequence diagrams 15](#_Toc351476533)

[Entity Relationship Model, 18](#_Toc351476534)

[Class Diagrams 19](#_Toc351476535)

[Activity Diagrams 20](#_Toc351476536)

[System Design 23](#_Toc351476537)

[Modularisation details 23](#_Toc351476538)

[EXPENSE MANAGER Engine: 23](#_Toc351476539)

[EXPENSE MANAGER GUI: 23](#_Toc351476540)

[EXPENSE MANAGER Storage: 23](#_Toc351476541)

[Google Calendar: 23](#_Toc351476542)

[Facebook/Twitter/LinkedIn API: 24](#_Toc351476543)

[Data integrity and constraints 24](#_Toc351476544)

[Entity integrity 24](#_Toc351476545)

[Referential Integrity 24](#_Toc351476546)

[Domain Integrity 24](#_Toc351476547)

[User Defined Integrity 24](#_Toc351476548)

[Database design 24](#_Toc351476549)

[User Interface Design 28](#_Toc351476550)

[Test Cases (Unit Test Cases and System Test Cases) 28](#_Toc351476551)

[Coding 28](#_Toc351476552)

[Complete Project Coding 28](#_Toc351476553)

[Comments and Description of Coding segments 28](#_Toc351476554)

[Standardization of the coding 28](#_Toc351476555)

[Code Efficiency 28](#_Toc351476556)

[Error handling 29](#_Toc351476557)

[Parameters calling/passing 29](#_Toc351476558)

[Validation checks 29](#_Toc351476559)

[Testing 29](#_Toc351476560)

[Testing techniques and Testing strategies used 29](#_Toc351476561)

[Testing Plan used 29](#_Toc351476562)

[Test reports for Unit Test Cases and System Test Cases 29](#_Toc351476563)

[Debugging and Code improvement: 29](#_Toc351476564)

[System Security measures: 29](#_Toc351476565)

[Database/data security: 29](#_Toc351476566)

[Creation of User profiles and access rights 29](#_Toc351476567)

[Cost Estimation of the Project along with Cost Estimation Model 29](#_Toc351476568)

[Reports (sample layouts should be placed) 30](#_Toc351476569)

[Future scope and further enhancement of the Project 31](#_Toc351476570)

[Bibliography 31](#_Toc351476571)

[Appendices (if any) 31](#_Toc351476572)

[Glossary. 31](#_Toc351476573)

# Introduction & Objective:

Expense Manager Software will let users track expenses and earnings. This software is basically developed for individual users so that they can track their daily expense, income and keep track of their money but this application will be useful to a company as well. Users can add daily earnings & expenses. It is to know how and where we are spending our money. It is the perfect system for individuals, families, and small business to manage their income and expense tracking. The friendly user interface makes it quick and easy to capture your income and expense transactions for each day. Expenses management software updates all our accounts daily so we'll always have access to the most current and accurate information about our finances and can clearly see how much money we have and owe, and where is your hard-earn money going.

At any point of time he can review his total expenses and generate report. There will be three different User Interfaces to allow the users use the software anywhere all the time. The User interfaces are Desktop Interface, Mobile Interface & Web Interface. The data from all the interfaces can be synced and merged to generate a final report.

The Desktop Interface is the main & fully featured version of the software. Users can add new data, browse old expenses and sync expenses from Mobile & Web Interfaces. The Mobile Interface will allow users to add data using their mobile and use customized feature set. While shopping people carry their mobile along with them and then they can add expenses and earnings instantly to avoid forgetting about certain expenses. The Web Interfaces provide the ultimate flexibility of login to the user accounts in the web and add/browse expenditures.



Fig 1: Different interfaces of Expense Manager Software

The main features of this software are listed below:

1. Calendar view to select any date and add/view the expenses & earnings.
2. Add tag with expenses so that user can remember the reason for spending
3. View Available balance
4. View Total expenses
5. Create a contact book for add contact of the person.
6. Generate weekly, monthly and yearly Expenses report Income report Profit and Loss report.
7. Online sync
8. Reminder facility for future expenses.
9. Searching the expenses.
10. On the go expense tracking using Mobile & Web Interfaces
11. Syncing data from all the interfaces such as: Desktop, Mobile & Web.

# System Analysis

## Identification of Need:

Nowadays people are so busy and technology dependants that they don’t even bother to remember their own phone number. And off course, I am one of those persons. One of the problems I have been facing since my childhood is that I really do not like to write down how much money did I spend in an entire day or in the entire month. Eventually I forget all those details. I used to struggle to provide exact details of my expenses to my parents. The problem has increased when I started earning. It can be easily understood that things really got more complicated and I also know that things are going to get worse when my income source and expense fields would increase. So I strongly felt the necessity of an application or software that would allow me to keep track all the details of my income and expenses easily and efficiently. I believe expense manager is a perfect application for that.

## Preliminary Investigation:

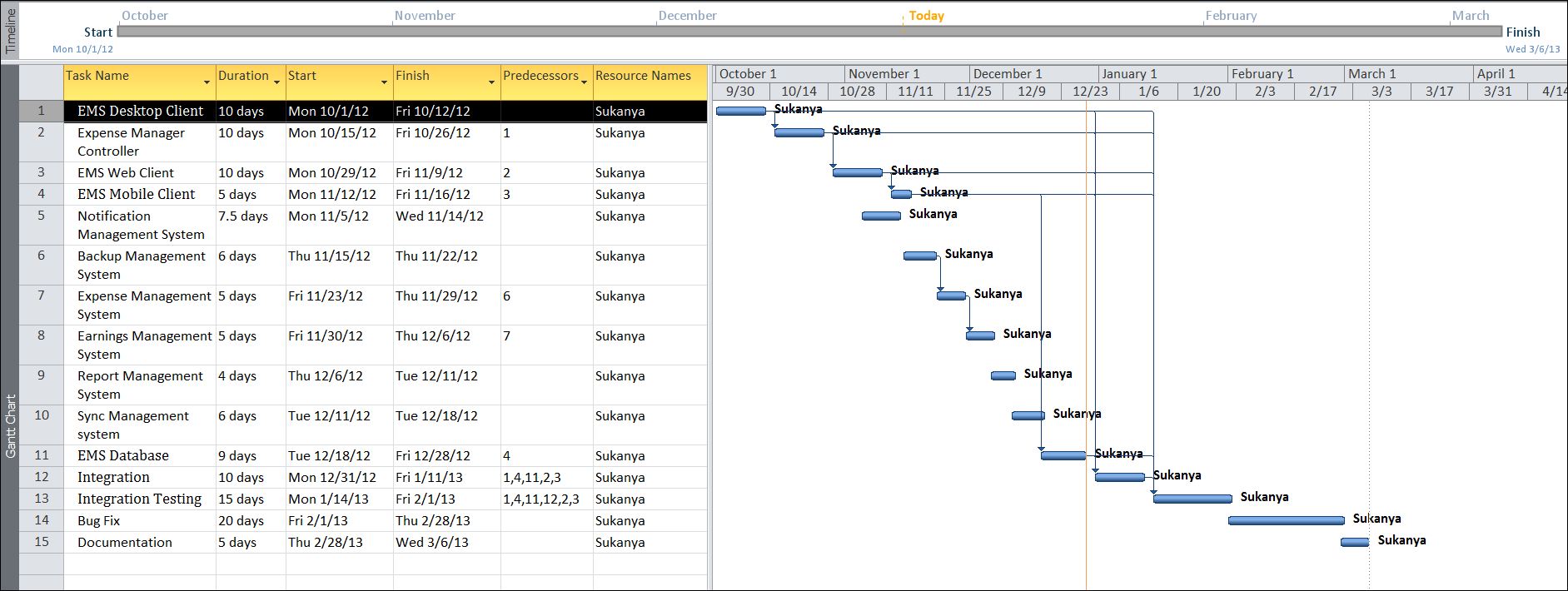
When I myself faced this problem and felt the need of an application, I started discussing about the problem and to my surprise, I found out that almost all of them are facing similar problems. Actually we nowadays are so technology dependant that we expect that all our tasks would be done by our computer or our mobile devices. So I noted down all the problems and their probable solution on a notepad and started consulting with some of my seniors who are IT professionals as well. After gathering all the information I felt this could be a really fruitful project.

