Expenses Management System

BY: Suveksha khanal



**Analysis Phase**

# **Requirements:**

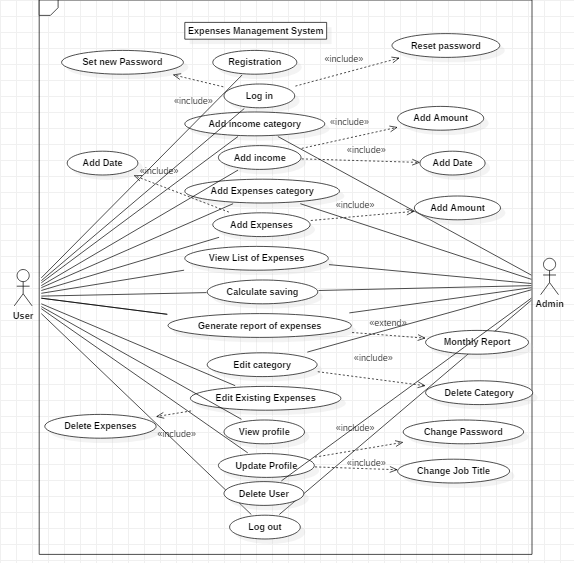
Functional requirement (F) are the main things that user expects from the software.

Non-function requirement (NF) are things that are not straight forward requirement of the system rather related to usability.

|  |  |  |
| --- | --- | --- |
| Functional/non-functional | Requirements | MoSCoW Prioritization |
| F(R1) | Admin Login | M |
| F(R2) | User login | M |
| F(R3) | Add Expenses | M |
| F(R4) | Add income | M |
| NF(R5) | View Profile | S |
| NF(R6) | Update Profile | S |
| F(R7) | Add Expenses Category | M |
| F(R8) | Add Income Category | M |
| NF(R9) | Edit Expenses Category | S |
| NF(R10) | Edit Income Category | S |
| F(R11) | Delete Expense Category | S |
| F(R12) | Delete income category | S |
| NF(R13) | Change Password | C |
| NF(R14) | View List of Expenses | S |
| F(R15) | Edit Existing Expenses | M |
| F(R16) | Edit Existing Income | M |
| F(R17) | Log out | M |
| F(R18) | Calculate Saving | M |
| NF(R19) | Report of Expenses | C |

# **Use Case Diagram:**

Use case is a graphical interpretation of the interaction among the elements of a system. it is used to identify, clarify and organize system requirement. Use case diagram shows relationship between and among the actors and the use cases.



**Justification**

I have used use case diagram so that it helps us design a system from user’s perspective. It is a useful technique for communicating system behaviour in the user's terms. Use case provides observable and valuable result to the actors of the system. it helps us to understand the requirement of the system and what the system is supposed to do.

**Advantage**

* It helps to gather functional requirements of a system
* It is easily understandable
* It gathers additional behaviour that can improve system robustness.

**Disadvantage**

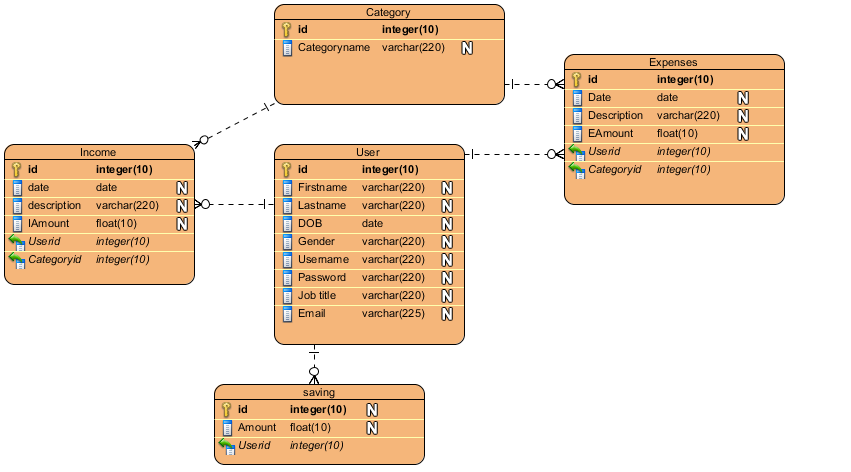
* Use case only summarises some of the relationships between use cases, actors and systems
* Use case does not show the order in which steps are performed
* Does not gather the non-functional requirements.

