***Write Ups***

**Movie Ticket Booking System**

**Abstract**

Internet based online movie ticket reservation system for cinema halls which can be accessed over internet. This web application will automate the reservation of tickets and enquiries about availability of the seats. This application includes email confirmation for the tickets. Users can subscribe to new release mailers.

**Users of the system**

Multiple profiles of users shall operate this system.

1. Administrator

2. Manager

3. External users

**High Level Requirements**

1. System to allow the administrator to maintain masters such as Cinema Halls, type details like multiplexes or single, number of screens, seating capacity per screen, approximate rates per screen, other facilities at the location and its map etc to be published.

2. The cinema halls listed are of different types and offer different types of seats/classes. Information like seating capacity, rates for different classes. Only 50% of the seats are available for online reservation.

3. System to allow the admin to upload the details of number of shows, movies being screened, ticket rates etc on the site only on registration.

4. System to allow users to search for cinema halls, movies, shows based on selection criteria, for 2 weeks in advance.

5. System to allow user to choose the seats which are available for a selected movie, selected screen, selected show, selected class. User can select the mode of transfer of tickets whether through the courier or collection at the counter, charges for the same to be added to the total of ticket charges.

6. User to subscribe for new release information.

7. System to mail information as per subscription.

.

**SYSTEM ANALYSIS**

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in [systems engineering](http://en.wikipedia.org/wiki/Systems_engineering), [information systems](http://en.wikipedia.org/wiki/Information_systems) and [software engineering](http://en.wikipedia.org/wiki/Software_engineering), is the process of creating or altering systems, and the models and [methodologies](http://en.wikipedia.org/wiki/Methodologies) that people use to develop these systems. In software engineering the SDLC concept underpins many kinds of [software development methodologies](http://en.wikipedia.org/wiki/Software_development_methodologies).

**IMPLEMENTATION**

**Introduction of Technologies Used**

Initially Java language was called as “oak” but it was renamed as “java” in 1995.The primary motivation of this language was the need for a platform-independent i.e. architecture neutral language that could be used to create software to be embedded in various consumer electronic devices.

**Applications and applets**

An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++ .Java’s ability to create Applets makes it important. An Applet I san application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet I actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can be react to the user input and dynamically change.

**Java Architecture**

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

**Compilation of code**

When we compile the code, the Java compiler creates machine code called byte code for a hypothetical machine called Java Virtual Machine (JVM). Compiling and interpreting java source code.

****

**Fig 4.1: Structure of compilation**

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

**Simple**

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer, learning Java will oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

### Object oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank state. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

### Robust

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks the code at compile time and runtime.

Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all run-time errors can and should be managed by your program.

#### SOFTWARE REQUIREMENT SPECIFICATION

#### 5.1 Requirements Specification:

#### Requirement Specification provides a high secure storage to the web server efficiently. Software requirements deal with software and hardware resources that need to be installed on a serve which provides optimal functioning for the application. These software and hardware requirements need to be installed before the packages are installed. These are the most common set of requirements defined by any operation system. These software and hardware requirements provide a compatible support to the operation system in developing an application.

#### 5.1.1 HARDWARE REQUIREMENTS:

#### The hardware requirement specifies each interface of the software elements and the hardware elements of the system. These hardware requirements include configuration characteristics.

#### System : Pentium IV 2.4 GHz.

#### Hard Disk : 100 GB.

#### Monitor : 15 VGA Color.

#### Mouse : Logitech.

#### RAM : 1 GB.

#### 5.1.2 SOFTWARE REQUIREMENTS:

#### The software requirements specify the use of all required software products like data management system. The required software product specifies the numbers and version. Each interface specifies the purpose of the interfacing software as related to this software product.

#### Operating system : Windows XP/7/10

* Coding Language : Html, JavaScript, Java/J2EE (Jsp Servlet)
* Development Kit : JDK 1.7
* Database : MySQL
* IDE : Netbeans
* Server : Tomcat 7.0

#### 5.2 FUNCTIONAL REQUIREMENTS:

The functional requirement refers to the system needs in an exceedingly computer code engineering method.

The key goal of determinant “functional requirements” in an exceedingly product style and implementation is to capture the desired behavior of a software package in terms of practicality and also the technology implementation of the business processes.

#### 5.3 NON FUNCTIONAL REQUIREMENTS

All the other requirements which do not form a part of the above specification are categorized as Non-Functional needs. A system perhaps needed to gift the user with a show of the quantity of records during info. If the quantity must be updated in real time, the system architects should make sure that the system is capable of change the displayed record count at intervals associate tolerably short interval of the quantity of records dynamic. Comfortable network information measure may additionally be a non-functional requirement of a system.

