**B.M.S. COLLEGE OF ENGINEERING BENGALURU**

Autonomous Institute, Affiliated to VTU



Lab Record

**Object Oriented Analysis and Design**

*Submitted in partial fulfillment for the 6th Semester Laboratory*

Bachelor of Technology

in

Computer Science and Engineering

*Submitted by:*

**TONY J MATHEW**

1BM17CS119

Department of Computer Science and Engineering

B.M.S. College of Engineering

Bull Temple Road, Basavanagudi, Bangalore 560 019

Jan-May 2019

**B.M.S. College of EngineerinG**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

****

***CERTIFICATE***

This is to certify that the Object-Oriented Analysis and Design(16CS6DCOOM) laboratory has been carried out by TONY J MATHEW (1BM17CS119) during the 6th Semester Jan-May-2019.

Signature of the Faculty Incharge:

NAME OF THE FACULTY:

Department of Computer Science and Engineering

B.M.S. College of Engineering, Bangalore

Table of Contents

|  |
| --- |
| 1.College Information System |
| 2.Hostel Management System |
| 3.Stock Maintenance System |
| 4.Coffee Vending Machine |
| 5.Online Shopping System |
| 6.Railway reservation system |
| 7.Graphics Editor |

**1. College Information System**

**Problem Statement:**

To design a College Information System to simplify the process of course registration. The system will allow students to register their courses online. The information is directly relayed to the college database and eliminates unnecessary paperwork, travelling and time consumption.

**SRS:**

**Purpose:**

The College Management System is an online system. The system is developed to facilitate more streamlined course registration process. The system allows students to register their courses online. The information is directly relayed to the college database and reduces paperwork, travelling and time consumption. Details of courses such as syllabus, credits and prescribed learning material can also be checked.

**Requirements:**

**Functional Requirements:**

* Provide Student Login:

Students can login and register for courses within prescribed time limits.

* Provide Course Information:

Course Details can be checked before and after registration. Data such as syllabus, credits and learning material are available for viewing.

* Provide Admin Login:

Admin login is required to update available courses as well as course details.

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

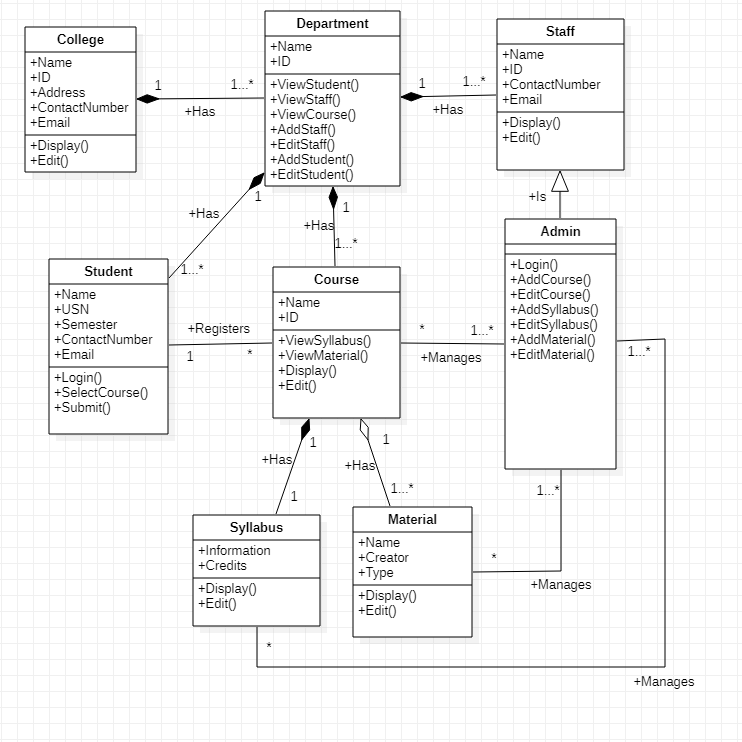
* Time:

The software product should be completed within deadlines

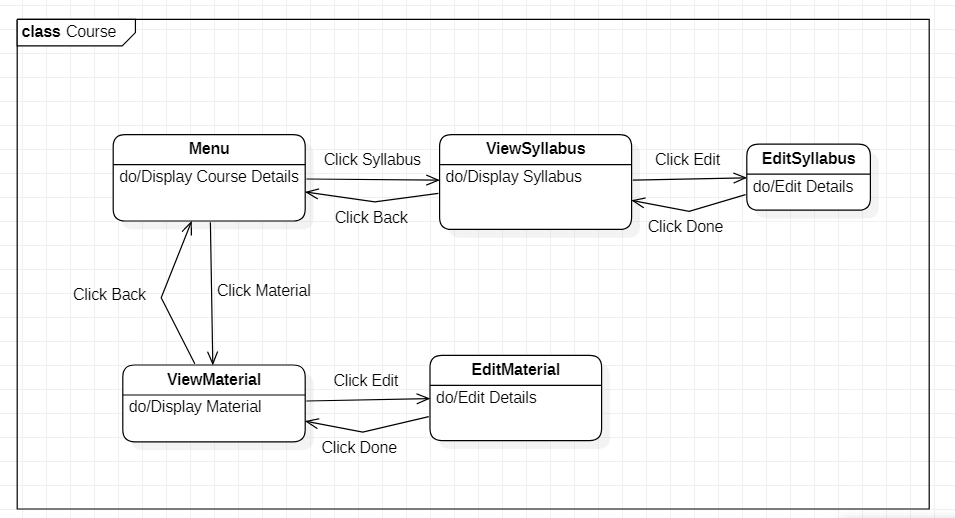
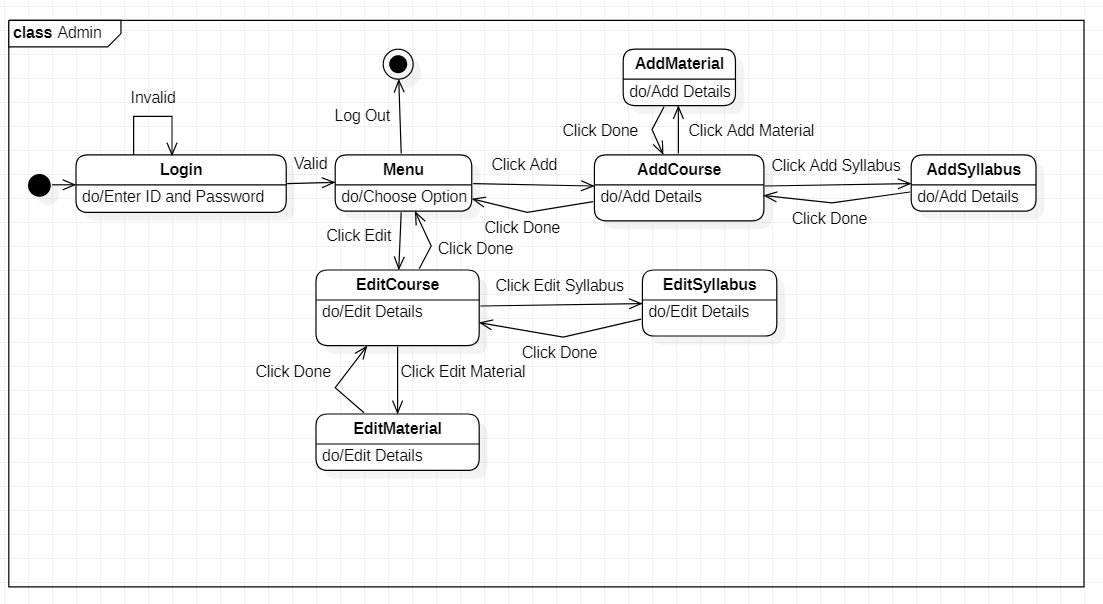
* Security:

The data should be secure

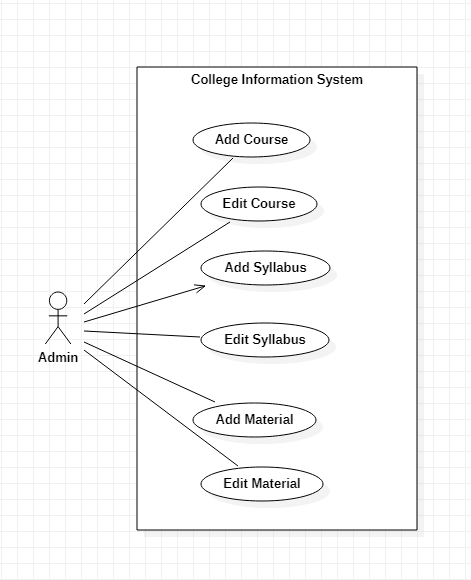
**Class Diagram:**



**State Diagram:**



**Use Case Diagrams:**

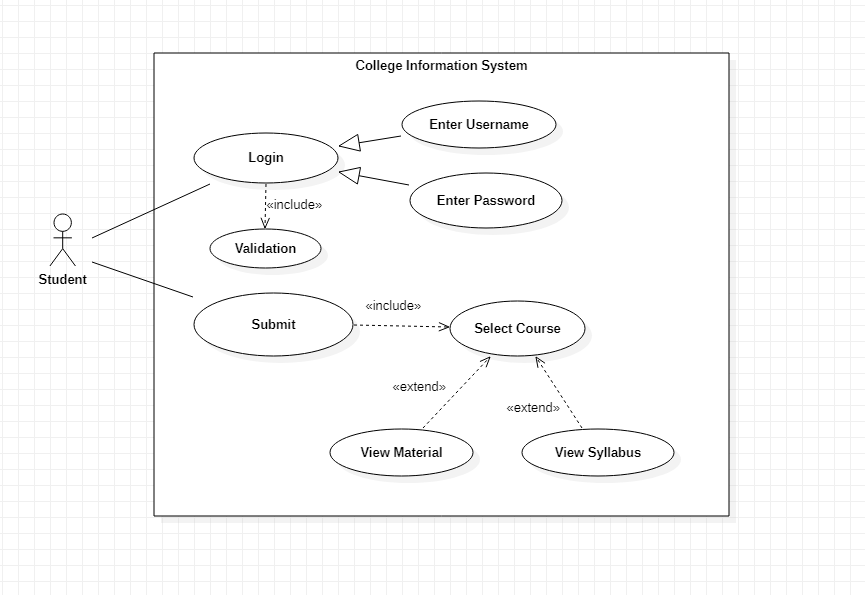


There is one actor:

* **Admin:** Person who manages the available courses

There are six use cases:

* **Add Course:** Add courses that can be chosen
* **Edit Course:** Edit courses that can be chosen
* **Add Syllabus:** Add syllabus to course
* **Edit Syllabus:** Edit syllabus of course
* **Add Material:** Add material to course
* **Edit Material:** Edit material of course



There is one actor:

* **Student:** Person who registers for courses

There are eight use cases:

* **Login:** Login to course registration website
* **Enter Username:** Enter username to login
* **Enter Password:** Enter password to login
* **Validation:** Validate login credentials and allow login
* **Submit:** Submit selected courses to college database
* **Select Course:** Select course from available courses
* **View Syllabus:** View syllabus of course
* **View Material:** View material of course

**2. Hostel Management System**

**Problem Statement:**

To design a Hostel Management System to manage various activities related to hostel services. The system will allow students as well as college administrators to interact with hostel services in a streamlined manner with user-friendly GUI. The information is directly stored in the college database and eliminates unnecessary paperwork, travelling and time consumption.

**SRS:**

**Purpose:**

The Hostel Management System is an online system. The system is developed to facilitate more streamlined hostel management process. The system allows students to request a room and pay hostel fees online. Administrators can get information regarding students staying at hostel as well as collect fees, set canteen menu and security patrol routes. The information is directly stored in the college database and reduces paperwork, travelling and time consumption.

**Requirements:**

**Functional Requirements:**

* Allow Student to request hostel room:

Students can login and request hostel room.

* Provide Option to Pay Fees:

Hostel fees can be payed online through system.

* Provide Admin Login:

Admin login is required to approve student requests as well as manage hostel services such as canteen menu and security patrol routs

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

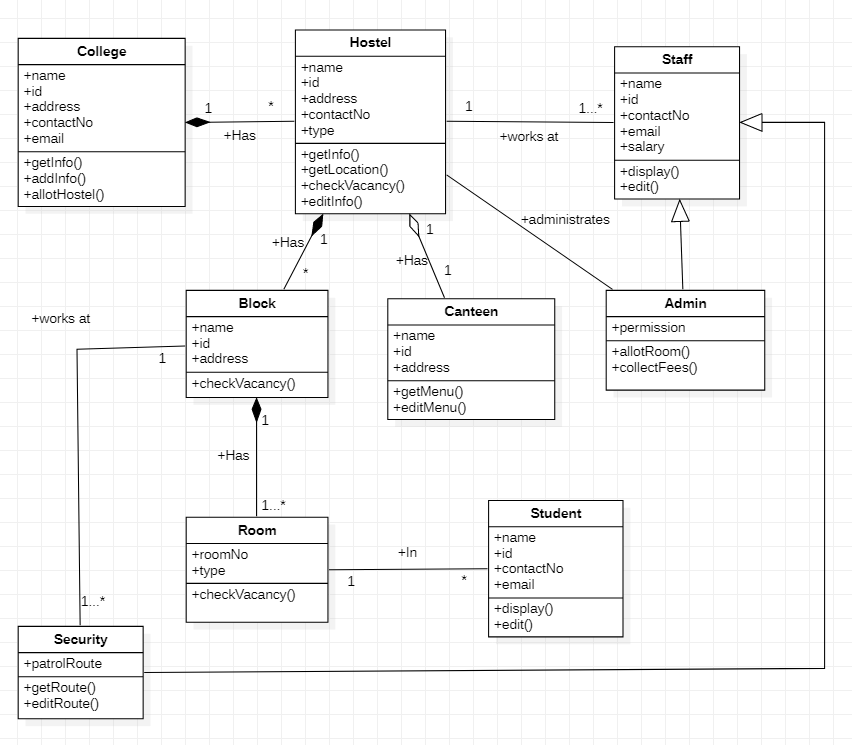
* Time:

The software product should be completed within deadlines

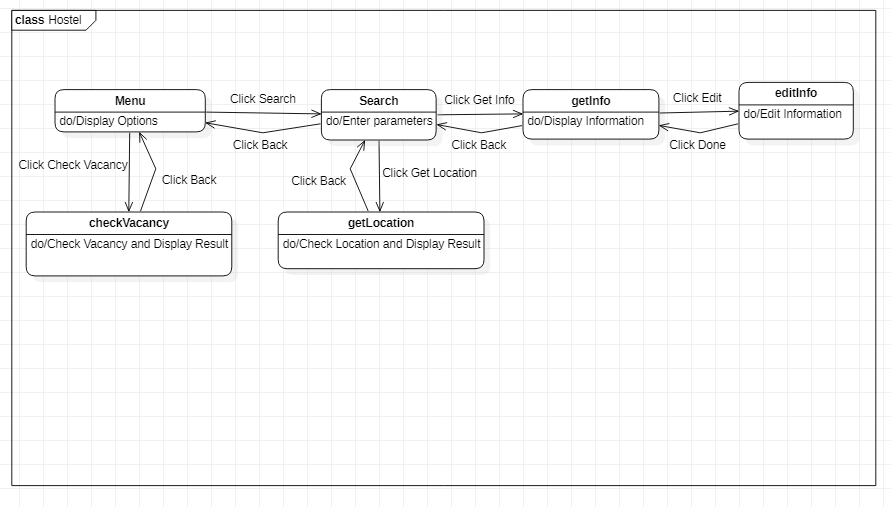
* Security:

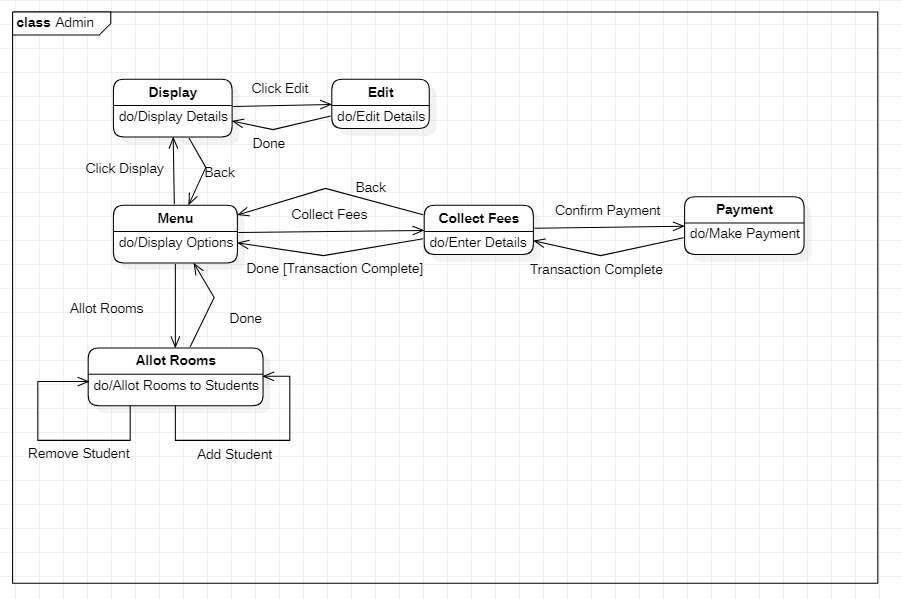
The data should be secure

**Class Diagram:**

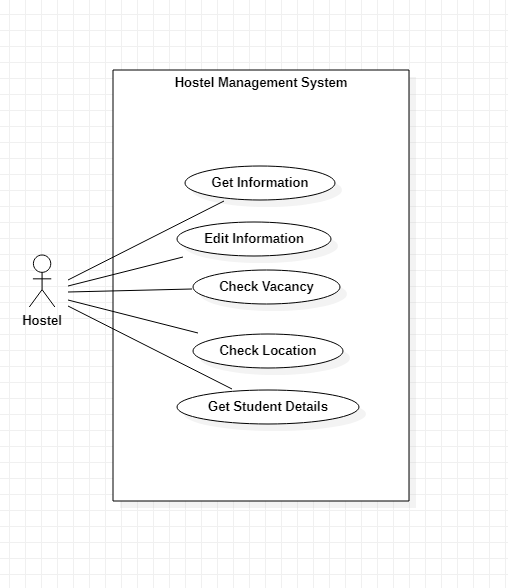
****

**State Diagram:**

****

****

**Use Case Diagrams:**

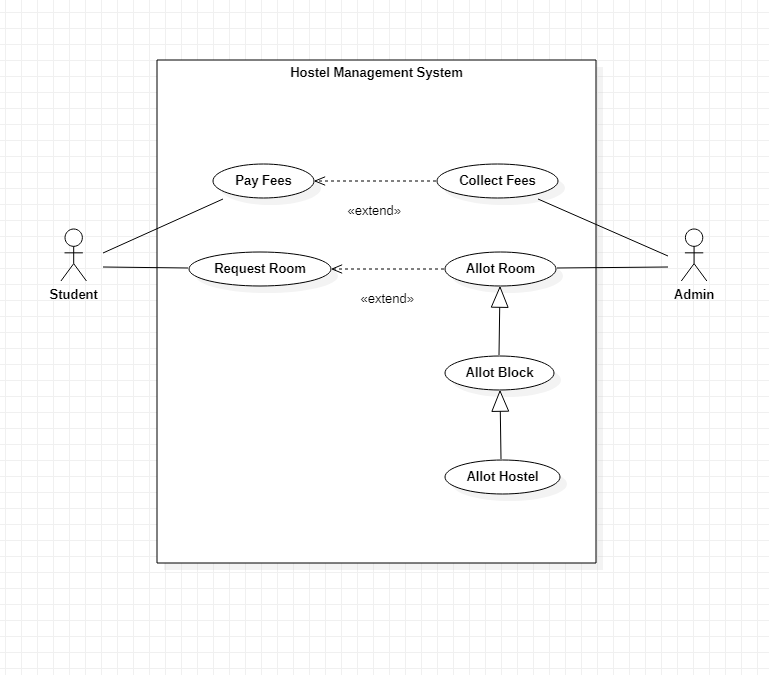


There is one actor:

* **Hostel:** The area where students are allowed to live

There are five use cases:

* **Get Information:** Get information about hostel
* **Edit Information:** Edit information about hostel
* **Check Vacancy:** Check for vacant rooms
* **Check Location:** Query location of vacant rooms
* **Get Student Details:** Get information about students residing in hostel



There are two actors:

* **Student:** Person who requests a room
* **Admin:** Person who allots rooms and collects fees

There are six use cases:

* **Pay Fees:** Pay fees for staying at hostel
* **Request Room:** Request hostel room
* **Collect Fees:** Collect fees for staying at hostel
* **Allot Room:** Allot room for student
* **Allot Block:** Allot block for student
* **Allot Hostel:** Allot hostel for student

**3. Stock Maintenance System**

**Problem Statement:**

To design a Stock Maintenance System to store information regarding stock of products and conduct transactions. The system will allow administrators to maintain stock of products by streamlining process of purchasing from suppliers and selling to customers. The information regarding stock is easily accessible and allows for ease of use.

**SRS:**

**Purpose:**

The Stock Maintenance System is an online system. The system is developed to provide detailed information regarding transactions related to stock maintenance and ease the process of carrying out such transaction. The system allows administrators to maintain stock of products by streamlining process of purchasing from suppliers and selling to customers.

**Requirements:**

**Functional Requirements:**

* Provide Admin Login:

Admin login is required to manage stock and carry out transactions.

* Provide Stock Details:

Details of product in stock such as quantity, quality and price can be viewed

* Maintain Transaction History:

All details of transactions carried out with suppliers and customers should be stored in database

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

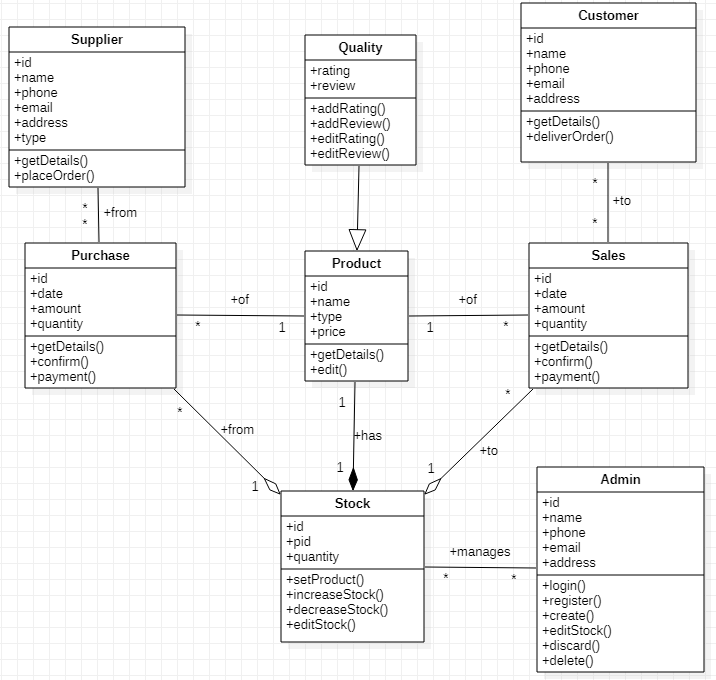
* Time:

The software product should be completed within deadlines

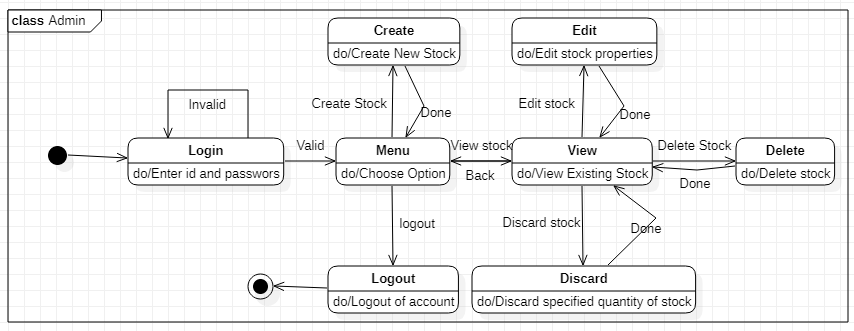
* Security:

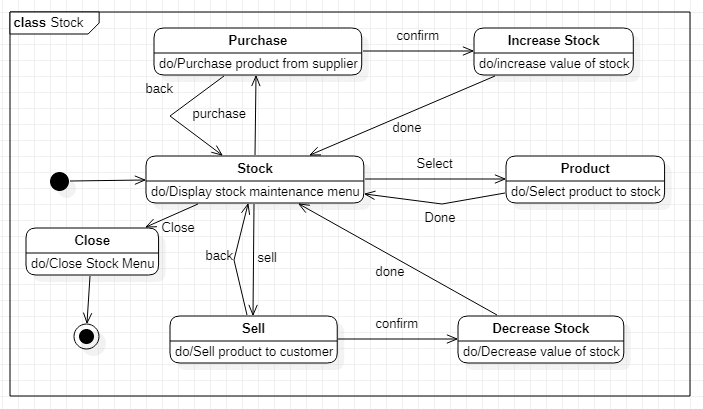
The data should be secure

**Class Diagram:**

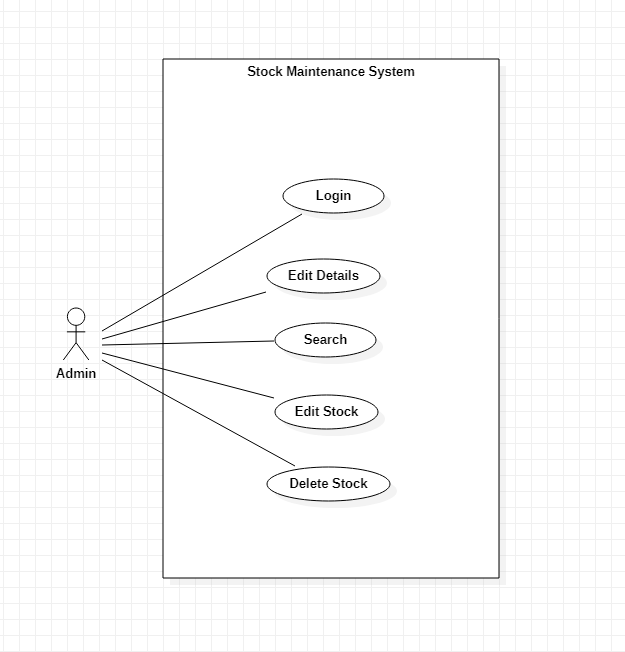
****

**State Diagram:**

****

****

**Use Case Diagrams:**

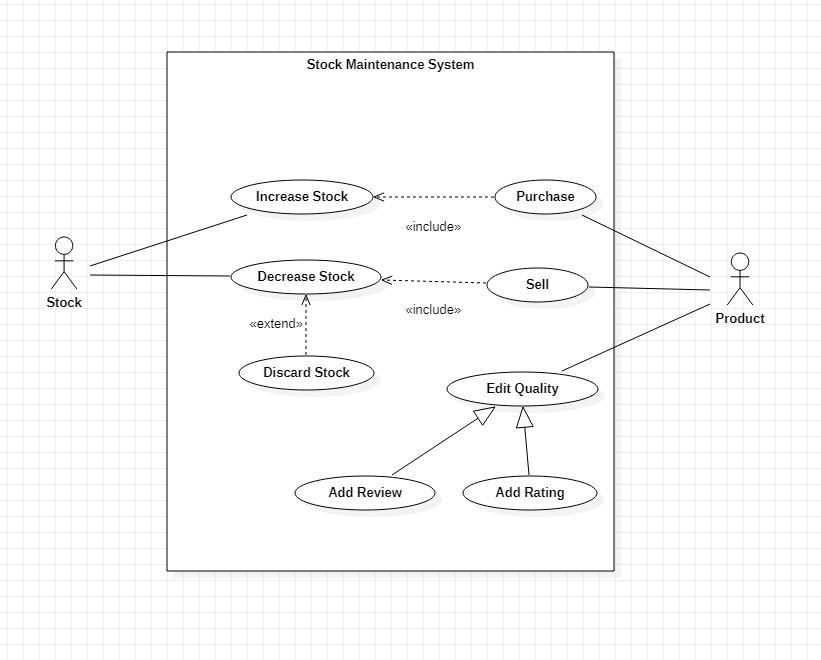


There is one actor:

* **Admin:** Person who manages the stock

There are five use cases:

* **Login:** Login to Stock Management Database
* **Edit Details:** Edit account details
* **Search:** Search for products
* **Edit Stock:** Edit Stock details
* **Delete Stock:** Delete stock from database



There are two actors:

* **Stock:** The place where merchandise is kept
* **Product:** The merchandise that is being stocked

There are eight use cases:

* **Increase Stock:** Increase stock of products
* **Decrease Stock:** Decrease stock of products
* **Discard Stock:** Discard faulty products
* **Purchase:** Purchase product from supplier
* **Sell:** Sell product to customer
* **Edit Quality:** Edit quality of product
* **Add Review:** Add product review
* **Add Rating:** Add product rating

**4. Coffee Vending Machine**

**Problem Statement:**

To design a Coffee Vending Machine to brew coffee based on user input. It reduces the time required to brew coffee by eliminating the need for manual preparation. The user enters the type of coffee they desire and inserts the required amount of money.

**SRS:**

**Purpose:**

The Coffee Vending Machine is an automated system. The system allows users to enter the type of coffee they desire and it is automatically prepares the coffee. It saves time and effort by eliminating the need for manual preparation of coffee. The ingredients are stored in the machine and the user does not require knowledge of how to make coffee in order to use it.

**Requirements:**

**Functional Requirements:**

* Choose Type:

User can enter type of coffee they desire.

* Get Coffee:

Checks the recipe for selected coffee type and brews the coffee.

* Set Temperature:

Allows user to set temperature of coffee.

* Validate Inserted Money:

Validates the money inserted by user and returns change.

* Dispense Coffee:

Dispenses coffee if required money inserted.

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

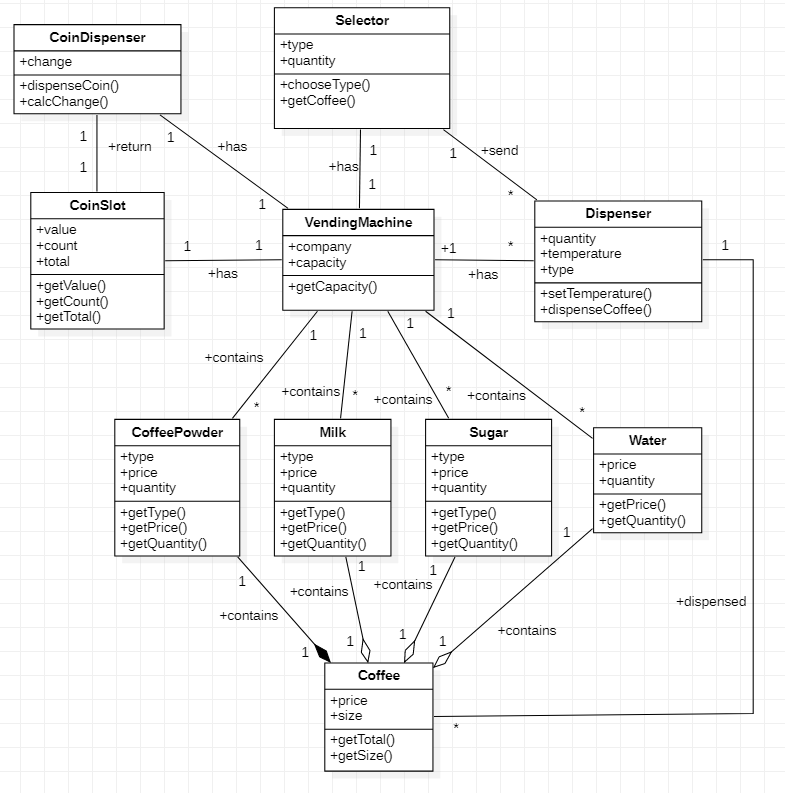
* Time:

The software product should be completed within deadlines

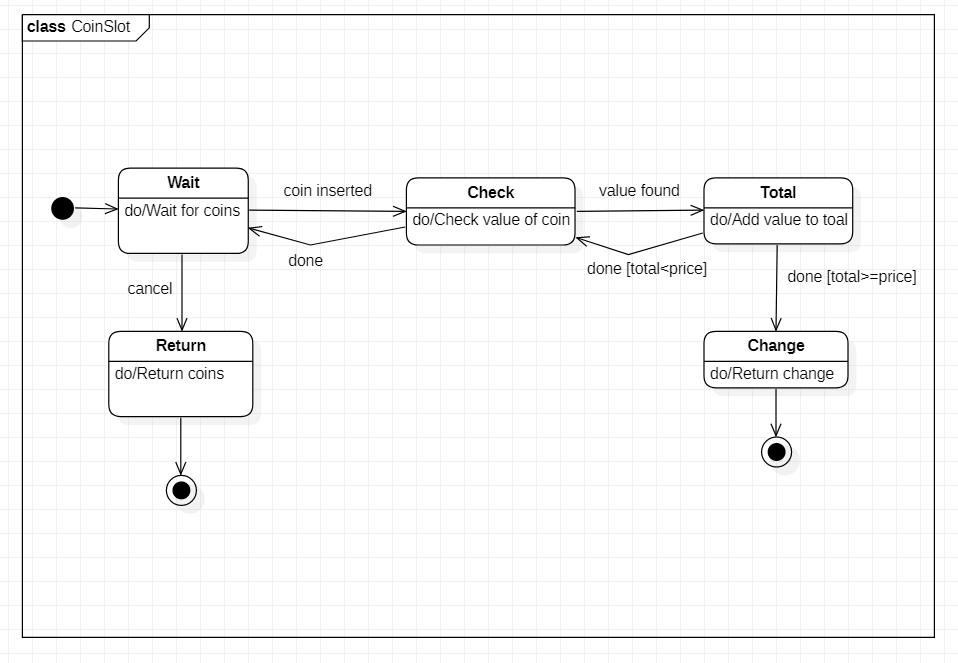
* Security:

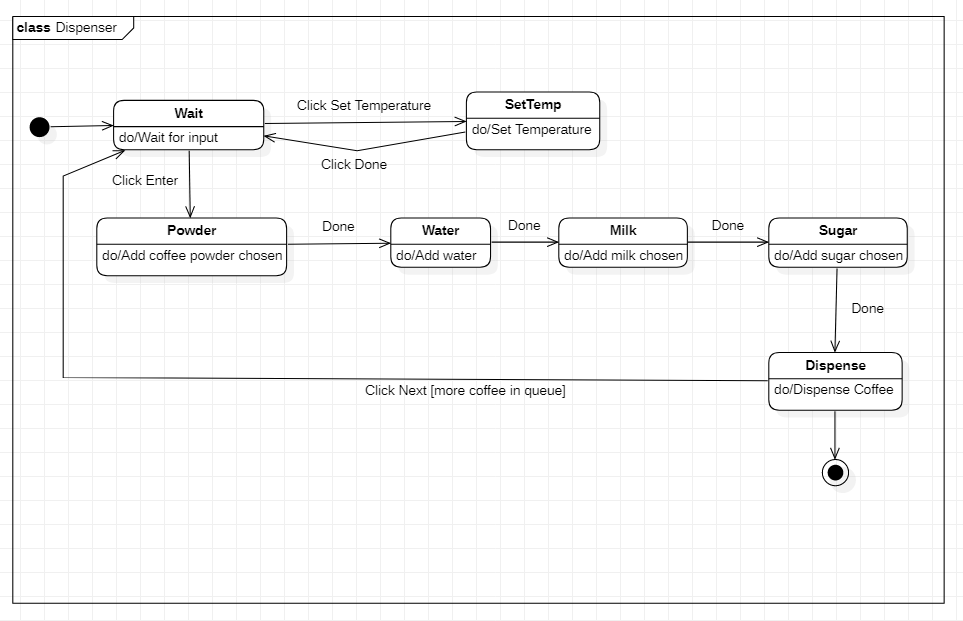
The data should be secure

**Class Diagram:**

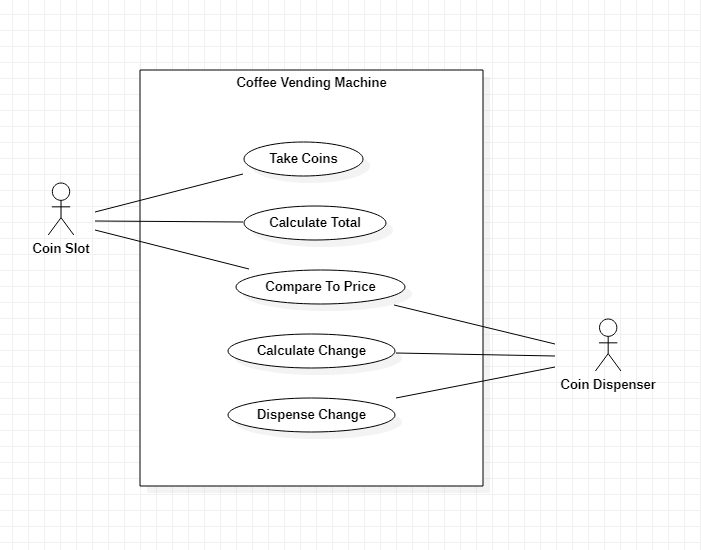
****

**State Diagram:**

****

****

**Use Case Diagrams:**

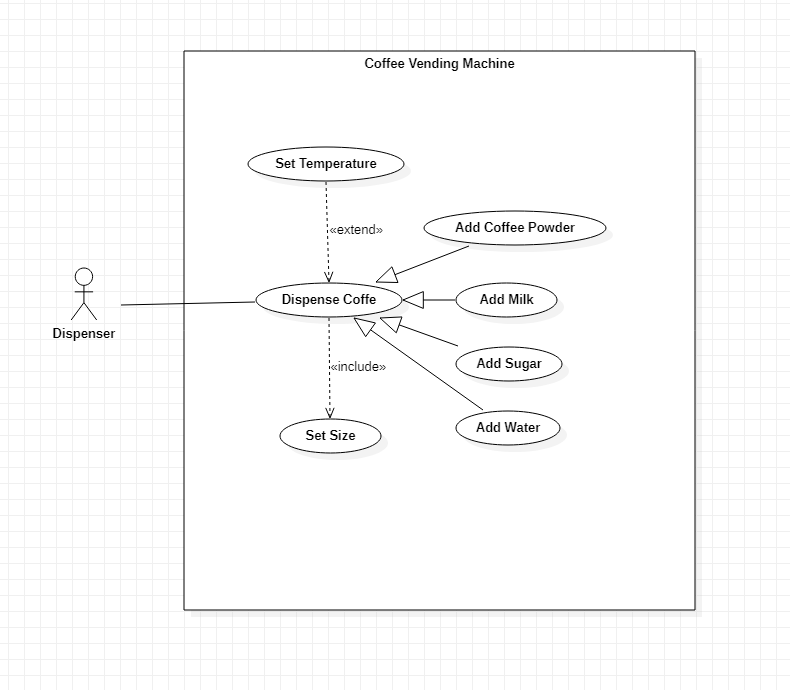


There are two actors:

* **Coin Slot:** The slot in which coins are inserted
* **Coin Dispense:** The dispenser that returns change

There are five use cases:

* **Take Coins:** Take coins from user
* **Calculate Total:** Calculate total value of inserted coins
* **Compare To Price:** Compare value inserted to price of coffee
* **Calculate Change:** Calculate change to return
* **Dispense Change:** Return change to user



There is one actor:

* **Dispenser:** The equipment that mixes and dispenses coffee

There are seven use cases:

* **Dispense Coffee:** Dispense coffee as selected
* **Add Coffee Powder:** Add coffee powder to mix
* **Add Milk:** Add milk to mix
* **Add Sugar:** Add sugar to mix
* **Add Water:** Add water to mix
* **Set Size:** Set size of coffee desired
* **Set Temperature:** Set desired temperature for coffee

**5. Online Shopping System**

**Problem Statement:**

To design an Online Shopping System to simplify to allow user to purchase products online. The system will allow users to search for and buy products online. The payment is made online and user can track the order.

**SRS:**

**Purpose:**

The Online Shopping System is an online system. The system is developed to allow users to search for and purchase products online. The user can make payments online and track the delivery of the product. All transactions are made and online and information is directly stored in database thereby reducing paperwork, travelling and time consumption.

**Requirements:**

**Functional Requirements:**

* Provide User Login:

Users can login and edit details of their account.

* Show Products:

User can search for and view details of products

* Provide Cart:

User can add product to cart and buy items together.

* Provide Online Payment Facility:

User can pay for items online.

* Provide Delivery Tracking Facility:

User can view details of delivery and check estimated delivery time.

* Provide Admin Login:

Admin login is required to update available.

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

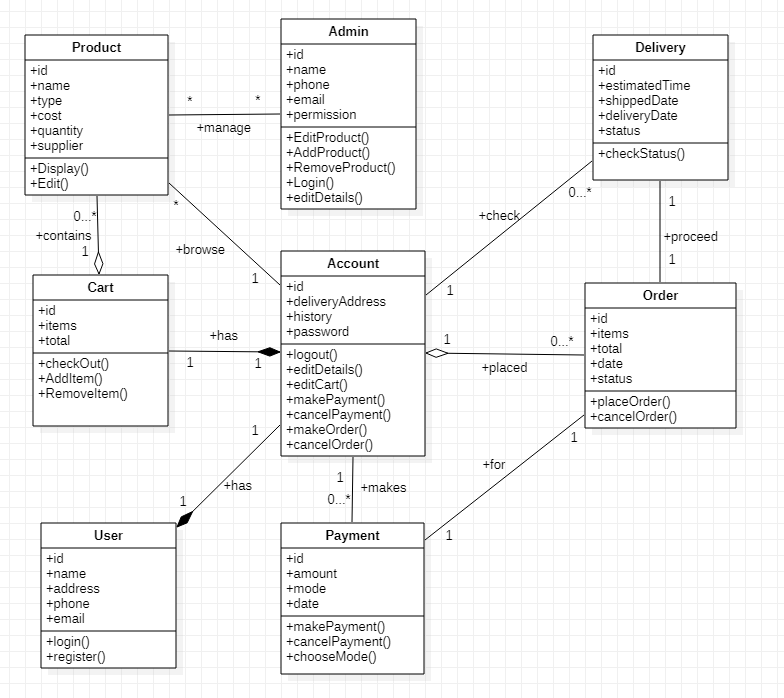
* Time:

The software product should be completed within deadlines

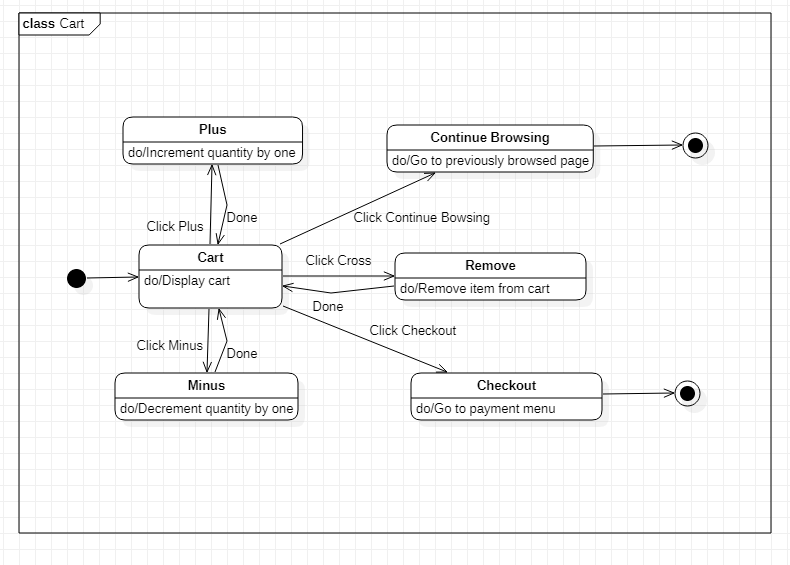
* Security:

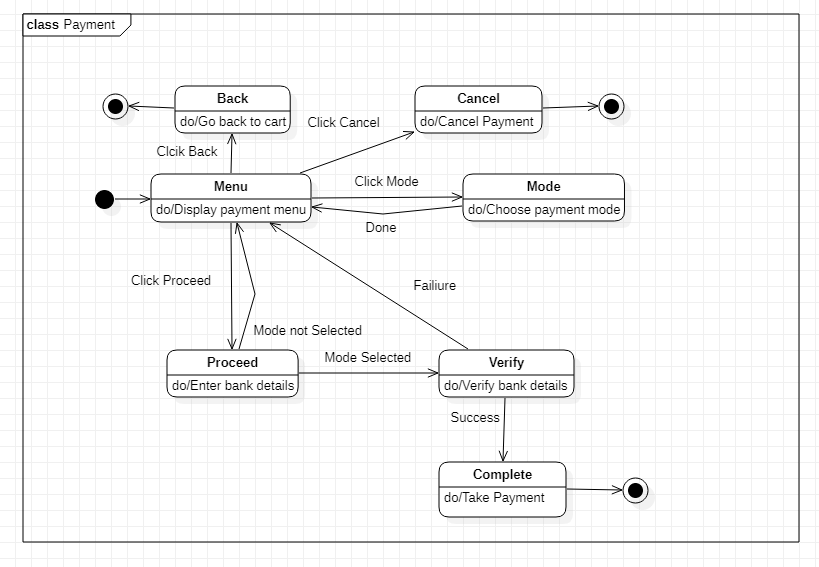
The data should be secure

**Class Diagram:**

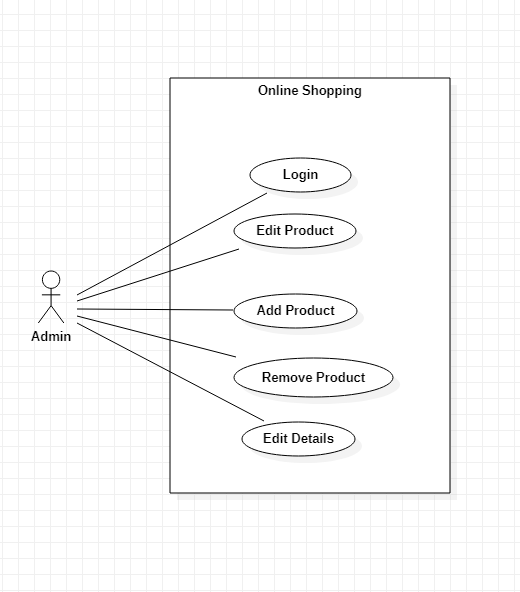
****

**State Diagram:**

****

****

**Use Case Diagrams:**

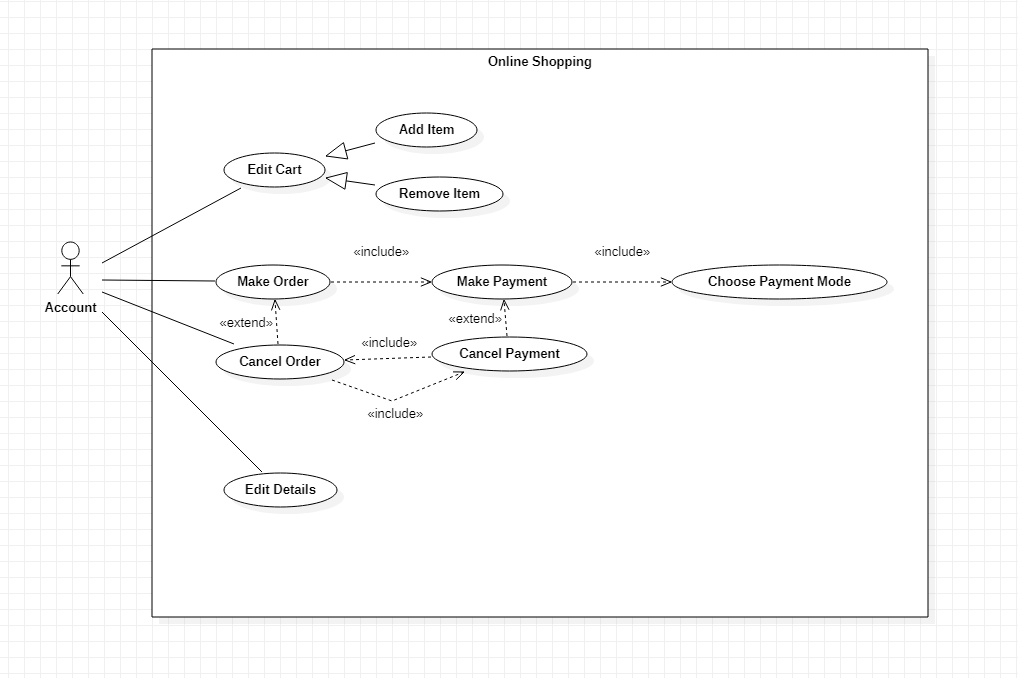


There is one actor:

* **Admin:** Person who manages products for sale

There are five use cases:

* **Login:** Login to online shopping website
* **Edit Product:** Edit Product details
* **Add Product:** Add product to store
* **Remove Product:** Remove product from store
* **Edit Details:** Edit account details



There is one actor:

* **Account:** Created by user to make transactions

There are nine use cases:

* **Edit Cart:** Edit shopping cart
* **Add Item:** Add items to shopping cart
* **Remove Item:** Remove items from shopping cart
* **Make Order:** Confirm order for items from cart
* **Make Payment:** Make payment for order
* **Cancel Order:** Cancel order before delivery
* **Cancel Payment:** Cancel payment for an order
* **Choose Payment Mode:** Choose payment mode for order
* **Edit Details:** Edit account details

**6. Railway Reservation System**

**Problem Statement:**

To design a Railway Reservation System to simplify the process of booking train tickets. The system will allow users to book tickets online. The system can access information regarding train schedules and seat availability and quickly provide tickets and eliminates unnecessary paperwork, travelling and time consumption.

**SRS:**

**Purpose:**

The Railway Reservation System is an online system. The system is developed to facilitate faster booking of train tickets. The system allows users to book train tickets online. The system can access information regarding train schedules and seat availability and quickly provide tickets and eliminates unnecessary paperwork, travelling and time consumption. Administrators can edit train schedules through the system as well.

**Requirements:**

**Functional Requirements:**

* Provide User Login:

Users can login and edit account details.

* Provide Facility to book tickets online:

User can enter trip details and tickets will be provided after system verification.

* Provide Online Payment Facility:

User can pay for tickets online.

* Provide Admin Login:

Admin login is required to update train schedules.

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

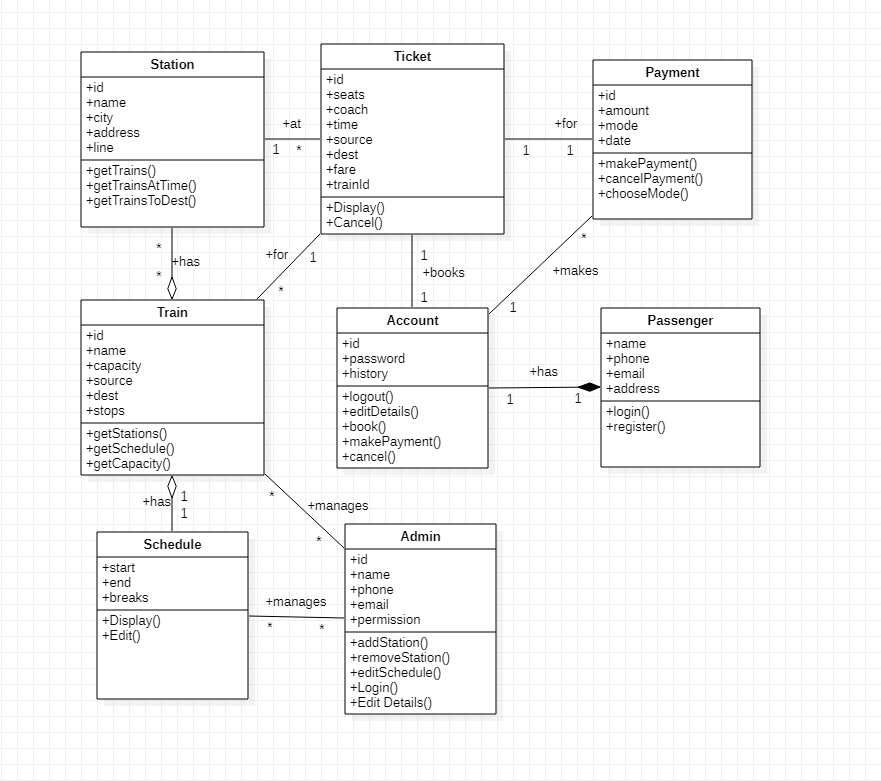
* Time:

The software product should be completed within deadlines

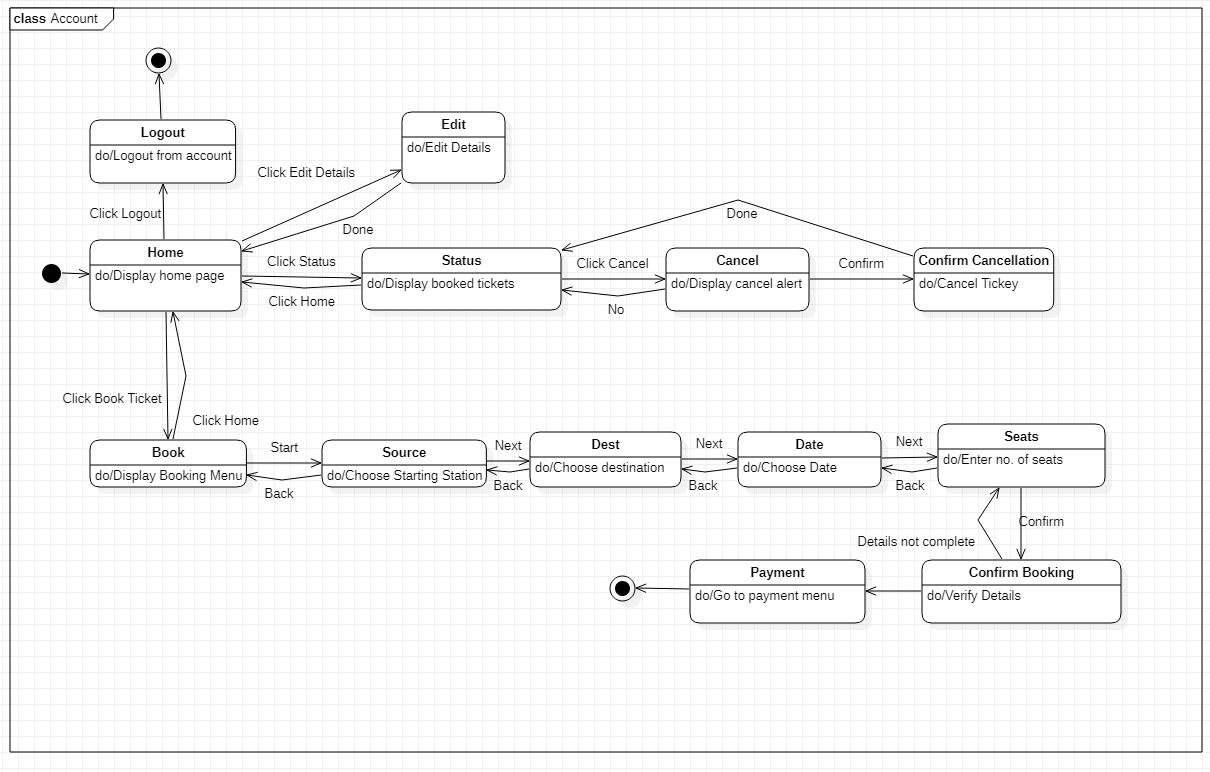
* Security:

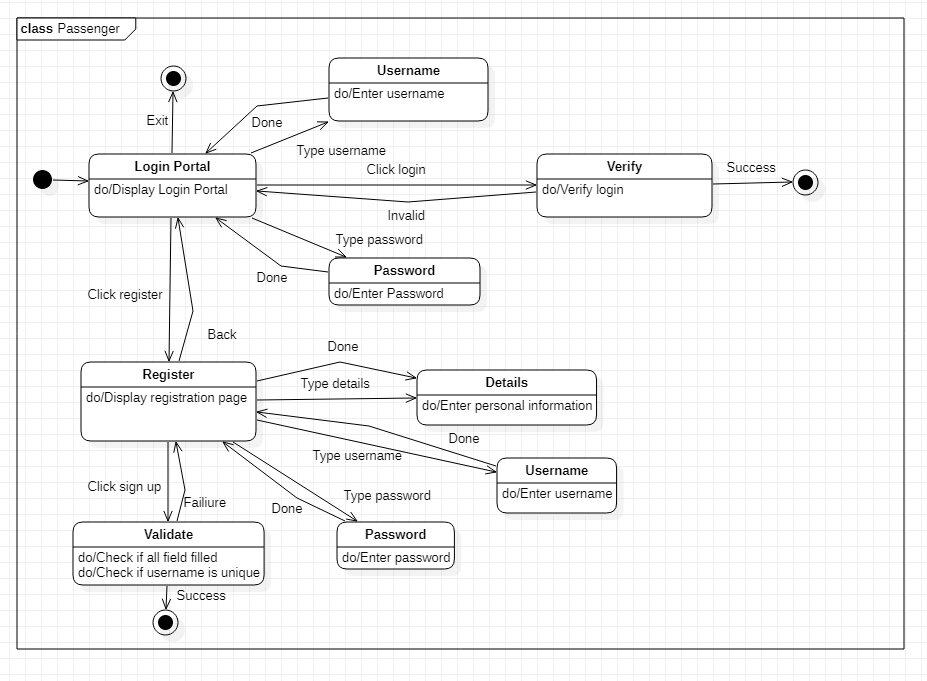
The data should be secure

**Class Diagram:**

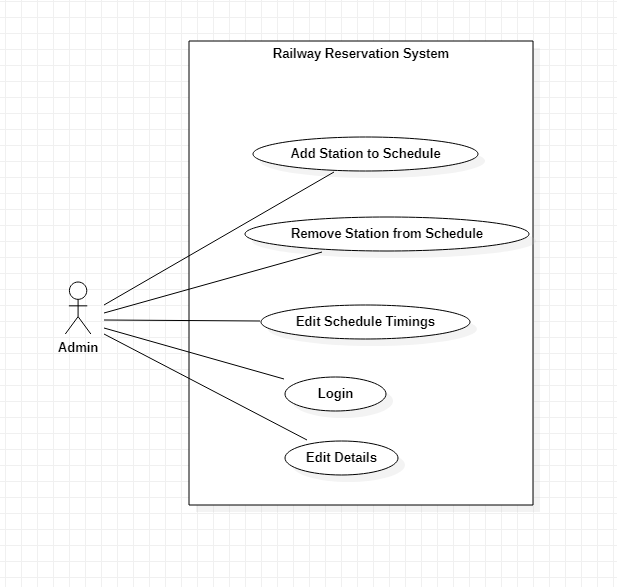
****

**State Diagram:**

****

****

**Use Case Diagrams:**

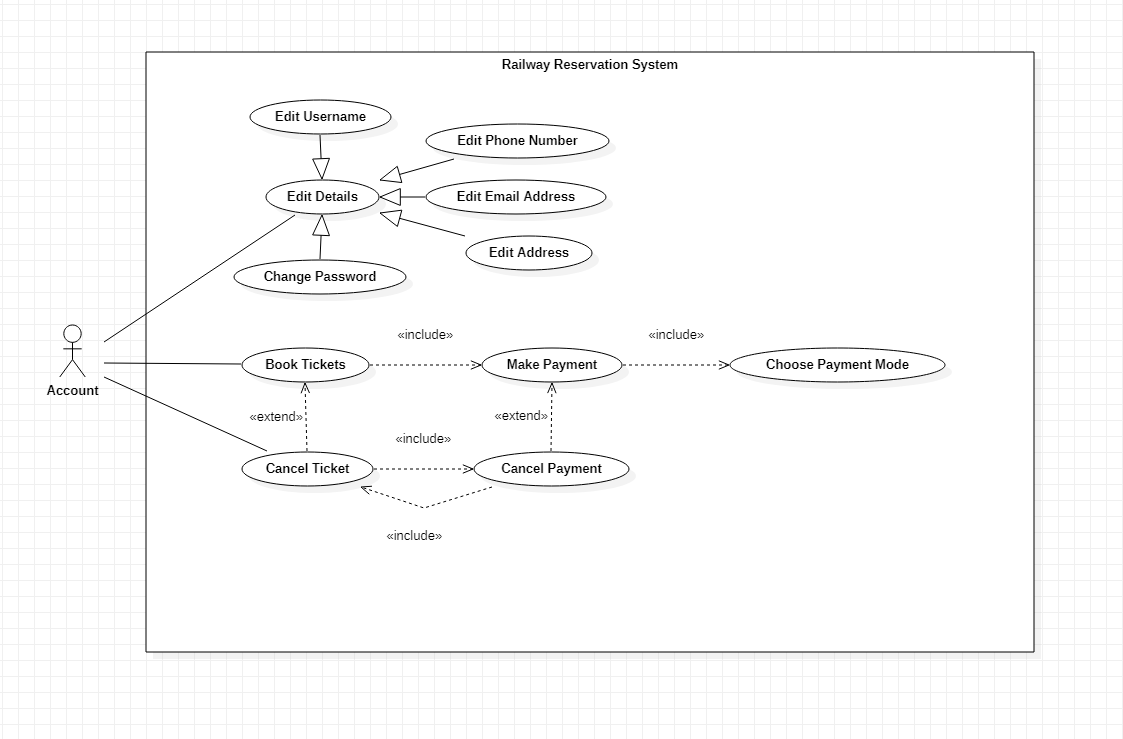


There is one actor:

* **Admin:** Person who manages train schedules

There are five use cases:

* **Add Station to Schedule:** Add station to schedule of train
* **Remove Station from Schedule:** Remove station from schedule of train
* **Edit Schedule Timings:** Edit timings in schedule of train
* **Login:** Login to railway reservation database
* **Edit Details:** Edit account details



There is one actor:

* **Account:** Created by user to book tickets

There are eleven use cases:

* **Edit Details:** Edit account details
* **Edit Username:** Edit account username
* **Edit Phone Number:** Edit account phone number
* **Edit Email Address:** Edit account email address
* **Edit Address:** Edit account address
* **Change Password:** Change password for account
* **Book tickets:** Confirm booking of tickets
* **Make Payment:** Make payment for tickets
* **Cancel Ticket:** Cancel booked tickets
* **Cancel Payment:** Cancel payment for tickets
* **Choose Payment Mode:** Choose payment mode for booking

**7. Graphics Editor**

**Problem Statement:**

To design a Graphic Editor to allow user to represent information using graphics. The editor will allow users to draw, edit text, color, etc. using tools provided. It allows for quick and easy graphical representation of data.

**SRS:**

**Purpose:**

The Graphic Editor is a visual editing software. The Graphics Editor is developed to allow users to visually represent data using graphics. The editor allows users to draw, edit text, color, etc. using tools provided. Various inbuilt templates are provided to represent information. It allows for quick and easy graphical representation of data.

**Requirements:**

**Functional Requirements:**

* Provide Document Options:

Allows users to open new or existing documents, save documents, print, share or export documents, etc. Document templates can be loaded. Layers can be added and objects can be grouped.

* Provide Toolbox:

Tools can be selected from toolbox to draw objects. Tools can be customized according to user preference.

* Provide Palette:

Colors can be selected from palette. Palette can be customized according to user preference.

* Provide Shapes:

User can add predefined shapes to document.

* GUI:

An interface is developed for above mentioned functional requirements.

**Non-Functional Requirements:**

* Cost:

The software should be cheap and within budget.