**Design Phase**

**Structural design**

**ER Diagram- Entity Relationship Diagram**

ER diagram is a part of a structural diagram used to design database. It has different symbols and connectors that are important. It shows the relationship of different entities and how they are related to each other in a system. these entities have attributes that define its properties.



**Justification**

This diagram shows relationship between attributes for Expenses Management System. It shows the different tables that will be appropriate for database and how they are related to each other. I have used Entity relationship diagram to provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization.

**Advantage:**

* It is very simple
* It gives a better visual representation
* It can be easily converted to any data model.

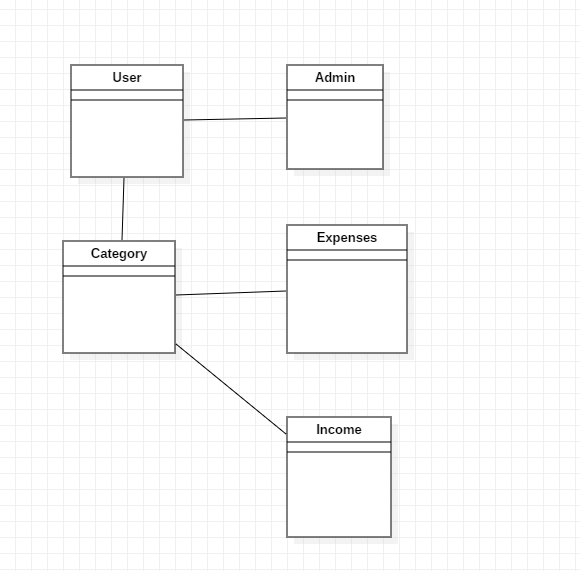
**Disadvantage:**

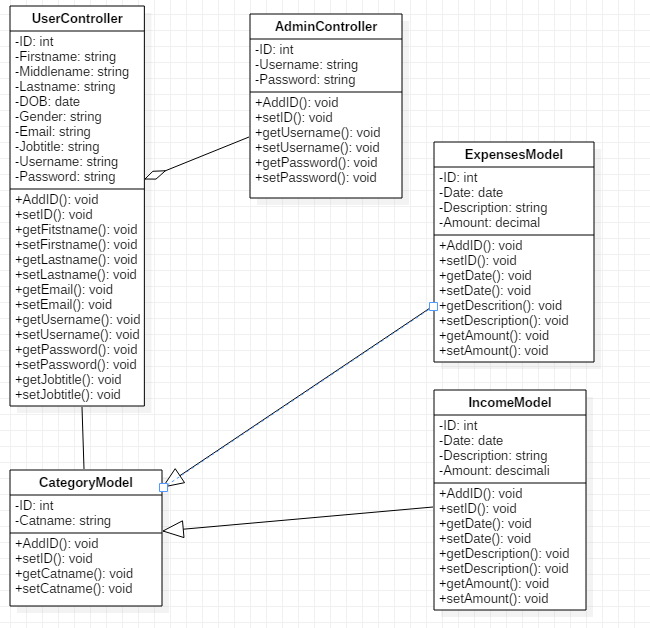
* There can be loss of information or content
* It can be difficult to show data manipulation
* There is limited relationship representation.

**Class Diagram**

Class diagram is also a part of structural diagram which describes the structure of a system. it shows the system’s classes, attributes, operations and relationships. Class diagram defines the method and variables in an object. It is useful in all forms of object-oriented programming.

**Initial Class Diagram:**



**Final Class Diagram:**

**Justification**

I have created a class diagram for expenses management system. it shows different classes and how they are related to each other. I have used class diagram to describe the structure of a system. class diagram helps to understand the system and plan accordingly.