## Feasibility Study:

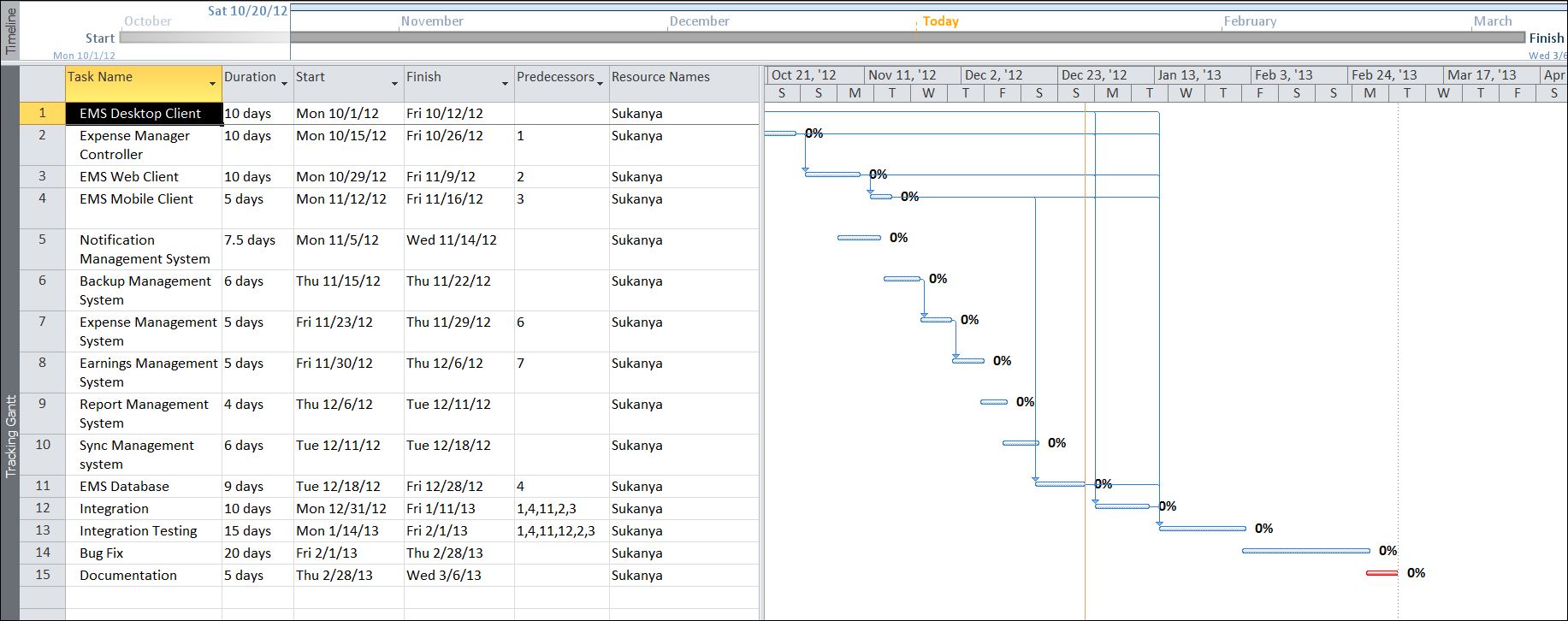
As explained before, people are now completely technology dependant and number of smart phone and computer users is increasing day by day. So, an application like expense manager would be really useful to them. With minimal effort people can manage and keep track the income and expense of their entire life. That is why, I know that after completion, this software will be used by thousands of users.

## Project Planning & Scheduling:

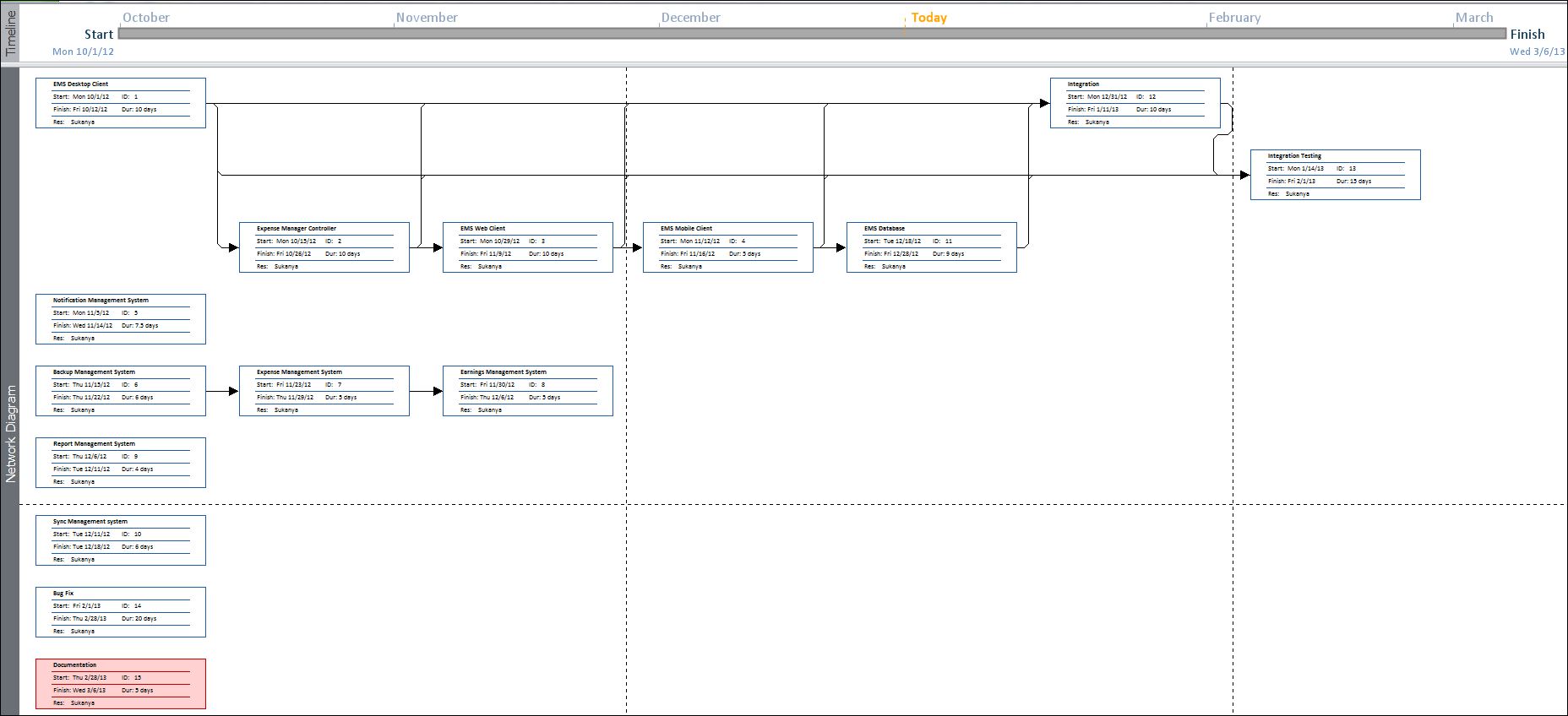
### Gantt chart



### Tracking Gantt



### Pert Chart



## Software requirement specifications (SRS):

### Functional Requirement

#### Enter new Expenses and Earning

**Introduction**

Entered new expenses and new earning store into the account.

**Input**

Earning and expenses data with purpose of earning apply for which class.

**Processing**

EMS saves the expense and earning details in database.

**Output**

EMS generates expense id and earning id for future reference.

#### View Report for the Income, expense and period transaction

**Introduction**

User can view the report for particular or total income, expenses of the weekly, monthly, yearly, or period transaction.

**Input**

Select weekly, monthly, yearly, or period transaction.

**Processing**

EMS queries the expense and earning details from database and prepares the report.

**Output**

User can see the report.

#### Graphical representation for the Income, expense of weekly, monthly, yearly and PERIOD TRANSACTION

**Introduction**

User can view the graphically for particular income, expenses of the weekly, monthly, yearly, or period transaction using timing and total graphically representation of the weekly, monthly, yearly, or period transaction.

**Input**

Select weekly, monthly, yearly, or period transaction.

**Processing**

EMS queries the expense and earning details from database and prepares the data to be displayed.

**Output**

User can see the graphically representation.

