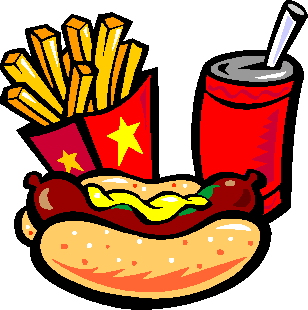
Food Court Transaction

**Software Requirements Specification**



*Submitted By*

**Akshay Malav(B16CS003)**

**Lovish Singla(B16CS013)**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **People** |
| 12 Feb 2018 | 1.0 | First draft | Developers: Akshay Malav , Lovish Singla  Maintenance: Akshay Malav, Lovish Singla |

**Content**

1. **Introduction 4**

1.1 Purpose 4

1.2 Scope 4

1.3 Constraints 5

1.4 Assumptions and Dependencies 5

1.5 Definitions, Acronyms and Abbreviations 5

1.6 References 5

1. **Overall Description 6**

2.1 Product Functions 6

2.2 User Characteristics 18

1. **Specific Requirements**

3.1 Use case description 19

3.2 Reliability 22

3.3 Performance Requirements 22

3.4 Supportability 23

3.5 Design Constraints 23

3.6 Interfaces 24

4. **Supporting Information 25**

4.1 Appendix 25

**￼ Software Requirements Specification**

1. **Introduction**
   1. **Purpose**

The purpose of this Document is to elucidate the functionalities of the Food court

transaction system .It also explains step by step how to use the software, that is act as a

guide to the person who is using it for the first time . It also talks about the constraints, the

prerequisites and the special features the system have. Proper explanation has been given

wherever required . This Document will not only help the clients but also the developers

who wants to improve the software further.

* 1. **Scope**

In today's world speed matters and it matters the most when it comes to food. People

want to place their order quickly and get their food as soon as possible. So there is a need

of a software/system that is fast, efficient and error free. So the need for a

efficient software is there.

This software provides the functionalities to the food court team. The registered

receptionist can take order form the customer , have to update the stock

corresponding to the items that are sold and give the customer a bill of the food items

purchased. Manager can keep track of the employees that are working in the food court ,

can make changes in the stock if required and can also see the report which will help the

manager to decide an optimal price for a particular product to increase the sale.

Also the salary is given to the receptionist according to the sale he/she makes . All the taxes

are also incorporated in the final bill.

* 1. **Constraints**
  + A receptionist can't make changes in the stock except when he/she is taking the order.
  + Manager can't take the order.
  + A receptionist cannot view the report.
  + A receptionist can't change his/her username.
  1. **Assumptions and Dependencies**
  + Whenever a input is required from the receptionist or manager , they should enter input in the correct format.
  + The making cost of a particular product is constant over a period of time.
  + There are only two type of products, one on which GST is applied and one on which GST is not applied.
  + There is only one manager for the food court.
  1. **Definitions, Acronyms and Abbreviations**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Receptionist | A person who sits at the counter and take orders from the customers. |
| Manager | A person who manages the food court. |
| Customer | A person who place the order. |

* 1. **References**

[1] The World Wide Web

[2]IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

[3] Cppreference.com, stackoverflow.com, tutorialpoint.com etc.

[4]UML @ Classroom: An Introduction to Object-Oriented Modeling , E Balaguruswamy Object Oriented Programming With C++ etc.

1. **Overall Description**
   1. **Product Functions**
      1. *Functional Requirements*
2. Take order

|  |  |
| --- | --- |
| **Input** | The input will be given by the receptionist as the customer chooses the food item either in terms of the food id or if the id is not known then the food item name. |
| **Output** | The order will be placed. |
| **Process** | After the input has been taken from the customer ,the customer will be asked to confirm the order ,if the customer agrees then the stock will be updated and the bill will be printed and if the customer does not agrees then he/she will be asked to remove the food items that is no longer required. |

(2) Show bill

|  |  |
| --- | --- |
| **Input** | The input will be given when the take order use case is being executed .The input will be the name ,price, type of the food item. |
| **Output** | The bill will be printed. |
| **Process** | After the order is confirmed the details of the order such as name , price, type will be given to the billing class and the corresponding bill will be printed. |

(3) Add receptionist

|  |  |
| --- | --- |
| **Input** | The input will be the name , username, password of the employee to be added. |
| **Output** | The database which contains the list of the receptionists will be updated. |
| **Process** | This is the functionality of the manager. Manager will be asked to enter the name , username , password for the new employee to be hired. |

(4) Update Stock

|  |  |
| --- | --- |
| **Input** | The input will be either updating the values such as name, price, type, quantity or adding a new food item in the list |
| **Output** | The menu will be updated |
| **Process** | Only manager can update the stock , manager will be asked to enter the values as described in the input section after that the stock will be updated. |

(5)Change details

|  |  |
| --- | --- |
| **Input** | The input will be the details to be changed by the user. |
| **Output** | Details updated. |
| **Process** | The employee will be asked to enter the new name or new password and any changes made will be updated in the database. |

(6) Remove receptionist

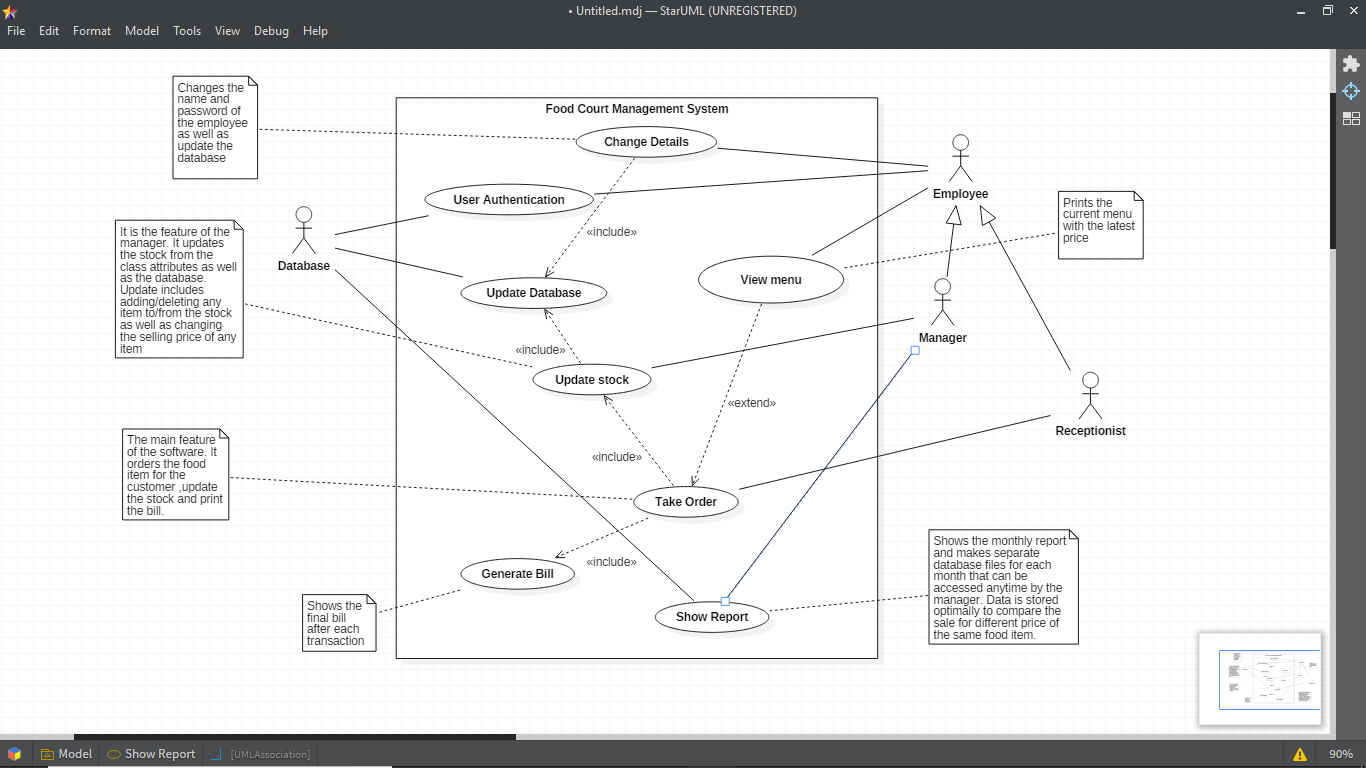
|  |  |
| --- | --- |
| **Input** | The input will be the details of the receptionist to be removed. |
| **Output** | The receptionist will be removed from the database. |
| **Process** | The Manager will have to enter the values like, name ,username, password and if the match is found in the database then the respective employee will be removed and if not then will be asked again to enter the correct values. |