**Advantage:**

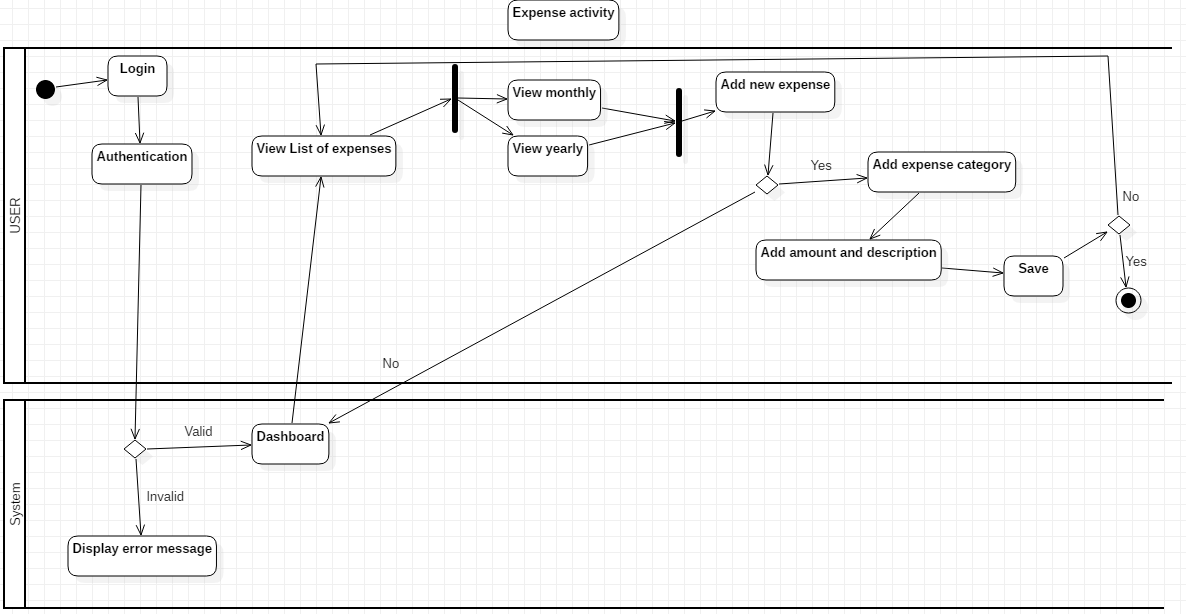
* Provides detail insight into the structure of the system
* They are simple and fast to read
* Reduces maintenance time.