#### Search transaction

**Introduction:**

Search transaction for income and expense.

**Input:**

Select timing, date, name of expenses or earning.

**Processing:**

The **EMS** will search for the requirement.

**Output:**

The **EMS** will display the search result.

#### Sync web & mobile data in desktop

**Introduction:**

Sync web & mobile data from desktop application

**Input:**

Select web account or mobile device

**Processing:**

The **EMS** will sync with web account or mobile device and save the data in database

**Output:**

The **EMS** will generate a sync id and display confirmation message.

#### Changing Password and Username

**Introduction**

Change existing username and password

**Input**

New username and password

**Processing**

Old username and password will be replaced by user provided new username and password after authenticating.

**Output**

Password and Username can be changed according to the Employee requirement whenever they want to change for better security of the System.

#### Mobile data entry & query

**Introduction:**

**EMS** data can be entered and queried using a mobile device.

**Input:**

User will enter the expenses and earning in the mobile device.

**Processing:**

The device will stored the entered data and sync with Server while manual sync operation. While querying device will search its internal storage for the query and display the result.

**Output:**

The mobile device will display the search result.

#### Web data entry & query

**Introduction:**

**EMS** data can be entered and queried using a web interface.

**Input:**

Admin will new user details as well as search query.

**Processing:**

Web interface will store new entry in the Google doc storage and while searching it will search its internal storage. Web interface will sync with main server while manual sync.

**Output:**

Website will show all the related information.

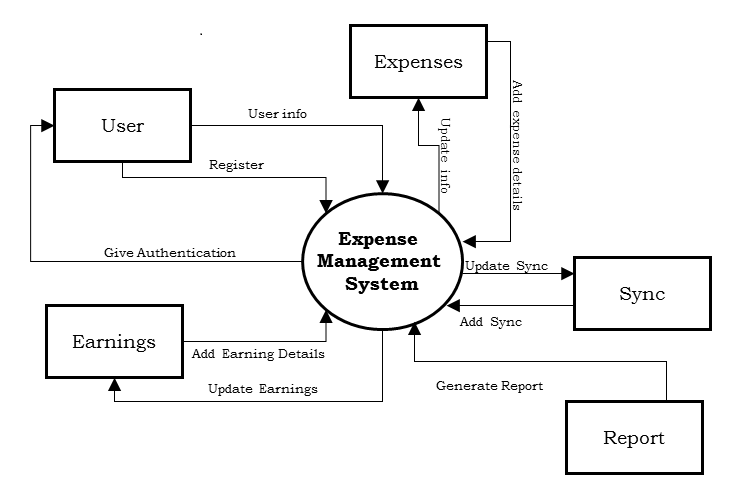
### Non-functional Requirements

* The application will be **self-dependent** and no dependency on other parties required.
* There will be a digital **backup** and restore system.
* There will be more **opportunity** to extend the application in various type of device in future.
* The response time will be low and the system will **response** fast.
* GUI should be easy to use and attractive as well.
* It will be very **user friendly** and **usable** by any person with minimal computer knowledge.
* In terms of **security** unauthorized access will be denied and register user will be able to change as necessary.
* It will be **efficient** as it reduces manual labor and searching.
* **EXPENSE MANAGER** will have user manual and help **documents**.
* It is designed such a way that it can be **maintained** with minimal effort.

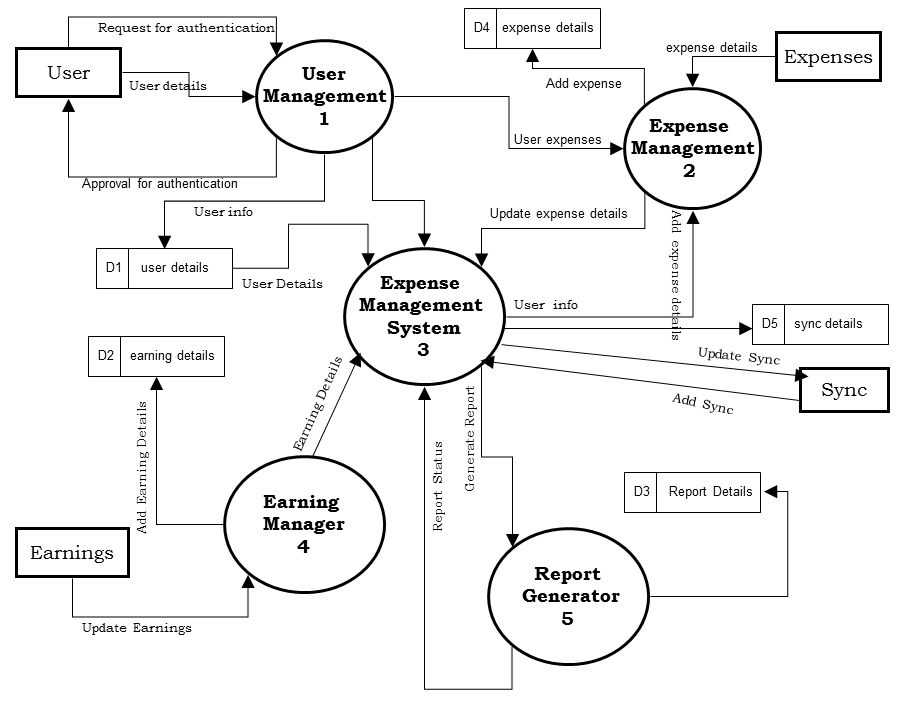
## Software Engineering Paradigm applied