The following are the features:

* Accessibility
* Availability
* Backup
* Certification
* Compliance
* Configuration Management
* Documentation
* Disaster Recovery
* Efficiency(resource consumption for given load)
* Interoperability

**5.4 PERFORMANCE** **REQUIREMENTS**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system
* The system should be accurate
* The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

**5.5 Feasibility Study:**

Preliminary investigation examines project feasibility; the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operation Feasibility

Economical Feasibility

**5.5.1 Technical Feasibility**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

* Does the necessary technology exist to do what is suggested?
* Do the proposed equipments have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

**5.5.2 Operational Feasibility**

**User-friendly**

Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

**Reliability**

The package wills pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

**Security**

The web server and database server should be protected from hacking, virus etc

**Portability**

The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

**Availability**

This software will be available always.

**Maintainability**

The system uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end.

The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

**5.5.3 Economic Feasibility**

The computerized system takes care of the present existing system’s data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports.

It should be built as a web based application with separate web server and database server. This is required as the activities are spread throughout the organization customer wants a centralized database. Further some of the linked transactions take place in different locations.

Open source software like TOMCAT, JAVA, Mysql and Linux is used to minimize the cost for the Customer.

**Methodology**

**SDLC (Software Development Life Cycle) – Umbrella Model**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

**Fig no. 6.1 Umbrella model**

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Requirements Gathering Stage**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



**Fig no. 6.2 Requirements Gathering stage**

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are not included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

Feasibility study is all about identification of problems in a project, number of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.

Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



**Fig no. 6.3 Analysis stage**

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



**Fig no. 6.4 Designing stage**

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



**Fig no. 6.5 Coding stage**

**Integration & Test Stage**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

**Fig no. 6.6 Integration and Testing Stage**

**Installation & Acceptance Test**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



**Fig no. 6.7 Installation**

**Maintenance**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category.

## SYSTEM DESIGN

## 7.1 SYSTEM ARCHITECTURE

## The purpose of the design phase is to arrange an answer of the matter such as by the necessity document. This part is that the opening moves in moving the matter domain to the answer domain. The design phase satisfies the requirements of the system. The design of a system is probably the foremost crucial issue warm heartedness the standard of the software package. It’s a serious impact on the later part, notably testing and maintenance.

## The output of this part is that the style of the document. This document is analogous to a blueprint of answer and is employed later throughout implementation, testing and maintenance. The design activity is commonly divided into 2 separate phases System Design and Detailed Design.

## System Design conjointly referred to as top-ranking style aims to spot the modules that ought to be within the system, the specifications of those modules, and the way them move with one another to supply the specified results.

## At the top of the system style all the main knowledge structures, file formats, output formats, and also the major modules within the system and their specifications square measure set. System design is that the method or art of process the design, components, modules, interfaces, and knowledge for a system to satisfy such as needs. Users will read it because the application of systems theory to development.

## Detailed Design, the inner logic of every of the modules laid out in system design is determined. Throughout this part, the small print of the info of a module square measure sometimes laid out in a high-level style description language that is freelance of the target language within which the software package can eventually be enforced.

## In system design the main target is on distinguishing the modules, whereas throughout careful style the main target is on planning the logic for every of the modules.

UML Diagrams

Unified Modeling Language (UML) is a standard language for writing software blueprints. UML can be used for visualizing, specifying, constructing, documenting the artifacts of a software-intensive system. The Unified Modeling Language will result in lower overall costs, more reliable and efficient software, and a better relationship with all parties involved.  Software documented with UML can be modified much more efficiently**.**

UML is a notation that resulted from the unification of Object Modeling Technique and Object Oriented Software Technology .UML has been designed for broad range of application. Hence, it provides constructs for a broad range of systems and activities.

1. Use case diagrams

Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from the external point of view. The actors are outside the boundary of the system, whereas the use cases are inside the boundary of the system.

2. Class diagrams

Class diagrams to describe the structure of the system. Classes are abstractions that specify the common structure and behavior of a set Class diagrams describe the system in terms of objects, classes, attributes, operations and their associations.

3. Sequence diagrams

Sequence diagrams are used to formalize the behavior of the system and to visualize the communication among objects. They are useful for identifying additional objects that participate in the use cases. A Sequence diagram represents the interaction that take place among these objects.

4. State Chart diagrams

State chart diagrams describe the behavior of an individual object as a number of states and transitions between these states. A state represents a particular set of values for an object. The sequence diagram focuses on the messages exchanged between objects, the state chart diagrams focuses on the transition between states.