(7) User authentication

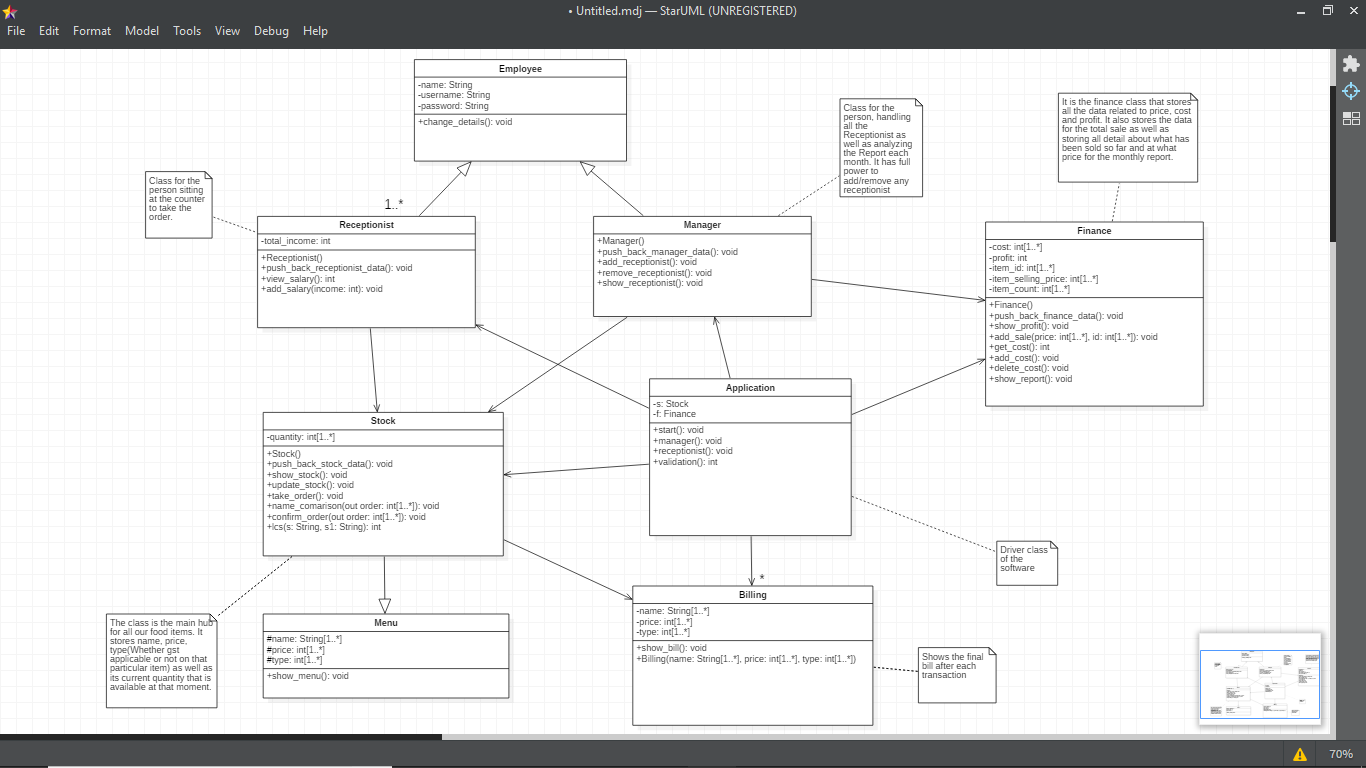
|  |  |
| --- | --- |
| Input | The input will be the username and password of the user. |
| Output | User will logged in to the system. |
| Process | The input that is taken from the user will be checked in the database and if credentials matched then the user will be logged into the system. |