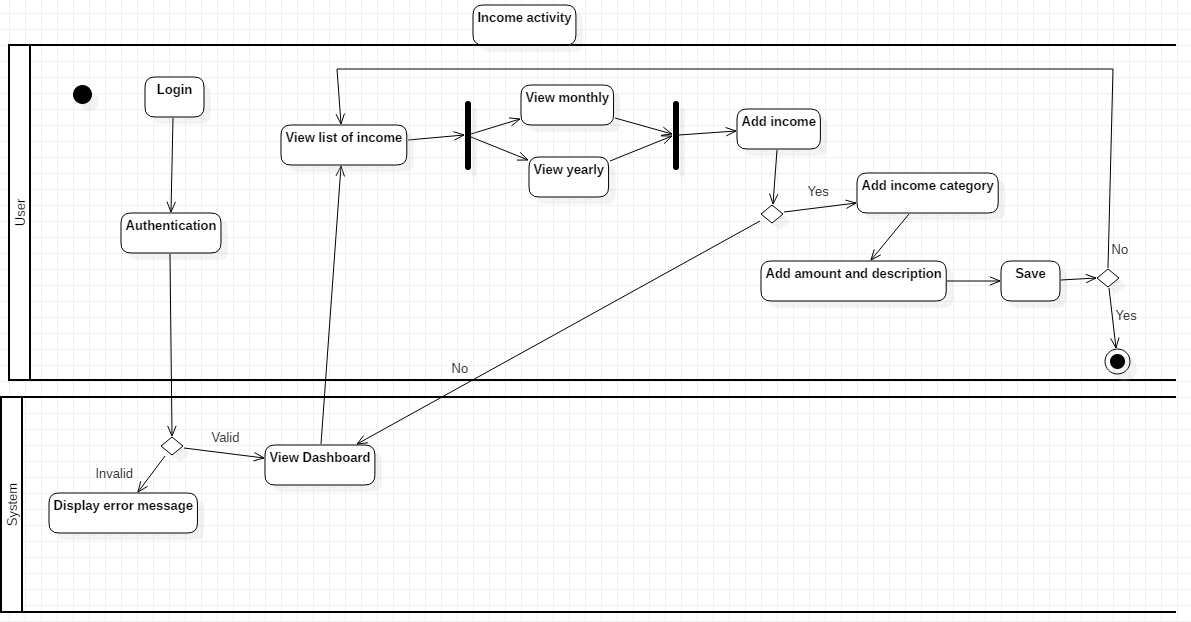
**Disadvantage:**

* If it is complicated, it may be difficult to correlate with the code
* They do not have any dynamics.

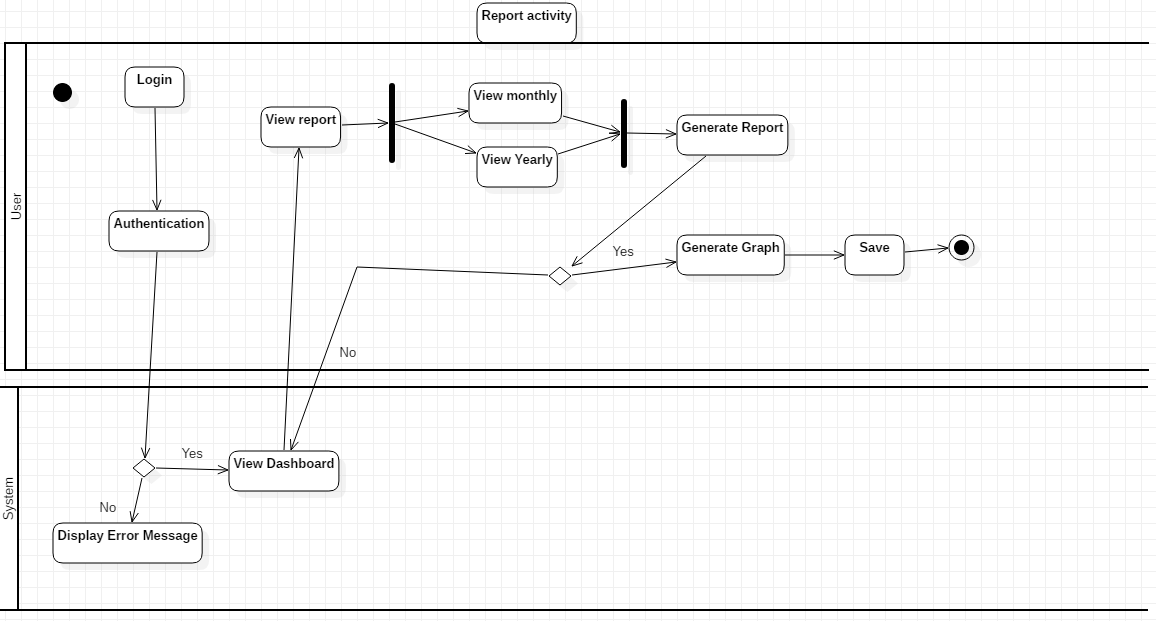
**Activity Diagram**

Activity diagram describes the dynamic aspects of the system. it represents the flow of the system, and show one activity to another activity. Activity diagram is sometimes considered as a flow of diagram. It also describes the parallel, branched and concurrent flow of the system.

**Expense Activity:**

**Income Activity:**

**Report Activity:**



**Justification**

I have used activity diagram to capture the dynamic behaviour of the system. It shows the flow of one activity to another activity. It helps to show clear understanding of the system. Activity diagrams are an essential part of the modelling process. I have used this diagram as they are also useful for analysing a use case by describing what actions need to take place and when they

**Advantage**

* It is simple to understand
* It is user-friendly
* It displays multiple conditions and actors within a work flow.
* It describes the steps performed in a UML use case.

**Disadvantage**

* It does not give detail about how object behave.
* Does not show any message flow from one activity to another.

**UI design:**