5. Activity diagrams

An activity diagram describes a system in terms of activities. Activities are statesthat represents the execution of a set of operations. Activity diagrams are similar to flowchart diagram and data flow

**Use case Diagram**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Use case Diagram:

****

Figure 4.4 Use case Diagram

**Sequence diagram**

UML sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes.  Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.

The following sequences of steps are involved in the system.

* The user has to login the website.
* If the user has login valid then he can process the data.
* Generate Sessions
* Apply clustering on generated sessions data sets.
* Finally we result from the clustering data.



Figure 4.6 **Fig: Sequence diagram for User**

****

**Fig: Sequence diagram for Managers**

****

**Fig: Sequence diagram for Admin**

**Class Diagram:**

****

Figure 4.4 Class Diagram

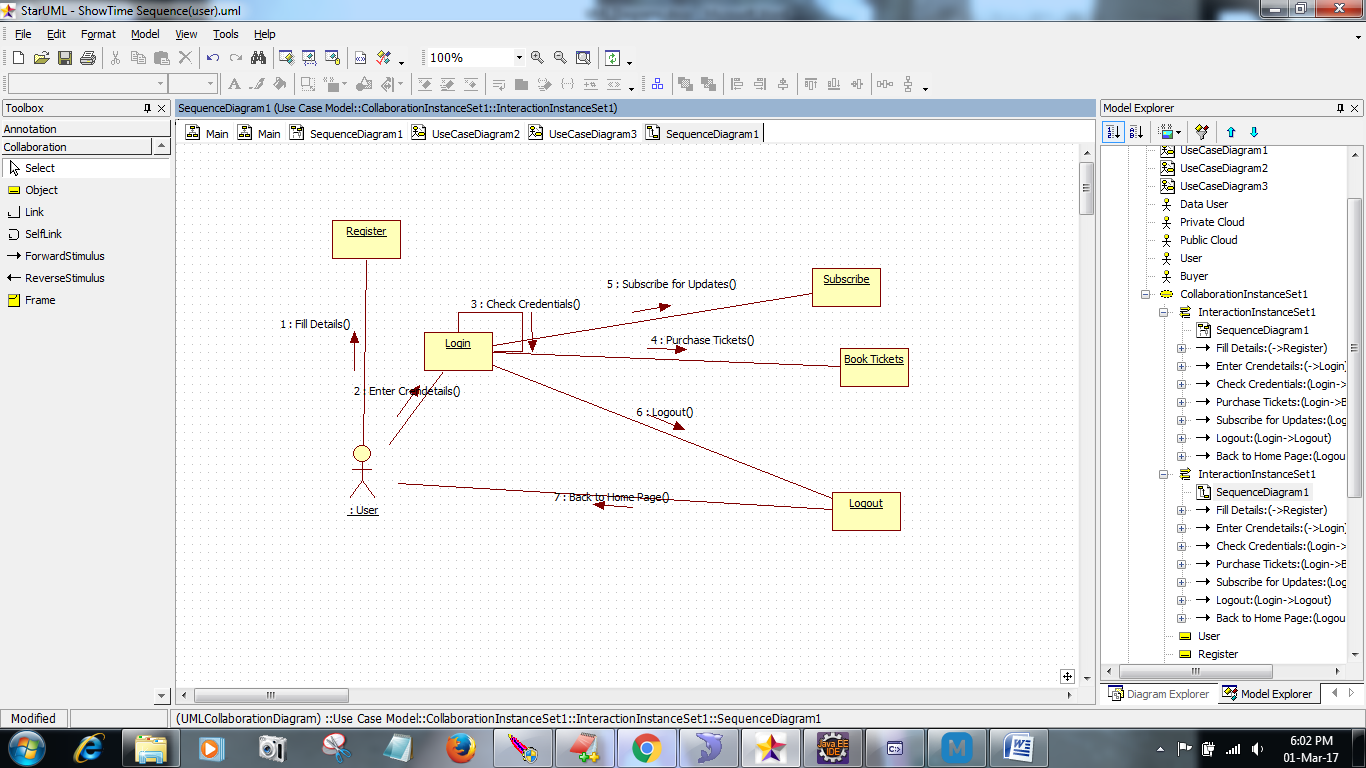
**4.6.6Activity diagram**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