**2.1.2 UML Diagrams**

2.1.2.1 Use Case Diagram

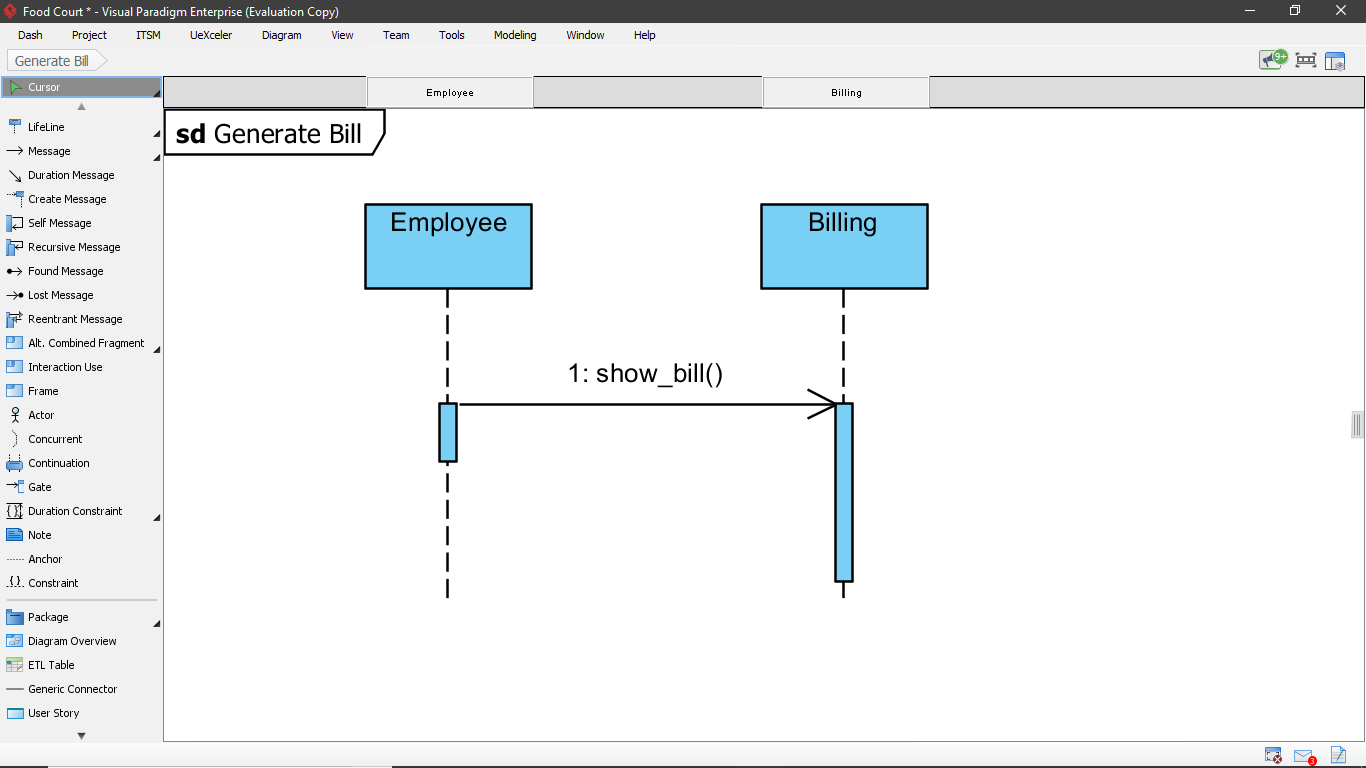


2.1.2.2 Class Diagram

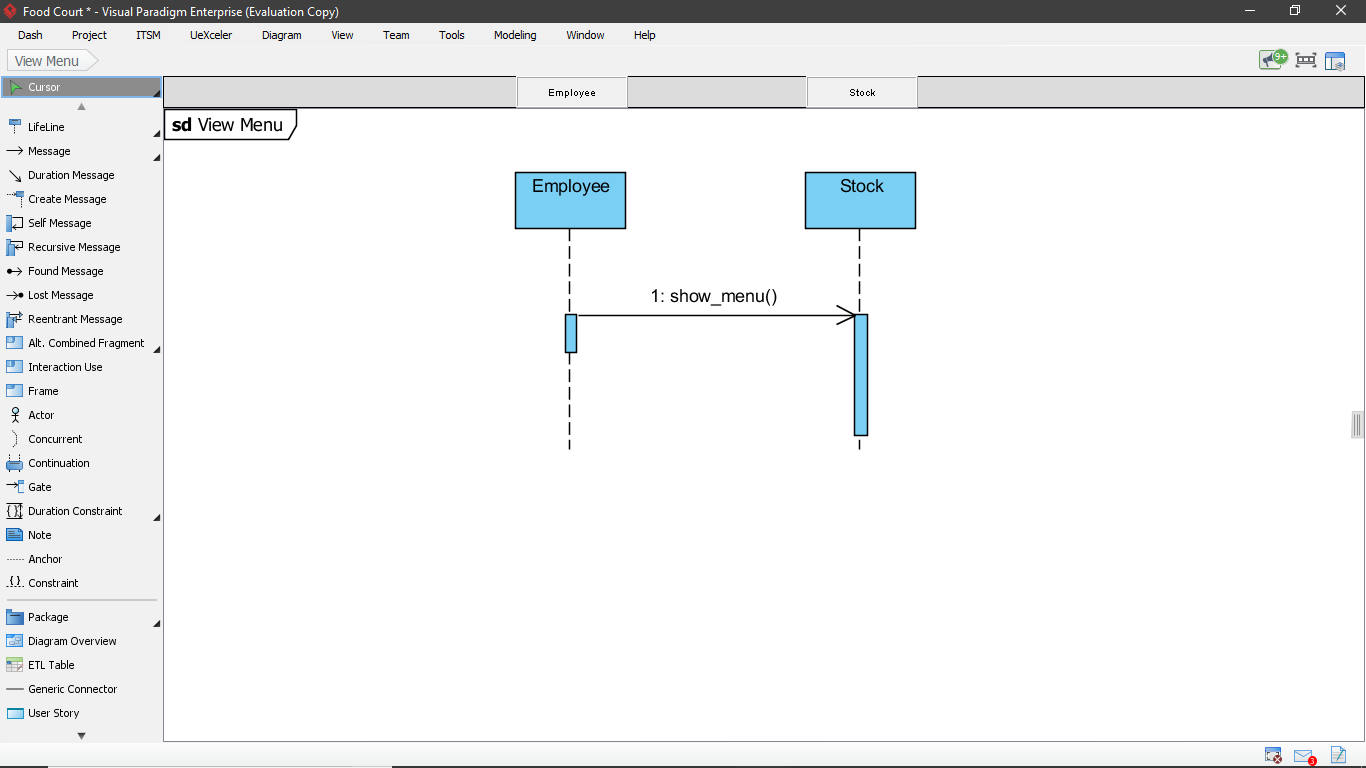


2.1.2.3 Sequence Diagram