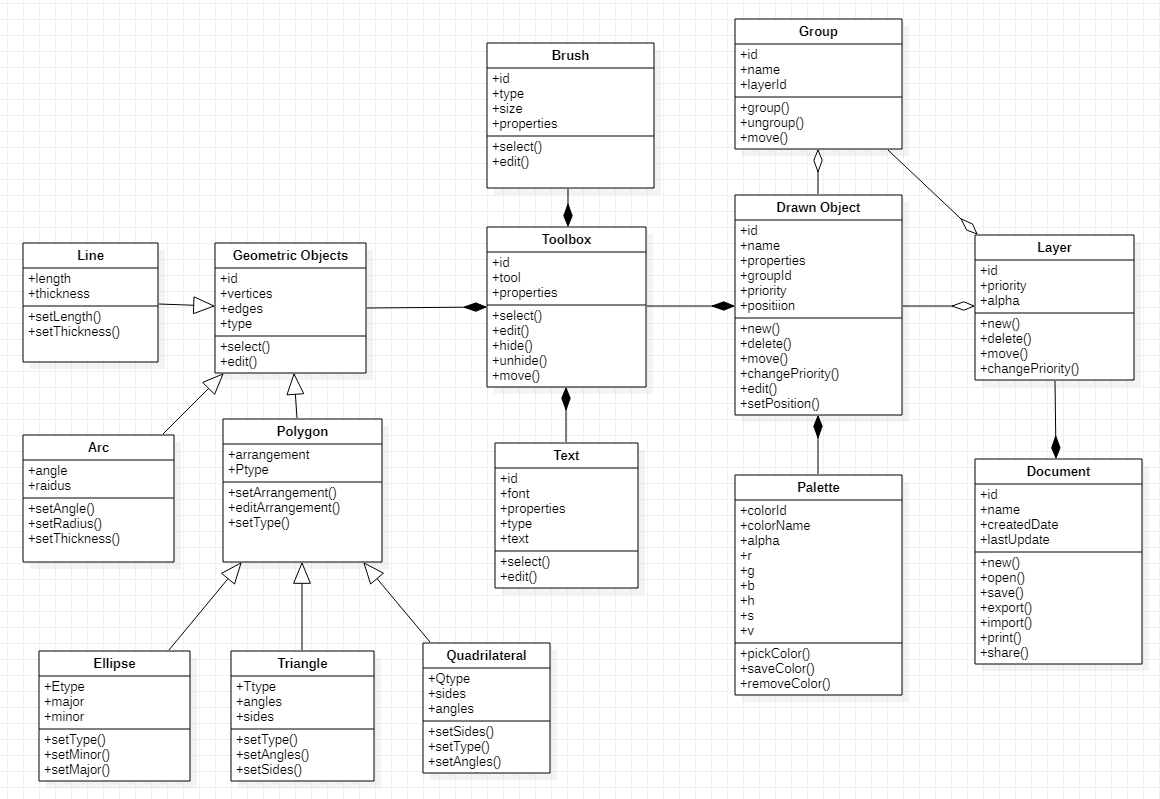
* Time:

The software product should be completed within deadlines

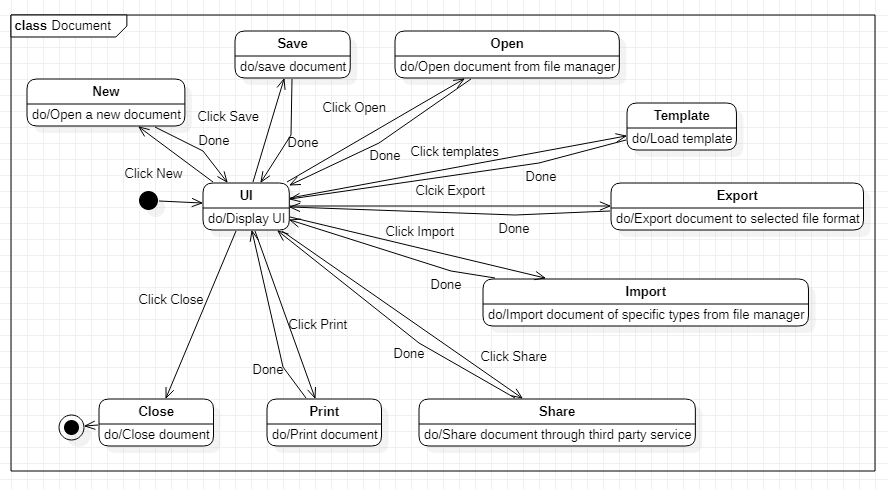
* Security:

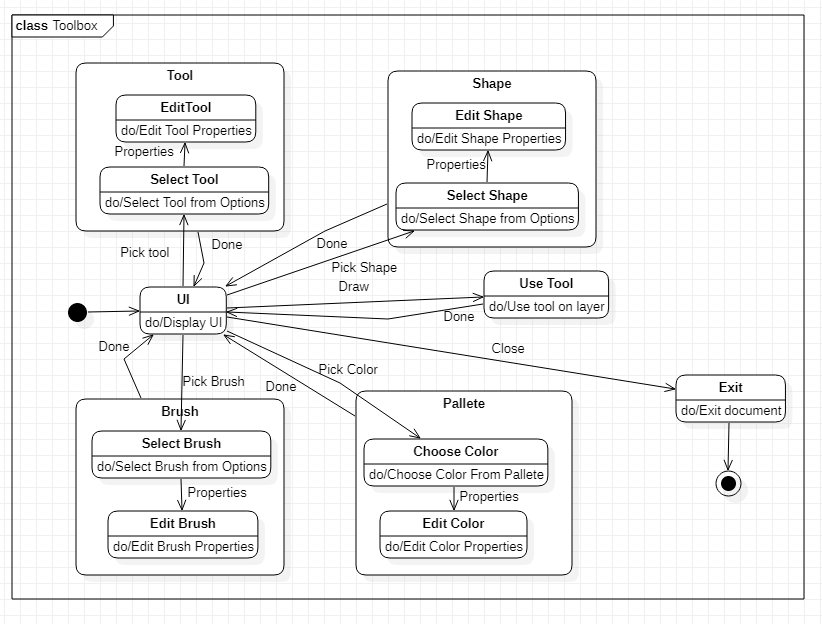
The data should be secure

**Class Diagram:**

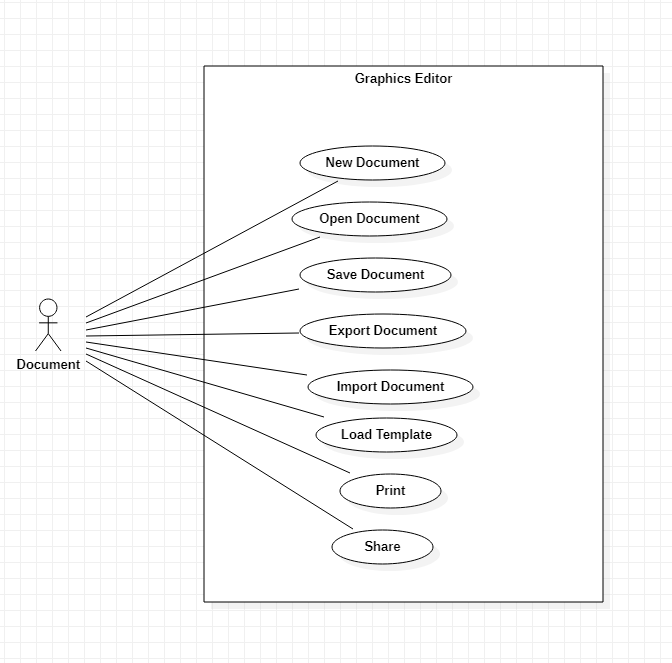
****

**State Diagram:**

****

****

**Use Case Diagrams:**

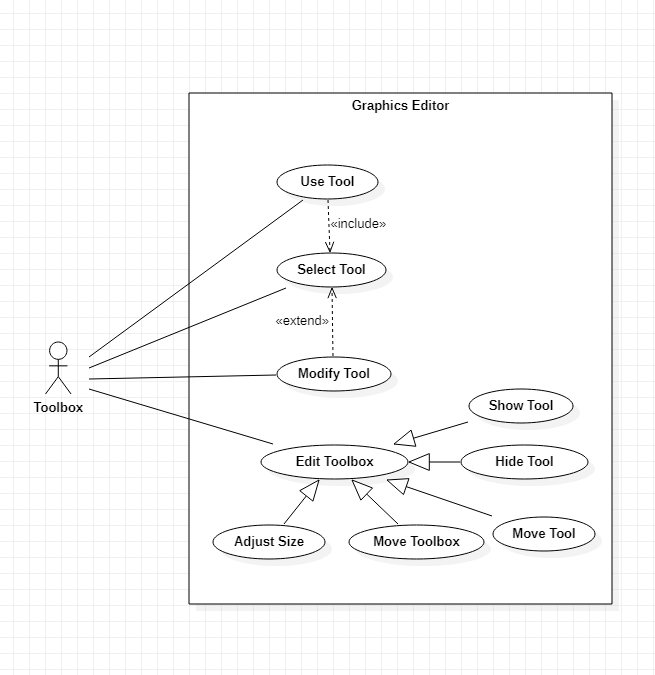


There is one actor:

* **Document:** File that stores changes made by user

There are eight use cases:

* **New Document:** Create a new document
* **Open Document:** Open am existing document
* **Save Document:** Save document on computer
* **Export Document:** Export document to other file formats
* **Import Document:** Import document from other file formats
* **Load Template:** Load template for document
* **Print :** Print document
* **Share:** Share document with others



There is one actor:

* **Toolbox:** UI interface that provides options for user to modify document.

There are nine use cases:

* **Use Tool:** Use selected tool
* **Select Tool:** Select tool from toolbox
* **Modify Tool:** Modify selected tool settings
* **Edit Toolbox:** Edit toolbox UI
* **Show Tool:** Show tool in toolbox
* **Hide Tool:** Hide tool in toolbox
* **Move Tool:** Move tool in toolbox
* **Move Toolbox:** Move toolbox on screen
* **Adjust Size:** Adjust size of toolbox