## Data models

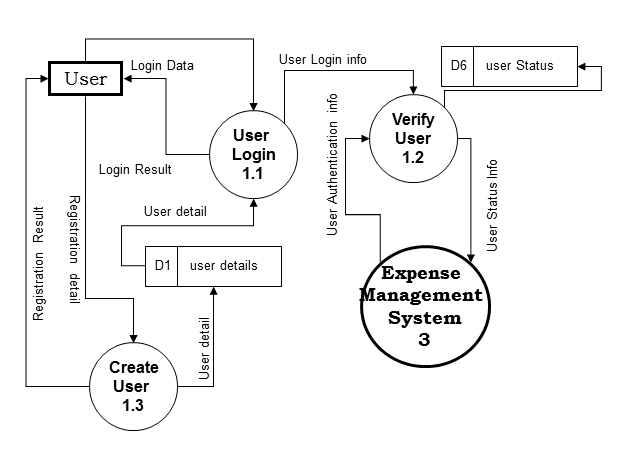
### Context Diagram

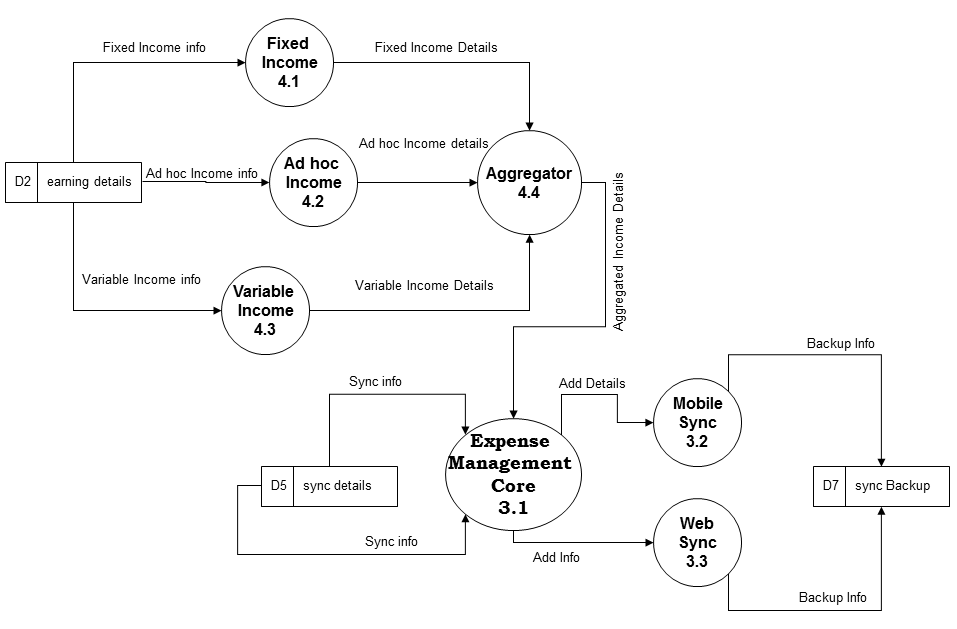


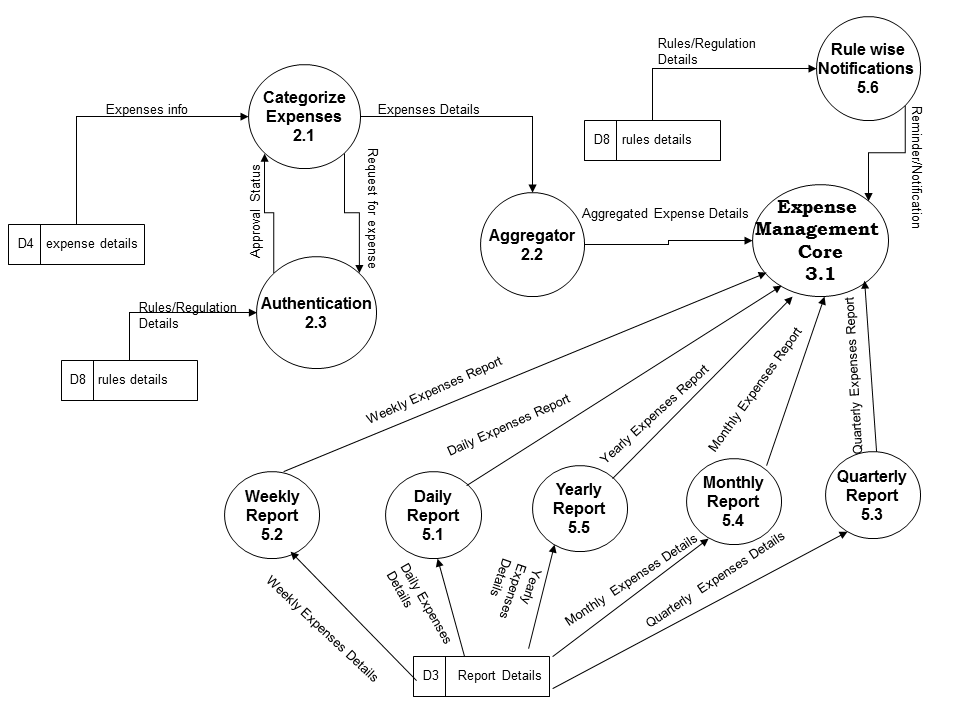
### 0-Level DFD



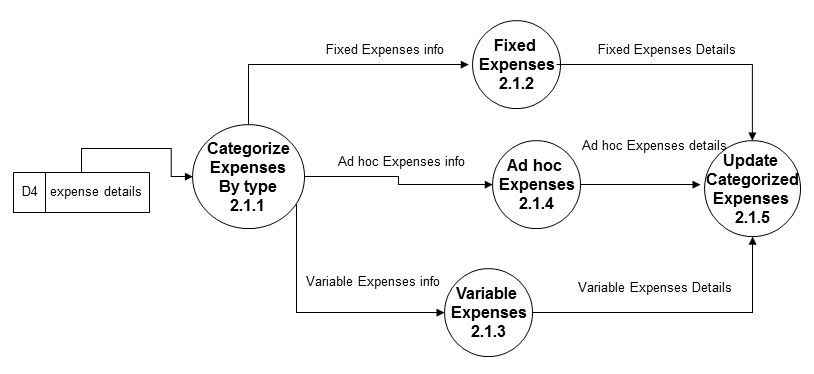
### 1-Level DFD







### 2-Level DFD



## Sequence diagrams

**Update Expense Report**

**:Login**

**User**

**:Register**

**User**

**Controller**

**:Update**

**Expenses**

**:Expenses**

**Report**

**:View**

**Update**

**Report**

**Register**

**Register**

**ModifyExpenses**

**AddExpenses**

**DeleteExpenses**

**ShowError**

**ShowError**

**ShowError**

**UpdateExpensReport**

**UpdateExpensReport**

**UpdateExpensReport**

**ShowError**

**ShowError**

**ShowError**

**ViewReport**

**ViewExpensesReport**

**DisplayExpensesReport**

**DisplayReport**

**Update Income Report**

**:Login**

**User**

**:Register**

**User**

**Controller**

**:Update**

**Income**

**:Income**

**Report**

**:View**

**Update**

**Report**

**Register**

**Register**

**ModifyIncome**

**AddIncome**

**DeleteIncome**

**ShowError**

**ShowError**

**ShowError**

**UpdateIncomeReport**

**UpdateIncomeReport**

**UpdateIncomeReport**

**ShowError**

**ShowError**

**ShowError**

**ViewReport**

**ViewIncomeReport**

**DisplayIncomeReport**

**DisplayReport**

**Sync Mobile Application and Desktop Application from Web Application**

**Register**

**Register**

**UpdateWApplication**

**SyncNotComplete**

**SyncComplete**

**SyncMApplication**

**SyncDApplication**

**DisplayStatus**

**SyncComplete**

**:Login**

**User**

**:Register**

**User**

**Controller**

**:Update**

**Web**

**Application**

**:Update**

**Mobile**

**Application**

**:Update**

**Desktop**

**Application**

## Entity Relationship Model

We will design a RDBMS for Daily notebook & Social Networking Updater. The entities and their attributes are listed below. Attributes in Bold letter is the unique key.

|  |  |
| --- | --- |
| **Entities** | **Attributes** |
| Note | **Note Id**, Content, Time, Size, Web Service id, user |
| RSS Feed | **RSS Feed Id** , Web Service Id, Content, time, size |
| Daily notebook & Social Networking Updater | **Sw Id**, Web Services Supported, Users, Size |
| User | **User Id**, Name, Social Network Data, Preferences. |
| Web Service | **Web Service Id,** Authentication Data, Feed Data, Preferences. |

**Relationship between Entities:**

* Daily notebook & Social Networking Updater has User 🡪 1 : N
* Users post Notes 🡪 1 : N
* Web Service generates Feeds 🡪 M : N

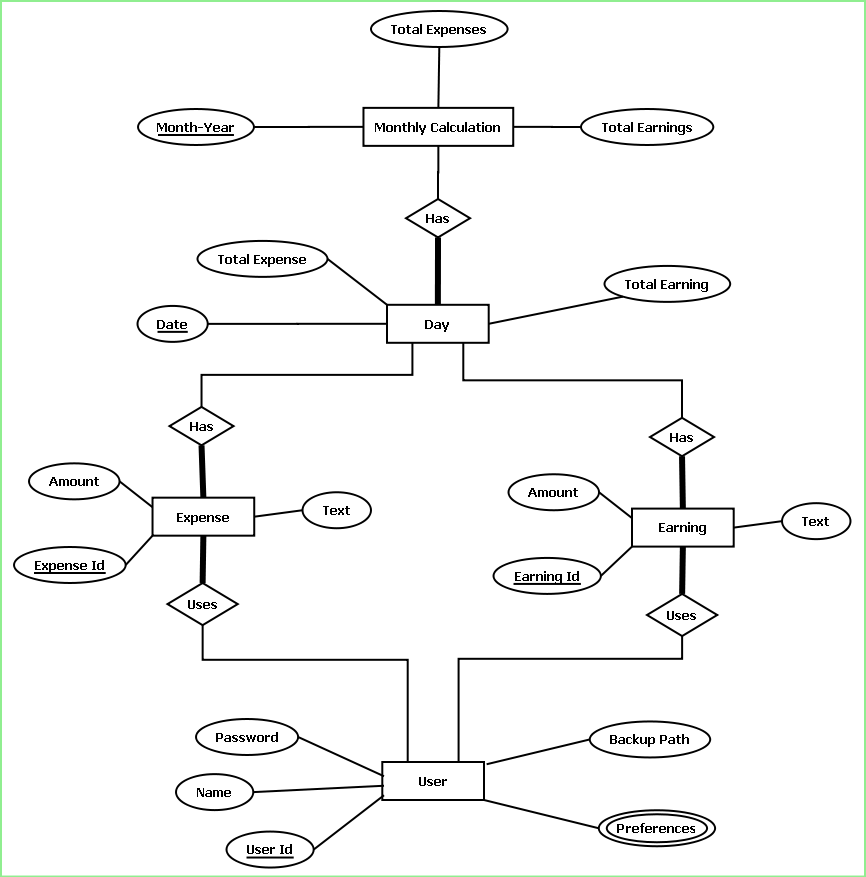
Expense Manager Database will be an optimized database which will save certain information about every expenses and earnings logged by User.