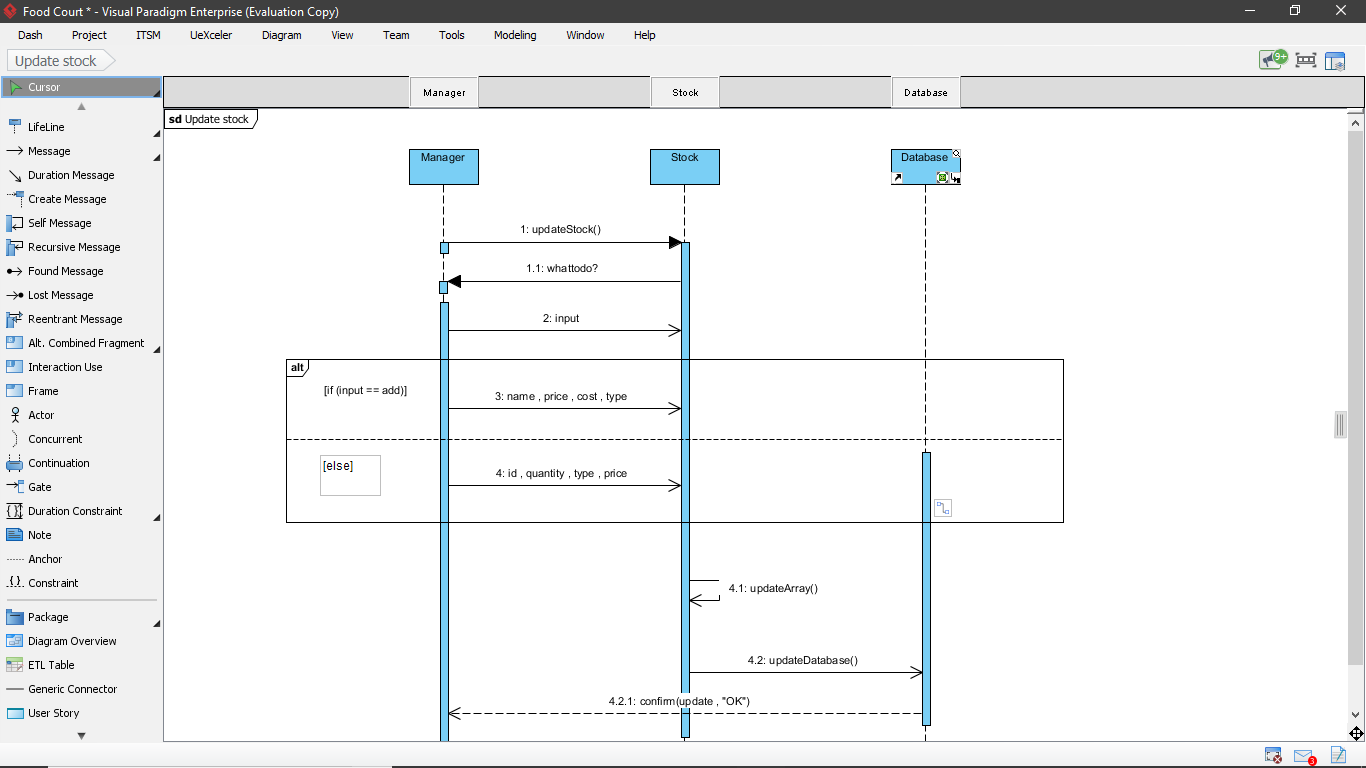
(I) Generate bill



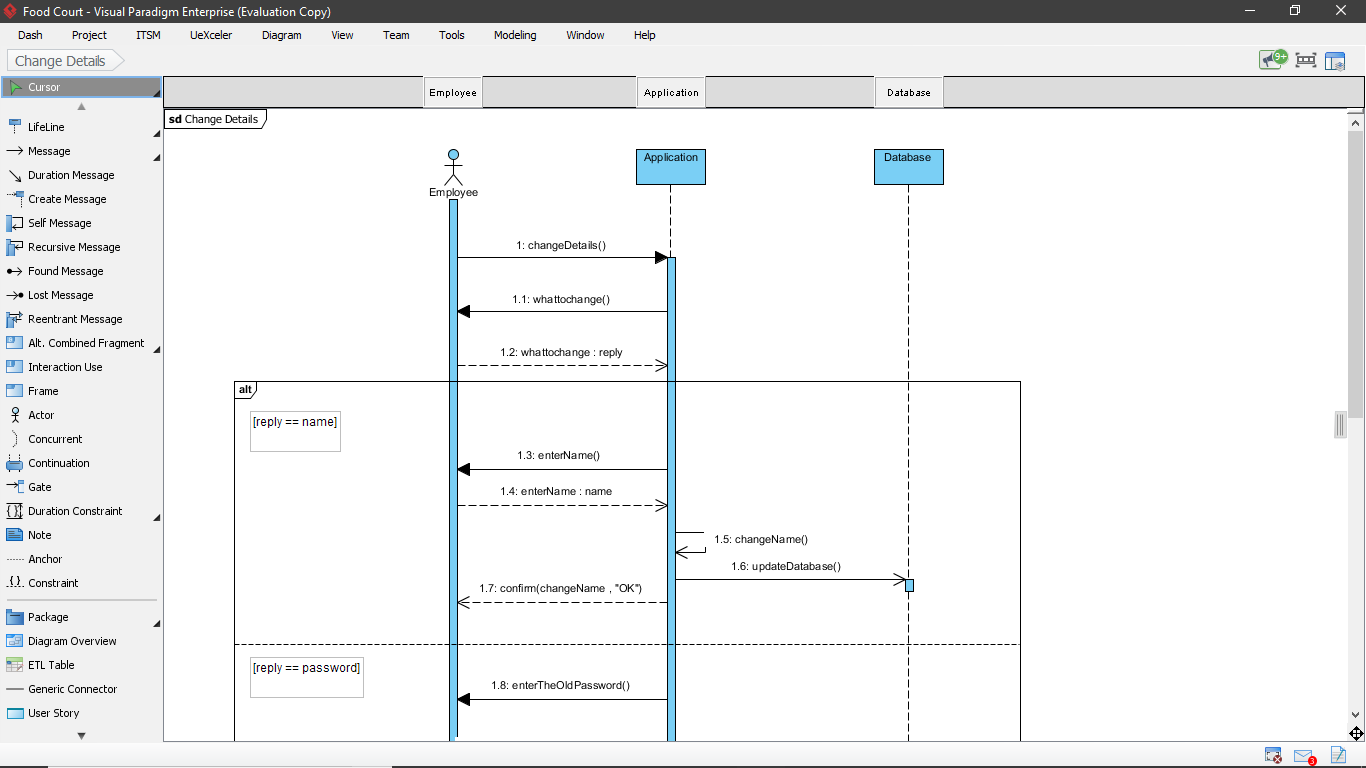
(II) View Menu



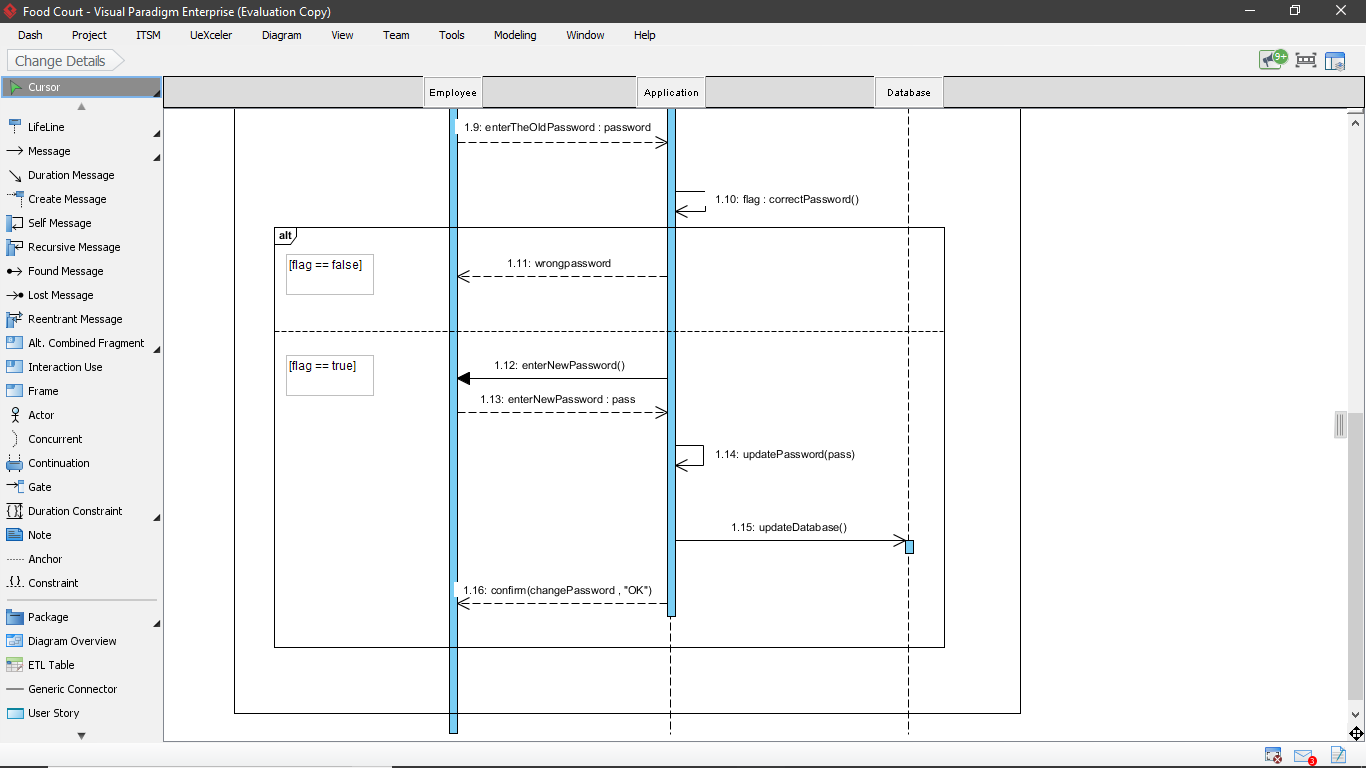
(III) Update Stock