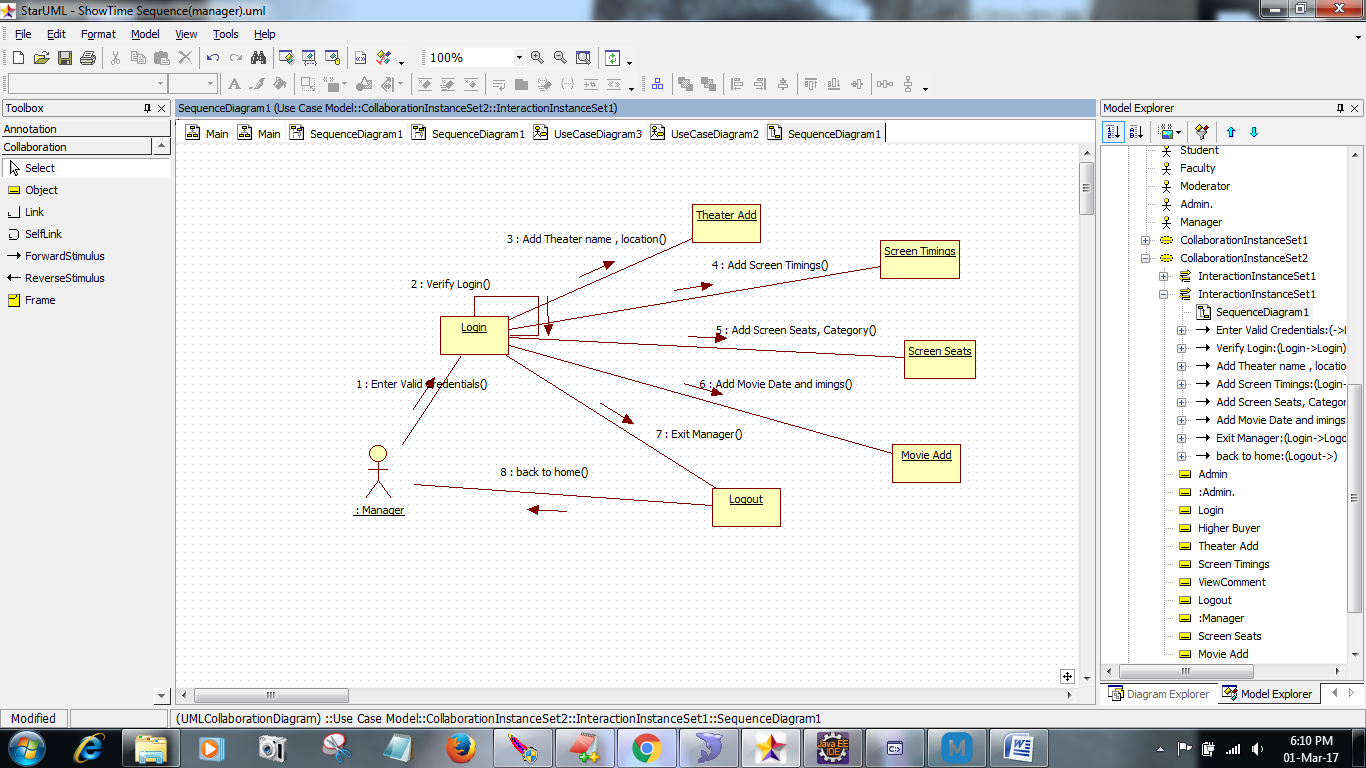


Figure 4.9 Activity Diagram

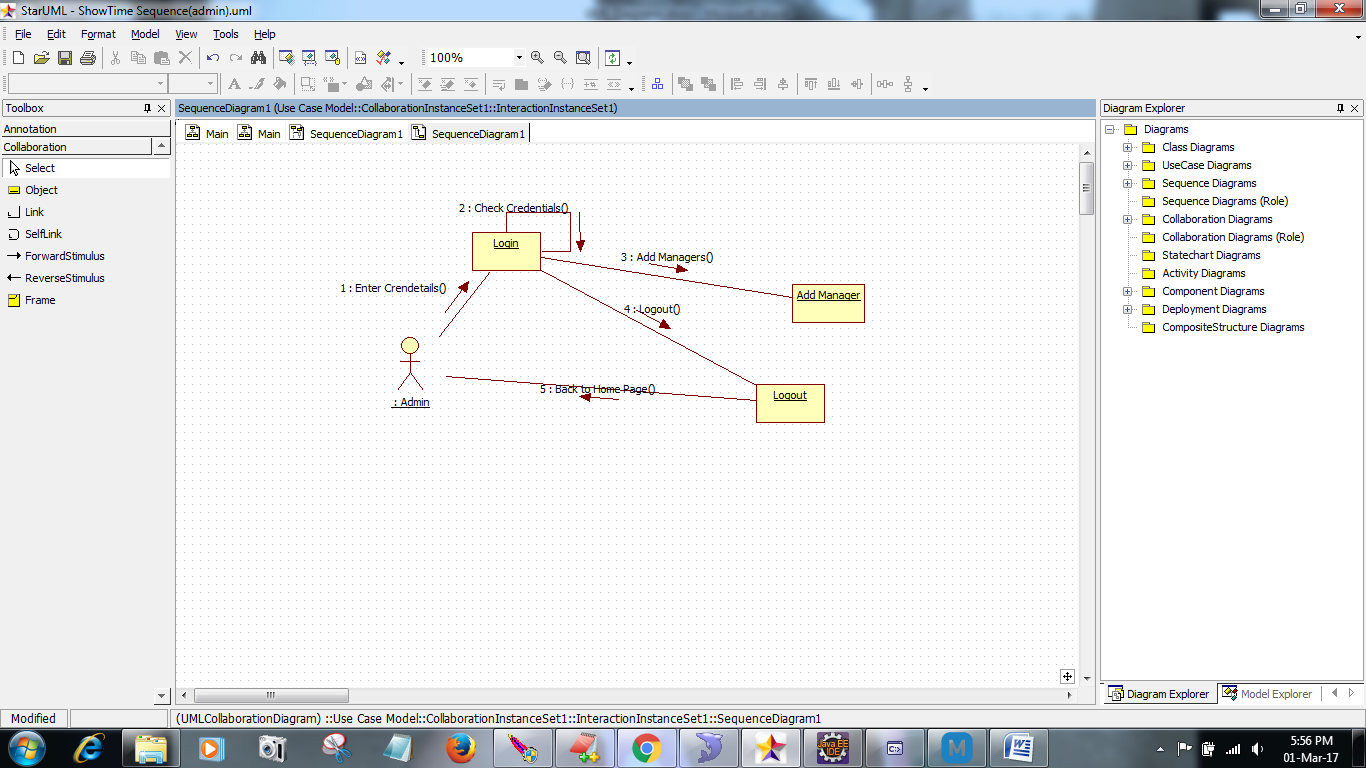
**Collaboration Diagrams:**

****

**Fig: Collaboration diagram for User**

****

**Fig: Collaboration diagram for Manager**

****

**Fig: Collaboration diagram for Admin**

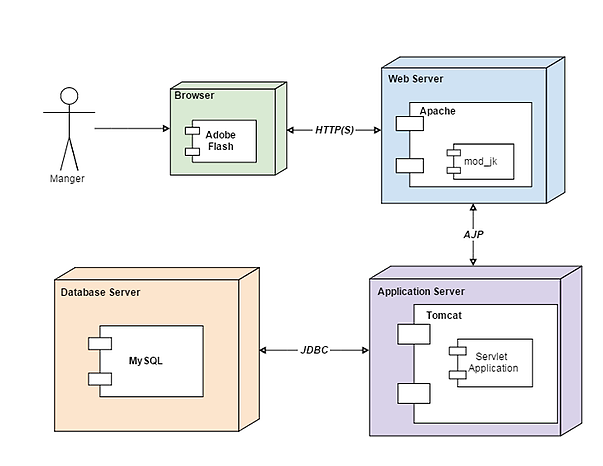
**StateChart Diagram**

****

**Component Diagram**

****

**Deployment Diagram**

****

**E-R Diagram**

****

**8. TESTING**

## Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. The following is the description of the testing strategies, which were carried out during the testing period.

## 8.1 SYSTEM TESTING

## Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to user the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

## 8.2 MODULE TESTING

## To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

## 8.3 INTEGRATION TESTING

## After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system

## 8.4 ACCEPTANCE TESTING

## When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

**8.5 TEST CASES:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | Upload the tasks dataset | Verify either file is loaded or not | If dataset is not uploaded | It cannot display the file loaded message | File is loaded which displays task waiting time | High | High |
| 02 | Upload patients dataset | Verify either dataset loaded or not | If dataset is not uploaded | It cannot display dataset reading process completed | It can display dataset reading process completed | low | High |
| 03 | Preprocessing | Whether preprocessing on the dataset applied or not | If not applied | It cannot  display the necessary data for further process | It can display the necessary data for further process | Medium | High |
| 04 | Prediction Random Forest | Whether  Prediction algorithm applied on the data or not | If not applied | Random tree is not generated | Random tree is generated | High | High |
| 05 | Recommendation | Whether predicted data is displayed or not | If not displayed | It cannot view prediction containing patient data | It can view prediction containing patient data | High | High |
| 06 | Noisy Records Chart | Whether the graph is displayed or not | If graph is not displayed | It does not show the variations in between clean and noisy records | It shows the variations in between clean and noisy records | Low | Medium |

TABLE 8.5.1 TESTCASES