We will design a RDBMS for Expense Manager. The entities and their attributes are listed below. Attributes in Bold letter is the unique key.

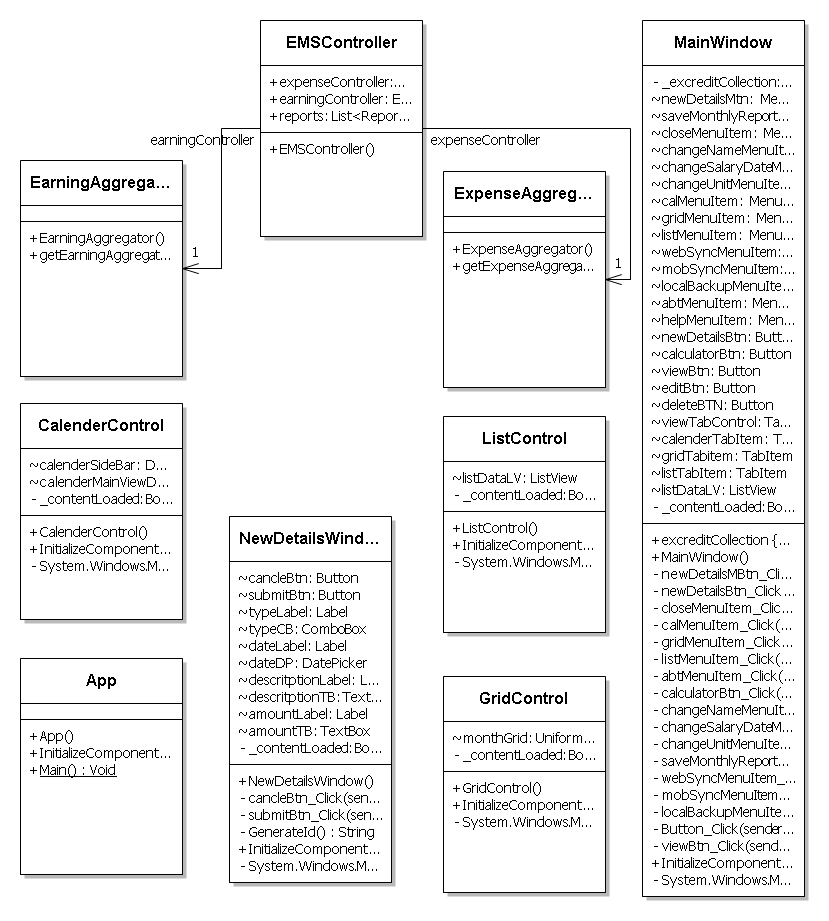
|  |  |
| --- | --- |
| **Entities** | **Attributes** |
| Expenses | **Expense Id,** Amount, Text |
| Earnings | **Earning Id,** Amount, Text |
| Monthly Calculation | **Month-Year**, Total Expense, Total Earning |
| Day | **Date**, Total Expense, Total Earning |
| User | **User Id**, Name, password, Backup Path, Preferences. |

**Relationship between Entities:**

* Monthly Calculation has Day 🡪 1 : N
* In a **Day** happens **Expenses**🡪 1 : N
* In a **Day** happens **Earnings**🡪 1 : N
* **User** does **Expenses** 🡪 M : N
* **User** does **Earnings** 🡪 M : N



## Class Diagrams



## Activity Diagrams

**User Login**

**User**

**Controller**

**Login**

**Enter**

**User name & password**

**Authorization**

**Update**

**Or**

**Check Status**

**Exit**

**Verify**

**Yes**

**No**

**Income Transaction**

**User**

**Controller**

**Enter**

**User name & password**

**Authorization**

**Exit**

**Income**

**Update Balance**

**Add**

**Income**

**Delete Income**

**Modify Income**

**Update Income**

**Income**

**Verify**

**Yes**

**No**

**Expense Transaction**

**Verify**

**Yes**

**No**

**User**

**Controller**

**Enter**

**User name & password**

**Authorization**

**Exit**

**Expenses**

**Update Balance**

**Add**

**Expenses**

**Delete**

**Expenses**

**Modify**

**Expenses**

**Update Expenses**

**Expenses**

**Verify**

**Yes**

**No**

**View Report**

**Verify**

**Yes**

**No**

**Verify**

**Yes**

**No**

**User**

**Controller**

**Enter**

**User name & password**

**Authorization**

**Exit**

**Report**

**View Report**

**Daily**

**Monthly**

**Weekly**

**Display**

**Report**

**Period**

**Verification**

**Yes**

**No**

**Sync**

**Verify**

**Yes**

**No**

**Verify**

**Yes**

**No**

**User**

**Controller**

**Enter**

**User name & password**

**Authorization**

**Exit**

**Sync**

**Sync**

**Desktop**

**Application**

**Mobile Application**

**Web Application**

**Sync Completed**

**Sync**

**Verification**

**Yes**

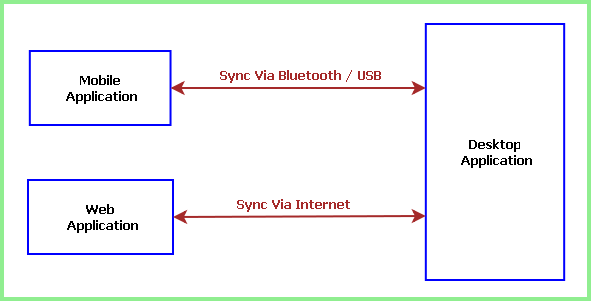
**No**

# System Design

## Modularisation details

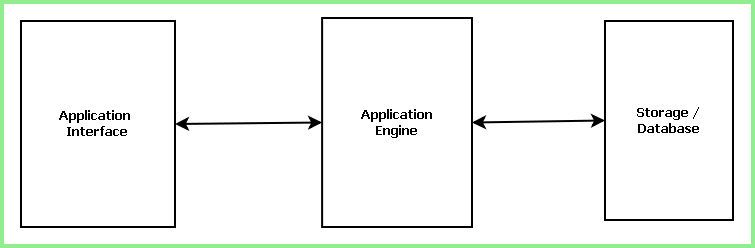
Expense manager software consists of three different applications:

* + Desktop Application
  + Mobile Application
  + Web Application



Each application is internally divided into three main modules such as:

* Application Interface
* Application Engine
* Storage / Database



Desktop Application is the full featured application which contains the Permanent storage or bigger database where as Mobile and Web application has a small & temporary storage. People can note down their expenses while roaming, at their mobile using expense manager. They can later sync and take the backup of their expenses and get a final report. They can sync with the web interface and store the data in the online database from where their important data would never be lost. Thus using three different interface and database data will not only be easy to maintain but also secure forever.

**Desktop Application Module:** It consists of three main parts, namely the GUI module, Engine/controller module and storage module.

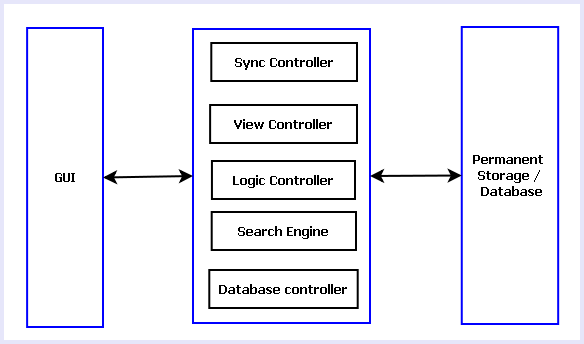


Fig: Modules of Desktop Application

**Desktop Application GUI:** The GUI for the desktop application will be designed using WPF (Windows Presentation Foundation) and XAML (Extensive Advanced Markup Language). The GUI will have several views like Calendar view, List view, Grid View. It will have options foe adding new expenses and earnings, searching for expenses/earnings, adding remainder for future expenses, syncing with mobile/web application.

**Desktop Application Engine:** Desktop Application Engine is the heart of the application. It controls the GUI interactions, logical calculations and database queries. It consists of 5 sub modules, such as:

**View Controller:**

It controls the look and feel of GUI. As mentioned earlier, the GUI will have three different views: List View, calendar View and Grid View.