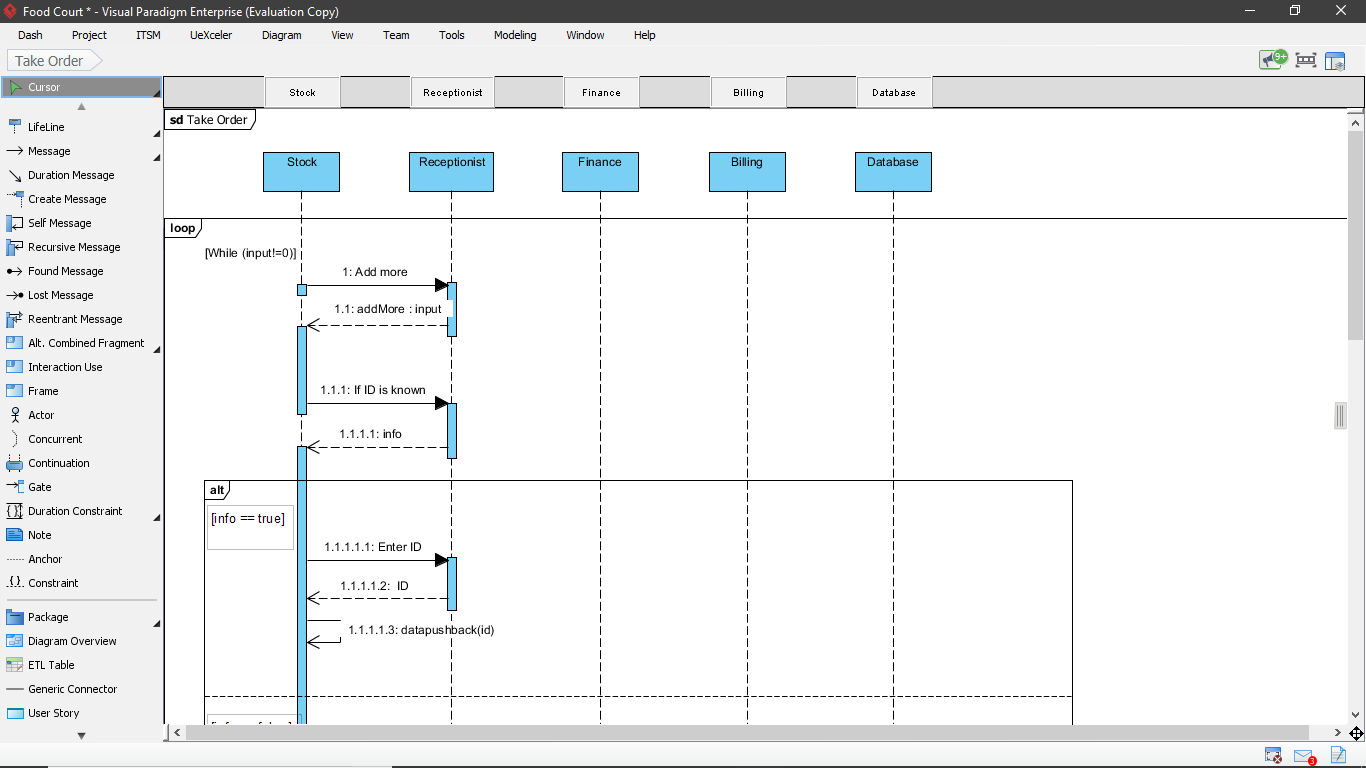
(IV) Change Details (i)



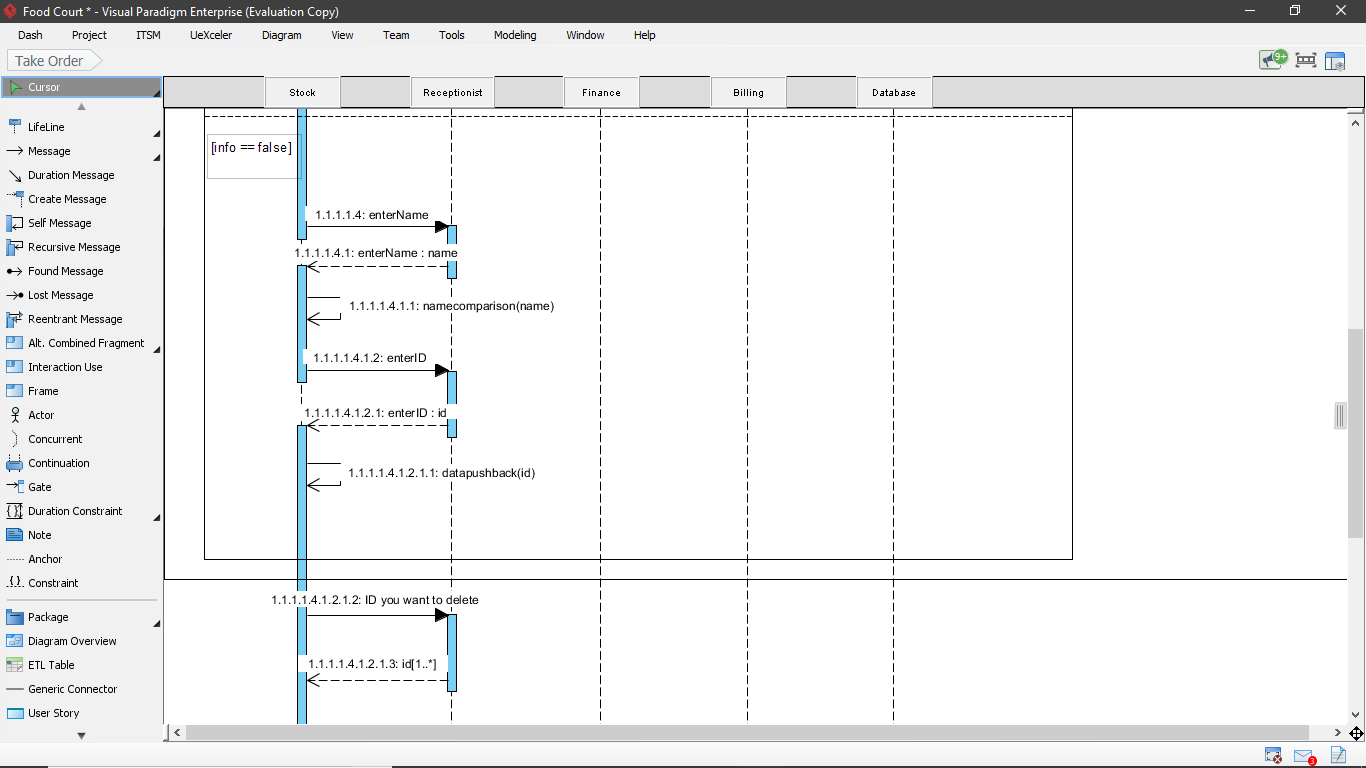
(IV) Change Details (ii)



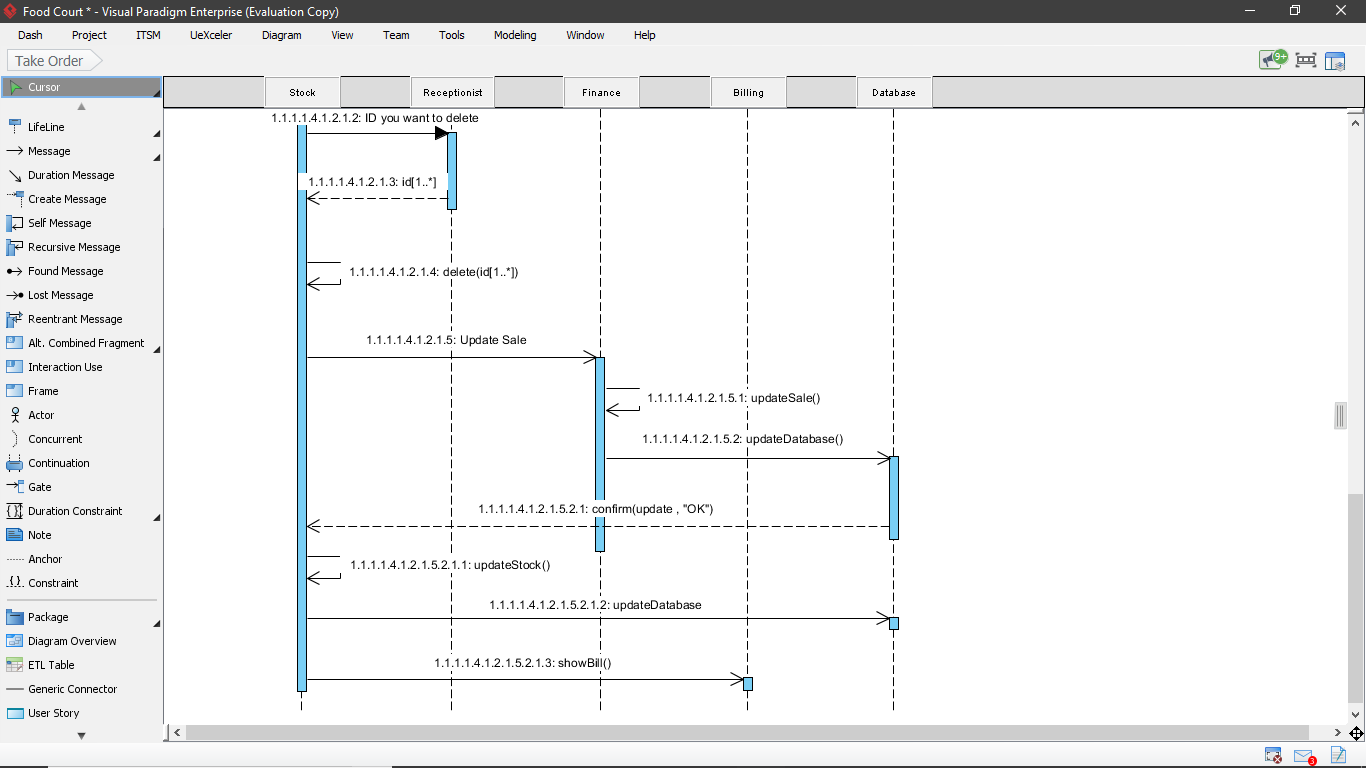
(V) Take order (i)



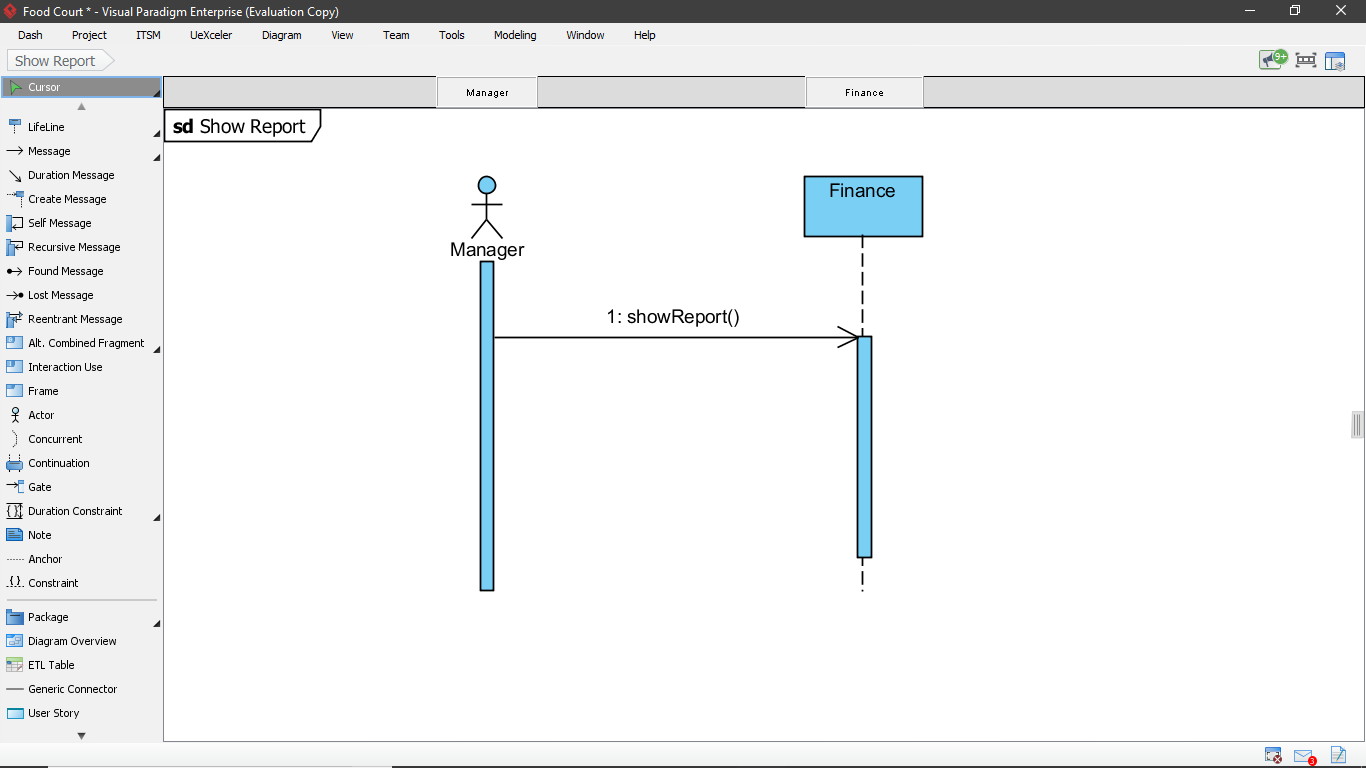
(V) Take order (ii)



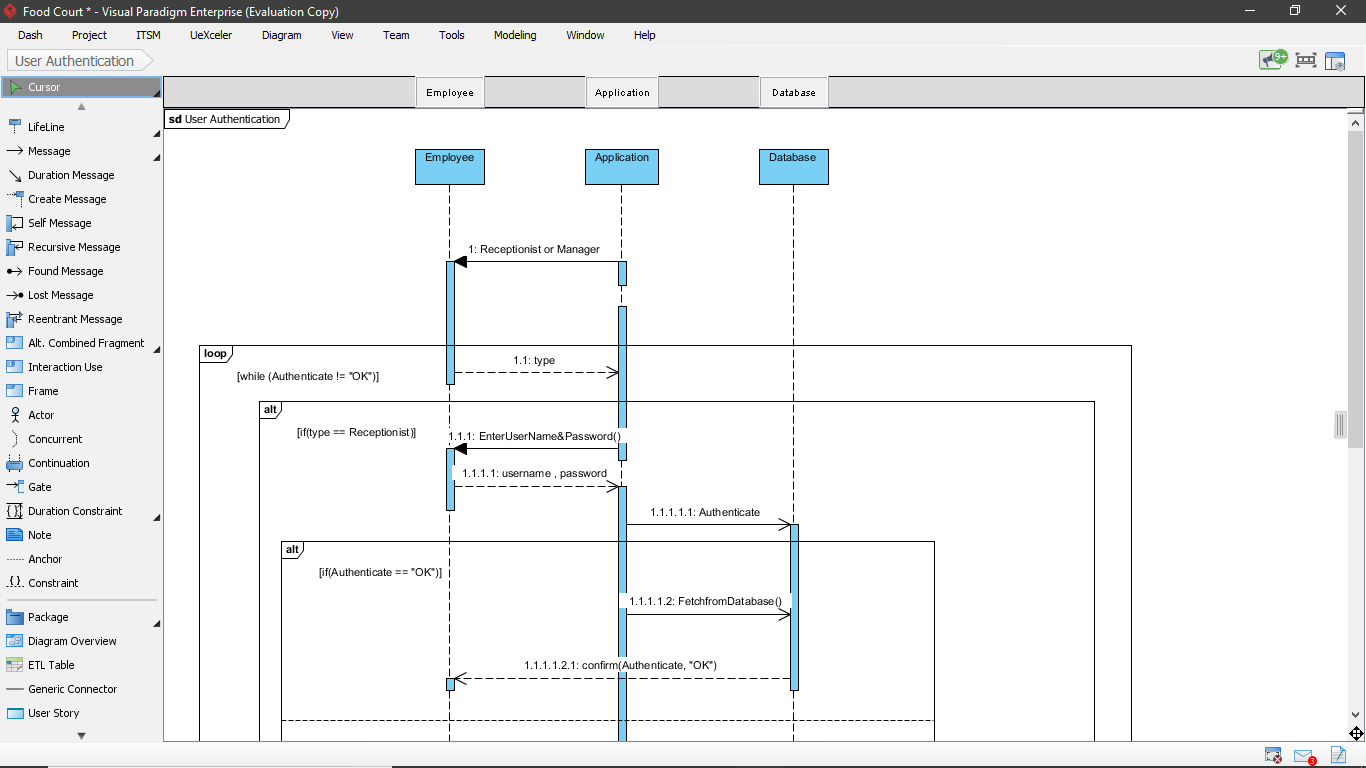
(V) Take order (iii)