List view will display data as a list with columns for date, tag text and amount. List view can be sorted by the columns.

In calendar view, user can select any date and data associated with the date will be displayed. Calendar view has three varieties like daily, weekly and monthly views.

Grid view will display the available data in contiguous grids. Each grid will display tag text, amount and date. The components of view controller are shown in the diagram below.

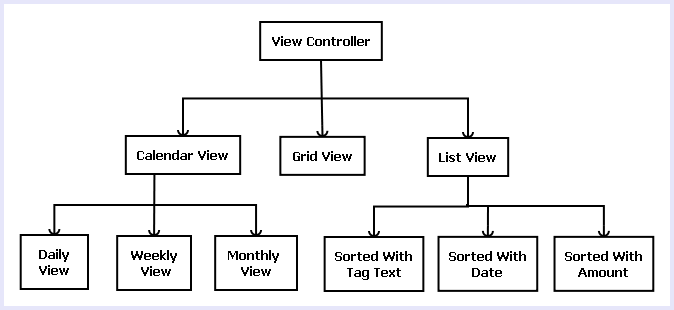


Fig: Various parts of View Controller

**Sync Controller:**

Sync controller handles the synchronization of data with mobile and web application. Sync controller receives data from other applications, processes data and saves data for future use.

**Logic Controller:**

Logic controller manages all the modules of application engine. It handles the interaction between other modules. The instances of all other module are created in logic controller so that it can control them. The application logics are written in this module.

**Search Engine:**

Search engine helps the user to search available data. It will have options for searching by tag text, amount and date. Search engine will form a query depending on the user input and fetch the result from database.

**Database Controller:**

Database controller handles the database interaction. It takes care of database addition, modification, deleting and retrieval of data from storage/ database.

**Desktop Application Storage:** Desktop application storage is the permanent storage/ database of the expense manager software. This module will be implemented using MySQL. The module sill store all the data related to this application. Users will be able to add, modify, erase and fetch/ view data on runtime.

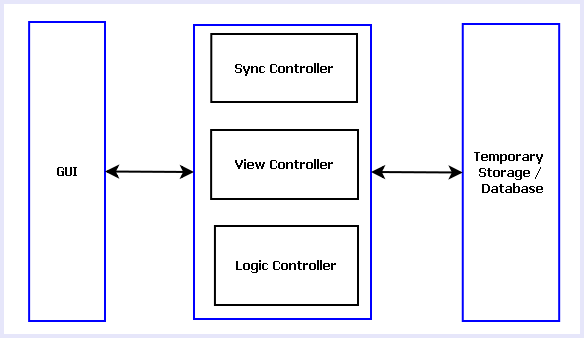


Fig: Modules of Mobile & Web Application

## Data integrity and constraints

### Entity integrity

### Referential Integrity

### Domain Integrity

### User Defined Integrity

## Database design

The database used for this software is called **Dnbdb**. Database tables and corresponding keys are shown in tabular form. It shows the tables and its columns. A key in **Bold** is the primary key.

Screenshots of table structures:

Table: user



Table: contact

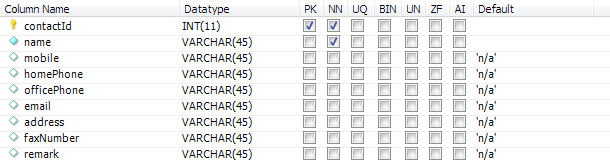


Table: Note



Table: password

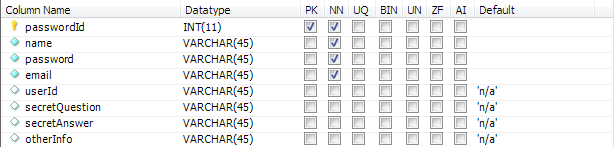
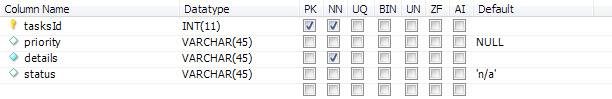


Table: tasks



|  |  |
| --- | --- |
| **Tables** | **Keys** |
| Note | **Note Id**, Content, Time, Size, Web Service id, user |
| RSS Feed | **RSS Feed Id** , Web Service Id, Content, time, size |
| Daily notebook & Social Networking Updater | **Sw Id**, Web Services Supported, Users, Size |
| User | **User Id**, Name, Social Network Data, Preferences. |
| Web Service | **Web Service Id,** Authentication Data, Feed Data, Preferences. |

## User Interface Design

Screen shot

## Test Cases (Unit Test Cases and System Test Cases)

# Coding

## Complete Project Coding

Code

## Comments and Description of Coding segments

Various types of comments and description we use in our coding section. Some of them are:

//open the connection

This comment is use at the data interaction section where the code to open the MySql connection.

//define the command reference

To define a command reference in MySql.

//define the connection used by the command object

To define the connection, which is used by the comment object.

//always close the connection

It is indicating to close connection after code is executed.

Manu Unused code in our project we did comment them also like :  
<!--<Condition Property="Password" Value="c" />-->

## Standardization of the coding

12

## Code Efficiency

We started working on the project keeping in mind that we must develop it in a way that it not only provides a very easy to use GUI but also provide a fast and flexible service to the users. We know that a particular work can be done in more than one ways. We have tried all the options and then chose the one which provides the fastest and most secure performance. First of all, we have used the latest technologies of Microsoft like visual studio 2010 as IDE and WPF as GUI to keep our application’s performance few steps ahead. We have studies all the rules of software development life cycle and applied them to keep our application flexible. We have given special attention to the storage related codes. We have avoided all the unnecessary database codes and kept them as short as possible without harming our purpose so that insertion, updating, deletion and fetching of data take place flexibly. You can see the result as a user; our application does all the works very smoothly.

## Error handling

## Parameters calling/passing

## Validation checks

# Testing

## Testing techniques and Testing strategies used

EXPENSE MANAGER application will be tested using following strategies to ensure that the application succeeds to complete all the functional and non functional requirements:

### Database & Data Integrity Testing

The databases and the database processes should be tested as a subsystem within the EXPENSE MANAGER Application. These subsystems should be tested with the target-of-test’s User Interface as the interface to the database.

|  |  |
| --- | --- |
| Test Objective: | Ensure that data is stored correctly, audits can be performed, access is controlled |
| Technique: | * SQL queries will be executed in the DB to verify the data content and correctness. |
| Completion Criteria: | * All planned tests have been executed. * All defects that have been identified have been resolved * All resolutions have been implemented. |

### Functional Testing:

Function testing focuses on any requirements for test that can be traced directly to use cases or business functions and business rules. The goals of these tests are to verify proper data acceptance, processing, and retrieval, and the appropriate implementation of the business rules. This type of testing is based upon black box techniques; that are verifying the application and its internal processes by interacting with the application via the Graphical User Interface (GUI) and analyzing the output or results. Identified below is an outline of the function testing recommended for EXPENSE MANAGER:

|  |  |
| --- | --- |
| Test Objective: | Ensure proper target-of-test functionality, including business process validation. |
| Technique: | Execute each use case, use-case flow, or function, using valid and invalid data, to verify the following:   * The expected results occur when valid data is used. * The appropriate error or warning messages are displayed when invalid data is used. * Business rules are properly applied. * Black Box end to end testing of configured processes. Manual validation of required and optional fields. |
| Completion Criteria: | * All planned tests have been executed. * All defects that have been identified have been resolved * All resolutions have been implemented. |

### Regression Testing:

Regression testing focuses on software functionality that may have been previously working however through subsequent changes may have been inadvertently impacted. The goals of these tests are to verify that the broader impact of changes has been verified. Identified below is an outline of the regression testing recommended for each application(s)/module(s) of EXPENSE MANAGER.

|  |  |
| --- | --- |
| Test Objective: | Ensure that previously passed test cases continue to pass as the new system development is deployed and that surrounding systems that may be impacted by a change are still functioning as expected. |
| Technique: | * Execute previous passed testing suites to ensure the following: * The expected results occur when valid data is used. * The appropriate error or warning messages are displayed when invalid data is used. * Each business rule is properly applied. |
| Completion Criteria: | • All planned regression tests have been executed.  • All identified defects have been resolved. |