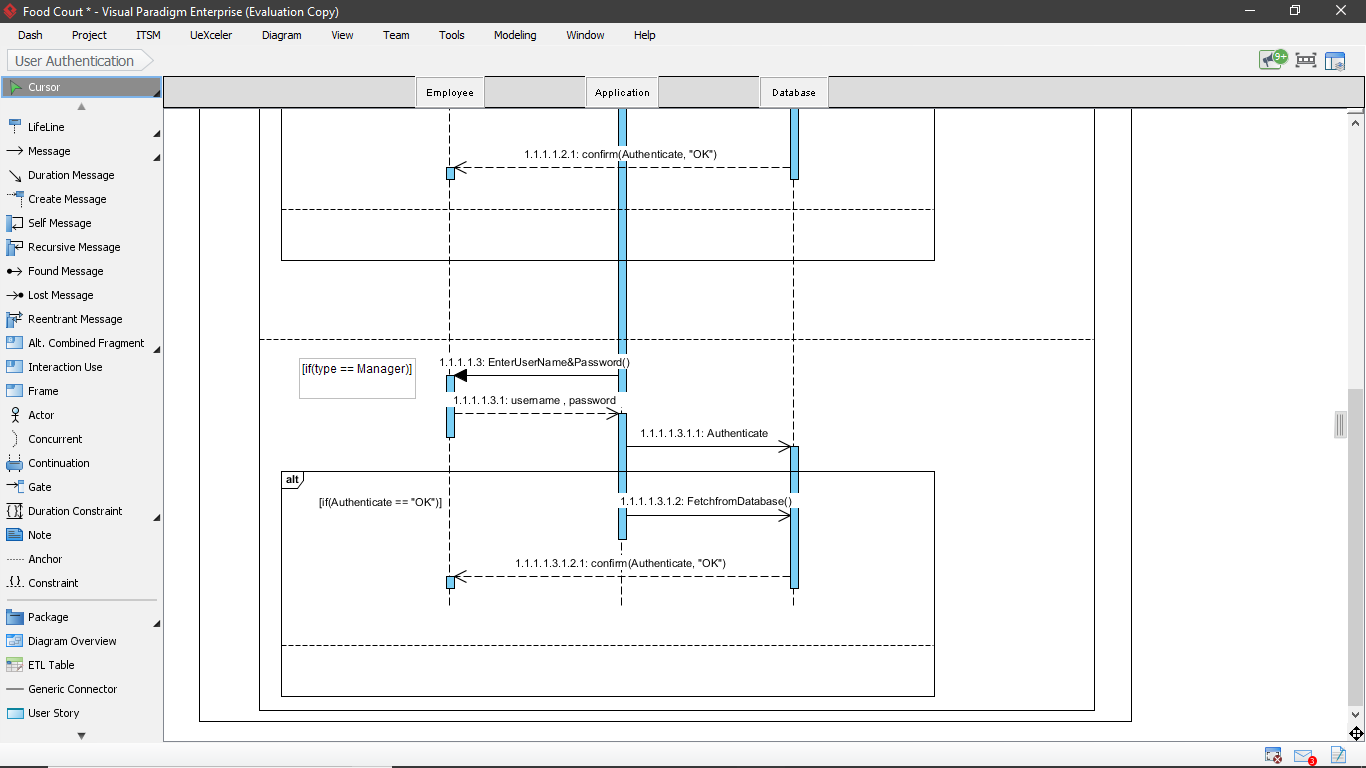
(VI) Show report



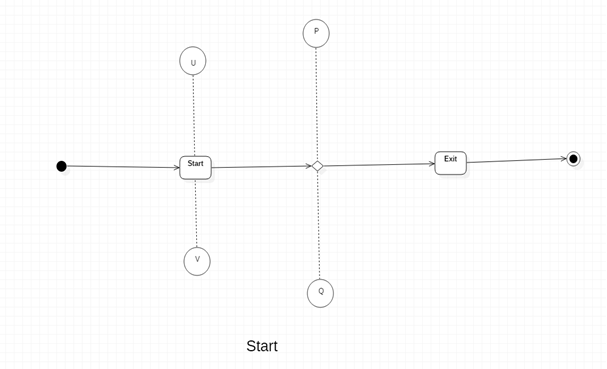
(VII) User authentication (i)



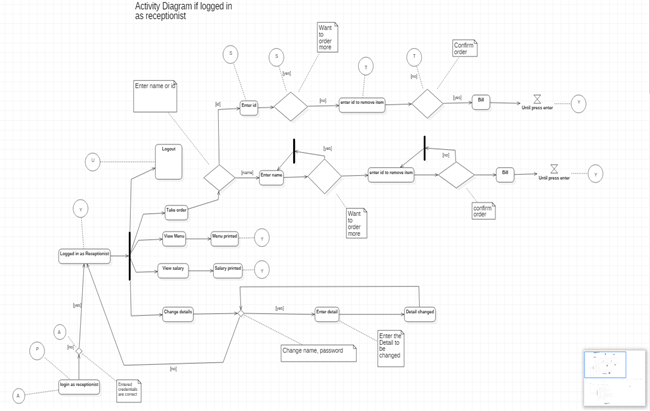
(VII) User authentication (ii)

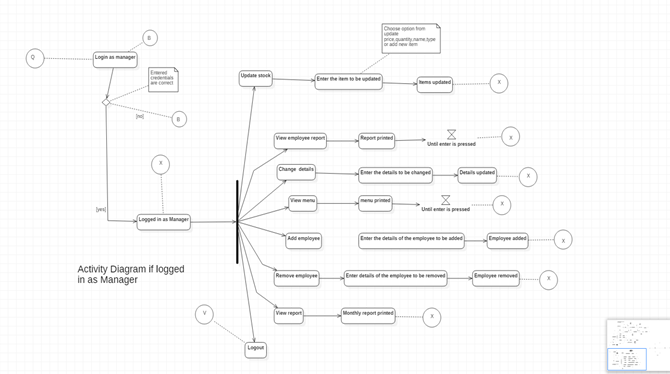


2.1.2.4 Activity Diagram



(I) Start

 (II)Activity Diagram if logged in as receptionist



(III) Activity Diagram if logged in as Manager

*2.1.3 Non-Functional Requirements*

This software is designed and developed for computers only. The computer should have a

operating system which is the basic requirement of the software to function. Also access should

be given by the computer to the software for reading or writing the data form the database which

will be stored on the hard drive of the computer. The computer should not have any kind of

viruses because it can harm the data stored in the database as well as the software.

* 1. **User Characteristics**

The software will be used by two type of users , one is receptionist and another is manager .

Although the functionalities of both the users are different but both also share some

common features.

A receptionist can take order whenever a customer comes , update the stock corresponding to

the food items that are sold. Also he/she can view the menu whenever required . A receptionist

can also view what total income he/she has earned through all the transactions that have

occurred when he/she was on the counter. A receptionist can also change personal details like

name, password.

The manager can view the monthly report which comprises of all the transactions that have

occurred during a particular month .The manager can also view menu, change personal details

like name , password (It is interesting to note that both these features are common between

manager and receptionist). Manager can also maintain the stock by updating it whenever

required and can also keep track of the employees(receptionists) by adding and removing them.

Also the manager can view a full list of the receptionists.

1. **Specific Requirements** 
   1. **Use case description**
      1. *Take order*

|  |  |
| --- | --- |
| **Name:** | Take order |
| **Short Description:** | The receptionist takes the order from the customer. |
| **Pre condition:** | The receptionist has been given access to take order from the customer after being logged in to the system. |
| **Post condition:** | The order is placed and a bill is printed. |
| **Error situations:** | A particular item is not available. |
| **System state in the event of an error:** | The order is not placed and an error message is shown. |
| **Actors:** | Receptionist |
| **Trigger:** | Customer wants to place an order. |
| **Standard process:** | 1. The receptionist logs in. 2. Customer places an order. 3. The receptionist checks the availability of the items. 4. The customer confirms the order and a bill is printed. 5. And the stock is updated. |
| **Alternative process:** | (4')The item is not available.  (5')An error message is shown. |

* + 1. *Show bill*

|  |  |
| --- | --- |
| **Name:** | Show bill |
| **Short Description:** | A bill is printed corresponding to an order. |
| **Pre condition:** | An order has to be placed. |
| **Post condition:** | A bill is printed |
| **Error situations:** | An order is not placed |
| **System state in the event of an error:** | The bill is not printed and will be at the logged in page of receptionist. |
| **Trigger:** | A bill is to be given to the customer |
| **Standard process:** | 1. An order is placed. 2. A bill is printed. |
| **Alternative process:** | (1')The order is not placed.  (2') The bill is not shown. |

* + 1. *Update Stock*

|  |  |
| --- | --- |
| **Name:** | Update stock |
| **Short Description:** | The manager make changes in the stock. |
| **Pre condition:** | The manager has been given access to update the stock after being logged in to the system. |
| **Post condition:** | The changes that the manager would have wanted are being done. |
| **Error situations:** | The item to which changes are to be made is not present. |
| **System state in the event of an error:** | An error message is shown and manager is asked to enter the correct details. |
| **Actors:** | Manager |
| **Trigger:** | Manager wants to update the stock. |
| **Standard process:** | 1. The manager logs in using the credentials. 2. On the page manager chooses to update stock. 3. Manager is asked to update the stock. 4. All the changes that are made, are saved. |
| **Alternative process:** | (3')The manager thinks to not update the stock  (4')Manager is redirected to it's home page. |

* + 1. *Show Report*

|  |  |
| --- | --- |
| **Name:** | Show report |
| **Short Description:** | Manager wants to see the monthly report. |
| **Pre condition:** | The Manger has to login in order to see the monthly report. |
| **Post condition:** | The report is shown to the manager. |
| **Error situations:** | The manager enters invalid/incorrect login credentials . |
| **System state in the event of an error:** | Manager is asked to enter the credentials again. |
| **Actors:** | Manager, Database |
| **Trigger:** | Manager wants to view report |
| **Standard process:** | 1. Manager is asked to login. 2. Manager selects the option to view report. 3. Monthly report is shown to the manager |

* + 1. *Change Details*

|  |  |
| --- | --- |
| **Name:** | Change details |
| **Short Description:** | An employee changes personal details. |
| **Pre condition:** | The employee have to login in order to change the details. |
| **Post condition:** | The personal details are changed. |
| **Error situations:** | While changing details the entered data is invalid. |
| **System state in the event of an error:** | The employee will be asked to enter it again. |
| **Actors:** | Employee |
| **Trigger:** | Employee wants to change personal details. |
| **Standard process:** | 1. The employee is asked to login. 2. Now employee can change personal details. |
| **Alternative process:** | None |

* + 1. *View menu*

|  |  |
| --- | --- |
| **Name:** | View menu |
| **Short Description:** | The employee view the menu of the food court. |
| **Pre condition:** | The employee have to login to view menu. |
| **Post condition:** | The menu is shown to the employee. |
| **Error situations:** | The login credentials are invalid. |
| **System state in the event of an error:** | Employee will be asked to enter the credentials again. |
| **Actors:** | Employee |
| **Trigger:** | Employee wants to view menu. |
| **Standard process:** | 1. The employee will be asked to login either as a manager or as a receptionist. 2. Then employee chooses to view menu. 3. Menu will shown to the employee. |
| **Alternative process:** | None |

3.1.7 User authentication

|  |  |
| --- | --- |
| **Name:** | User authentication |
| **Short Description:** | The user will be authenticated. |
| **Pre condition:** | The user has to enter login credentials. |
| **Post condition:** | The user will be logged in to the system. |
| **Error situations:** | The login credentials are invalid. |
| **System state in the event of an error:** | Employee will be asked to enter the credentials again. |
| **Actors:** | Employee, Database |
| **Trigger:** | User wants to login. |
| **Standard process:** | 1. The user will be asked to login. 2. The credentials will be matched with the one in the database. 3. If they matched the user will logged in . 4. Otherwise will be asked login again. |
| **Alternative process:** | None |

* 1. **Reliability** 
     1. *Maintenance*

The software will not allow multiple logins at a time. Only one login is permitted at a

time. Database is separate from the software . Also the software is developed

using concepts of object oriented programming which helped to keep

the structure of the software in a well defined manner and easy to understand .All

these factors made the maintenance of the software easy and fast.

* + 1. *Maximum bug rate*

There will be a maximum of 1 bug/KLOC.

* + 1. *Security Considerations*

The software is designed using object oriented concepts which makes sure that

important data (attributes) is kept private.

Only authorized employees will be able to login and use the software. Since database

is separated from software , in case of crash of software the data will remain intact.

* 1. **Performance Requirements**
     1. *Response time*

Since the software is developed using c++ programming language the software runs very

fast, that is in fraction of seconds. But the time increases linearly if the items in the menu or

number of receptionists increases. Also the maximum time to print/show any information

will not exceed 30 seconds.

* + 1. *Capacity*

The database can store as much data as the hard drive permits. Also since the menu

is stored in some stl the size should not exceed the maximum size of the stl .

* + 1. *Scalability*

The software is designed for a single food court which has a local database and no

connections with external networks. But there can be food courts which are

organized by a single firm which have multiple outlets. So apart from the local

database they should also have a one combined database so this software can be

scaled by doing appropriate changes in the code.

* 1. **Supportability**
     1. *Naming Convention*

All code is written as specified by the Hungarian Naming Convention. The variables are

defined in a such way that are easy to understand and one can easily predict the functions

that they form just by looking at the name.

* 1. **Design Constraints**
     1. *Software Language*

This software is developed using c++ programming language. The software is not

connected to any server it is hosted on a local system.

* 1. **Interfaces**
     1. *User Interfaces*

The software will first ask for the client to login according to the role of the user ,

that is either as a manager or as a receptionist . Then if the logged in person is a

receptionist then he/she can take a order from the customer or can view their personal

details .While taking an order if the receptionist does not enters the correct name of a

food item then he/she is shown a list of names that are closest to the name that was

entered by the receptionist.

If the logged in person is manager then he/she can change personal details, in addition to

that manager can also hire an employee or can remove an employee from the job

Also manager will have an option of viewing a monthly report if wishes to do so.

All the interaction of the client with the software is through keyboard and computer

screen ,that is client have to select the desired option through keyboard and then the

output will be shown on computer screen.

* + 1. *Hardware Interfaces*

The software will be executed on the computer and the input by the client will be given

through keyboard. Also one would require minimum of 1 Gb of Ram and 1Gb of storage for

the smooth functioning of the software.

* + 1. *Software Interfaces*

The software interacts with the operating system, also there is a c++ compiler for the

execution of the software . The software uses database which is stored in the local

computer . Also it uses certain math libraries.

1. **Supporting Information**

*Appendix (Definition)*

* 1. Actors – an actor in the uml specifies a role played by a user or any other system that interacts with the different type of use cases
  2. Attribute – an attribute is a specification that defines a property of an object, property or file
  3. Compiler – a program that converts computer instructions into a form that computer understands ,that is machine code so that it can be executed by the computer.
  4. Hardware – these are the physical parts of a computer.
  5. Hungarian naming convention – in this convention the name of the variables/processes/functions are named according to the work they do.

* 1. Interface – an interface is a common/shared boundary across which two or more than two computer system exchange any kind of information.
  2. KLOC – it is the measure of the size of the computer program.
  3. Manager – a person who manages the work and growth of an organization.
  4. Receptionist – a person who sits at the counter or at the entrance of an organization to greet people.
  5. Software – it is the part of the computer system that consist of computer instructions ,data ,programs etc.