### User Interface Testing:

User Interface (UI) testing verifies a user’s interaction with the software. The goal of UI testing is to ensure that the User Interface provides the user with the appropriate access and navigation through the functions of the target-of-test. In addition, UI testing ensures that the objects within the UI function as expected and conform to corporate or industry standards. Most of this testing will have been done during functional testing. The areas of focus will be on design, layout and navigation of the screens.

|  |  |
| --- | --- |
| Test Objective: | UI testing will verify the screens and the layouts and navigation |
| Technique: | * Verify the design and layout of the screen. * Identify the integration links. * Test the functioning of the links – that the proper page is displayed and correct messages, pop-ups are shown when they need to be displayed etc * Validation of general navigation |
| Completion Criteria: | * All navigation test cases have been executed. * All screens have been verified as per design and layouts * All defects that have been identified have been resolved. |

### Performance Profiling:

Performance profiling is a performance test in which response times, transaction rates, and other time-sensitive requirements are measured and evaluated. The goal of Performance Profiling is to verify performance requirements have been achieved. Performance profiling is implemented and executed to profile and tune performance behaviours as a function of conditions such as workload or hardware configurations

|  |  |
| --- | --- |
| Test Objective: | The purpose of performance profiling is to ensure the performance of the EXPENSE MANAGER application is up to the desired level. |
| Technique: | * Use a subset of Test Procedures developed for Function and Business Cycle Testing. * Modify data files to increase the number of transactions or the scripts to increase the number of iterations each transaction occurs. * This will be done by using Load Runner or Quick Test Professional (QTP). |
| Completion Criteria: | * Single Transaction or single user: Successful completion of the test scripts without any failures and within the expected or required time allocation per transaction. * Results are recorded and a performance baseline is created for the major logical functions within the scenarios listed above. * All performance defects are reviewed and triaged to an acceptable resolution. |

.

### Load Testing:

Load testing is a performance test which subjects the target-of-test to varying workloads to measure and evaluate the performance behaviours and ability of the target-of-test to continue to function properly under these different workloads. The goal of load testing is to determine and ensure that the system functions properly at the expected maximum workload. Additionally, load testing evaluates the performance characteristics, such as response times, transaction rates, and other time sensitive issues.

|  |  |
| --- | --- |
| Test Objective: | The purpose of load testing is to verify performance behaviour time for designated transactions or business cases under varying workload conditions. |
| Technique: | * Use a subset of Test Procedures developed for Function and Business Cycle Testing. * Scripts will be executed to simulate the peak load for 1 hour and report will be generated and analysed. * This will be done using Load Runner. |
| Completion Criteria: | * Multiple transactions or multiple users: Successful completion of the test scripts without any failures and within acceptable time allocation. * Results are recorded and a performance baseline is created for the major business functions within the scenarios listed above. * All load testing defects are reviewed and triaged to an acceptable resolution. |

### Stress Testing:

Stress testing is a type of performance test implemented and executed to find errors due to low resources or competition for resources. Low memory or disk space may reveal defects in the target-of-test that aren't apparent under normal conditions. Other defects might result from competition for shared resources like database locks or network bandwidth. Stress testing can also be used to identify the peak workload the target-of-test can handle, which is often beyond the production workload.

### Volume Testing:

Volume Testing subjects the target-of-test to large amounts of data to determine if limits are reached that cause the software to fail. Volume Testing also identifies the continuous maximum load or volume the target-of-test can handle for a given period. For example, if the target-of-test is processing a set of database records to generate a report, a Volume Test would use a large test database and check that the software behaved normally and produced the correct report.

### Security & Access Control Testing:

Security and Access Control Testing focus on following key areas of security:

* Application-level security, including access to the Data or Business Functions

Application-level security ensures the authentication and authorization of a user. Authentication ensures that the user is a valid user of the system and authorization ensures that the user has the proper privileges to perform specific tasks on desired resources of the system. Testing will be conducted to validate the rules by taking into considerations the various roles applicable for the system.

### Failover & Recovery Testing:

Failover and Recovery Testing ensures that the target-of-test can successfully failover and recover from a variety of hardware, software or network malfunctions with undue loss of data or data integrity.

Failover testing ensures that, for those systems that must be kept running, when a failover condition occurs, the alternate or backup systems properly “take over” for the failed system without loss of data or transactions.

Recovery testing is an antagonistic test process in which the application or system is exposed to extreme conditions, or simulated conditions, to cause a failure, such as device Input/ Output (I/O) failures or invalid database pointers and keys. Recovery processes are invoked and the application or system is monitored and inspected to verify proper application, or system, and data recovery has been achieved.

### Configuration Testing:

Configuration testing verifies the operation of the target-of-test on different software and hardware configurations. In most production environments, the particular hardware specifications for the client workstations, network connections and database servers vary. Client workstations may have different software loaded⎯for example, applications, drivers, and so on⎯and at any one time, many different combinations may be active using different resources.

### Installation/Deploy & Back out Testing:

Installation testing has two purposes. The first is to ensure that the software can be installed under different conditions⎯such as a new installation, an upgrade and a complete or custom installation⎯under normal and abnormal conditions. Abnormal conditions include insufficient disk space, lack of privilege to create directories, and so on. The second purpose is to verify that, once installed; the software operates correctly and can be backed out successfully. This usually means running a number of the tests that were developed for Function testing before and after the back out.

### Post Production Testing:

The purpose of Post production testing is to verify that, once deployed, the software operates correctly. This usually means running a number of the tests that were developed for Function Testing ensuring that no data is changed/ modified in production. Typically, the business SME’s assist with Post production testing.

### Unit Testing:

Unit testing will take place within the construction phase of the project. After application module has been built to meet design specifications, each component (screen, view, interface, conversion program, etc.) will be tested individually to help confirm that it functions properly as an individual unit. Basic performance testing will also be completed during unit test to resolve obvious issues with performance prior to the System Testing.

The resource responsible for development will conduct testing of their module using the unit test conditions defined by the developer based on detailed design documents. The final step of unit test will be a review by the team lead to obtain their signoff on the component test checklist.

### Smoke Testing:

|  |  |
| --- | --- |
| Test Objective: | Verifies the major functionality at high level in order to determine if further testing is possible. |
| Technique: | * After initial deployment to the test environment validate all critical components of the application prior to proceeding with testing. |
| Completion Criteria: | * Navigation through the application at high level is possible, testing can continue. |

### Data Migration Testing:

This is the process of testing to verify whether or not the data migration (or conversion) has been successfully completed. The testing process will be carried out by running SQL scripts on both the source and destination databases.

The fields which are present in the new data Model in the Destination DB(s) will be migrated from the existing systems source DB(s) to Destination DB(s).

|  |  |
| --- | --- |
| Test Objective: | The objective of this test is to verify that data migration is successful from source DB(s) to destination DB(s). |
| Technique: | * The Team is notified before the data migration. * Team runs queries on the source DB and fetches the data. * Data Migration is done. * Mapped data needs to be determined. * Team runs the queries on the Destination DB and fetches the data. * Cross verification of the data is done to see that data fetched from the old database is same as the data fetched from the new database. * Verification of the table structure. * Verification of record counts. * Verification of the data formatting. |
| Completion Criteria: | * Data fetched from the Source DB(s) and Destination DB(s) matches. * The record count in the Source and the Destination databases should be equal. * No data are truncated. * Data formatting is proper (if required at any instance). * All defects that have been identified have been resolved. |

## Testing Plan used

## Test reports for Unit Test Cases and System Test Cases

## Debugging and Code improvement:

# System Security measures:

## Database/data security:

* + This software requires a valid password to login and then it allows using any of its features.
  + The login password will be saved in encrypted format in database.
  + This software will use Google open-id authentication for web interface.
  + A backup and restore feature has been used in case of loss of data due to database crash and other problems.

## Creation of User profiles and access rights

* A user first must create a new account to use this software.
* A predefined password will be present for a first time user and he/she must immediately change that predefined password and add his own to make his data completely secure.

# Cost Estimation of the Project along with Cost Estimation Model

## Estimation of development effort

## Estimation of development time

# Reports (sample layouts should be placed)

List of reports that are likely to be generated in this software are given below:

* List of Expenses can be generated
* List of Earnings can be generated
* Daily report can be generated
* Weekly report can be generated
* Monthly report can be generated
* Yearly report can be generated

# Future scope and further enhancement of the Project

* + To support UNIX / Linux based operating systems.
  + To Support Mobile operating systems for Windows Mobile, Nokia, Blackberry.
  + To port it on handheld device like iPad, Galaxy Tab & netbooks.
  + To enhance the web interface

# Bibliography

# Appendices

## IDE (Visual Studio 2010):

Microsoft Visual Studio is a powerful IDE that ensures quality code throughout the entire application lifecycle, from design to deployment. Whether we are developing applications for SharePoint, the web, Windows, Windows Phone, and beyond, Visual Studio is the ultimate all-in-one solution. Visual Studio includes a [code editor](http://en.wikipedia.org/wiki/Code_editor) supporting [IntelliSense](http://en.wikipedia.org/wiki/IntelliSense) as well as [code refactoring](http://en.wikipedia.org/wiki/Code_refactoring). The integrated [debugger](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building [GUI](http://en.wikipedia.org/wiki/GUI) applications, web designer, [class](http://en.wikipedia.org/wiki/Class_(computing)) designer, and [database schema](http://en.wikipedia.org/wiki/Database_schema) designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for [source-control](http://en.wikipedia.org/wiki/Source_control) systems (like [Subversion](http://en.wikipedia.org/wiki/Subversion_(software)) and [Visual SourceSafe](http://en.wikipedia.org/wiki/Visual_SourceSafe)) and adding new toolsets like editors and visual designers for [domain-specific languages](http://en.wikipedia.org/wiki/Domain-specific_language) or toolsets for other aspects of the [software development lifecycle](http://en.wikipedia.org/wiki/Software_development_lifecycle) (like the [Team Foundation Server](http://en.wikipedia.org/wiki/Team_Foundation_Server) client: Team Explorer).

## Front End - WPF (Windows Presentation Framework)

Windows Presentation Foundation (WPF) provides developers with a unified programming model for building rich Windows smart client user experiences that incorporate UI, media, and documents. Windows Presentation Foundation (WPF) is a next-generation presentation system for building Windows client applications with visually stunning user experiences. With WPF, you can create a wide range of both standalone and browser-hosted applications. The core of WPF is a resolution-independent and vector-based rendering engine that is built to take advantage of modern graphics hardware. WPF extends the core with a comprehensive set of application-development features that include Extensible Application Markup Language (XAML), controls, data binding, layout, 2-D and 3-D graphics, animation, styles, templates, documents, media, text, and typography. WPF is included in the Microsoft .NET Framework, so you can build applications that incorporate other elements of the .NET Framework class library.

## Extensible application Markup Language (XaML)

XAML stands for Extensible Application Markup Language. Its a simple language based on XML to create and initialize .NET objects with hierarchical relations. Altough it was originally invented for WPF it can by used to create any kind of object trees.

Today XAML is used to create user interfaces in WPF, Silverlight, declare workflows in WF and for electronic paper in the XPS standard.

All classes in WPF have parameter less constructors and make excessive usage of properties. That is done to make it perfectly fit for XML languages like XAML.

All you can do in XAML can also be done in code. XAML ist just another way to create and initialize objects. You can use WPF without using XAML. It's up to you if you want to declare it in XAML or write it in code. Declare your UI in XAML has some advantages:

* XAML code is short and clear to read
* Separation of designer code and logic
* Graphical design tools like Expression Blend require XAML as source.
* The separation of XAML and UI logic allows it to clearly separate the roles of designer and developer.

## Programming Framework (.NET 4)

The .NET 4 Framework is Microsoft's platform for building applications that have visually stunning user experiences, seamless and secure communication, and the ability to model a range of business processes. The .Net Framework consists of:

Common Language Runtime – provides an abstraction layer over the operating system

Base Class Libraries – pre-built code for common low-level programming tasks

Development frameworks and technologies – reusable, customizable solutions for larger programming tasks.

The framework's Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. The class library is used by programmers, who combine it with their own code to produce applications.

## Database/backend - MySQL

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout its history.

The MySQL Community Edition includes:

* Pluggable Storage Engine Architecture
* Multiple Storage Engines: InnoDB , MyISAM, NDB (MySQL Cluster),Memory ,Merge , Archive, CSV
* MySQL Replication to improve application performance and scalability
* MySQL Partitioning to improve performance and management of large database applications
* Stored Procedures to improve developer productivity

## ide for Database –MySQL workbench

MySQL Workbench is a visual database design tool that integrates SQL development,administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the successor to DBDesigner 4 from fabFORCE.net, and replaces the previous package of software,MySQL GUI Tools Bundle.

## Programming Language (C#)

C# is a type-safe, object-oriented language that is simple yet powerful, allowing programmers to build a breadth of applications. C# is a [multi-paradigm programming language](http://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [imperative](http://en.wikipedia.org/wiki/Imperative_programming), [declarative](http://en.wikipedia.org/wiki/Declarative_programming), [functional](http://en.wikipedia.org/wiki/Functional_programming), [generic](http://en.wikipedia.org/wiki/Generic_programming), [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming)([class-based](http://en.wikipedia.org/wiki/Class_(computer_science))), and [component-oriented](http://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. It was developed by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) within the [.NET](http://en.wikipedia.org/wiki/.NET_Framework) initiative and later approved as a standard by [Ecma](http://en.wikipedia.org/wiki/Ecma_International) (ECMA-334) and [ISO](http://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO/IEC 23270). C# is one of the programming languages designed for the [Common Language Infrastructure](http://en.wikipedia.org/wiki/Common_Language_Infrastructure).

C# is intended to be a simple, modern, general-purpose, object-oriented programming language.

## Dia for Diagram Drawing & Modeling

Dia is free and open source general-purpose diagramming software, developed as part of the GNOME project's office suite and was originally created by Alexander Larsson. Dia uses a controlled single document interface (CSDI) similar to GIMP and Sodipodi.

Dia has a modular design with several shape packages available for different needs: flowchart, network diagrams, circuit diagrams, and more. It does not restrict symbols and connectors from various categories from being placed together.

Dia is a gtk+ based diagram creation program released under the GPL license.

Dia is inspired by the commercial Windows program 'Visio', though more geared towards informal diagrams for casual use. It can be used to draw many different kinds of diagrams. It currently has special objects to help draw entity relationship diagrams, UML diagrams, flowcharts, network diagrams, and many other diagrams. It is also possible to add support for new shapes by writing simple XML files, using a subset of SVG to draw the shape.

It can load and save diagrams to a custom XML format (gzipped by default, to save space), can export diagrams to a number of formats, including EPS, SVG, XFIG, WMF and PNG, and can print diagrams (including ones that span multiple pages).

## Google Spreadsheet Interface:

*With Google Spreadsheets, we can easily create, share, and edit spreadsheets online. Here are a few specific things we can do:*

* *Import and export these file types: .xls, .csv, .txt and .ods. We can also export data to a PDF or an HTML file.*
* *Format cells and edit formulas so we can calculate results and make data look the way we want it.*
* *Chat in real time with others who are editing our spreadsheet.*
* *Embed a spreadsheet, or a section of a spreadsheet, in our blog or website.*

## Windows Mobile 6 Professional SDK

Windows Mobile is a mobile operating system developed by Microsoft that was used in smartphones and mobile devices. This features a suite of basic applications developed with the Microsoft Windows API. It is designed to be somewhat similar to desktop versions of Windows, feature-wise and aesthetically. Additionally, third-party software development is available for Windows Mobile, and software applications can be purchased via the Windows Marketplace for Mobile. Windows Mobile 6.1 was announced April 1, 2008. It is a minor upgrade to the existing Windows Mobile 6 platform which brings with it various performance enhancements, a redesigned Home screen featuring horizontal tiles that expand on clicking to display more information, although this new home screen is featured only on Windows Mobile Standard edition. This feature was inexplicably left out of the Professional edition. Several other changes such as threaded SMS, full page zooming in Internet Explorer and 'Domain Enroll' have also been added, along with a "mobile" version of the Microsoft OneNote program and an interactive "Getting Started" wizard. Windows Mobile 6.1 also featured improved bandwidth efficiency in its push-email protocol "Activesync" of "up to 40%", this reduced data usage was the cause of considerably improved battery life in many devices.

# Glossary.