**Part 8**

**Project Documentation**

**P8.1 INTRODUCTION**

P1.1 Project Overview

Universal Cinemas will be a simple platform for users to book cinema tickets. It provides the services to the users who are searching for movie tickets. This is a website in which we can book tickets for the movies in the theatre who are registered in this website and can make payment for the ticket through online. Once the user register, he/she can book the tickets using this website. This system provides the theatre owners registration too. After the registration, the theater owners can login and can add shows to their theatre, view profile and view films added by the admin.

**P1.2 PROJECT SPECIFICATION**

1. Theatre owner Module

The theater owners can manage the shows in his theater he can add shows, add theater details, and add seat details, online seat availability and generating ticket.

Thus the main features of the project are that

* Booking movie ticket Facility
* Add new theaters
* Facility to check available movie on your locality

**P8.2 SYSTEM STUDY**

P8.2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute’s detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analysing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

**P8.2.2 PROPOSED SYSTEM**

Universal Cinemas will be a simple platform for users to book cinema tickets. It provides the services to the users who are searching for movie tickets. This is a website in which we can book tickets for the movies in the theatre who are registered in this website and can make payment for the ticket through online. Once the user register, he/she can book the tickets using this website. This system provides the theatre owners registration too. After the registration, the theater owners can login and can add shows to their theatre, view profile and view films added by the admin.

The main features include:

* + Registered users can search for theaters and films.
  + Registered users can book ticket for any movie and pay bill using online payment gateways

**P8.2.3 ADVANTAGES OF PROPOSED SYSTEM**

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

* **Better security: -**

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

* **Ensure data accuracy: -**

The proposed system eliminates the manual errors while entering the details of the users during the registration.

* **Better service: -**

The product will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for longer period with no loss of data.

* **User friendliness and interactive: -**

The proposed system will help the user to reduce the workload and provides user friendly environment so that they can easily do their jobs. The system alerts the users for each activity to be carried out, through notification.

* **Minimum time required: -**

The data are management is in such a way that a particular registered user can search service provider very easily.

**P8.3 REQUIREMENT ANALYSIS**

#### P8.3.1 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

**P8.3.1.1 Economical Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

* The costs conduct a full system investigation.
* The cost of the hardware and software.
* The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

Universal Cinemas will be a simple platform for users to access services for their huge needs. It is completely free. Using this system large number people can solve their problems with free of cost.There are website like bookmyshow which give information theatre.In that system. At the same time UNIVERSAL CINEMAS is completely free for registration and book tickets.

**P8.3.1.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

* Does the existing technology sufficient for the suggested one?
* Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using php in front end and MySql in server in back end, the project is technically feasible for development.

**P8.3.1.3 Behavioural Feasibility**

This includes the following questions:

* Is there sufficient support for the users?
* Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioural aspects are considered carefully and conclude that the project is behaviourally feasible.

Universal Cinemas, GUI is simple so that users can easily use it. UNIVERSAL CINEMAS is simple enough so that no training is needed.

### **P8.4 REQUIREMENT MODELING**

P8.4.1 UML Use Cae Diagram

**Use case diagrams** are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more **external users** of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

Note, that UML 2.0 to 2.4 specifications also described **use case diagram** as a specialization of a class diagram, and class diagram is a structure diagram.

Use case diagrams are in fact twofold - they are both behavior diagrams, because they describe behavior of the system, and they are also structure diagrams - as a special case of class diagrams where classifiers are restricted to be either actors or use cases related to each other with associations.

Use case diagrams are used to specify:

* (external) requirements, required usages of a system under design or analysis (subject) - to capture what the system is supposed to do;
* the functionality offered by a subject – what the system can do;
* requirements the specified subject poses on its environment - by defining how environment should interact with the subject so that it will be able to perform its services.



**Figure 5 : Usecase Diagram of UNIVERSAL CINEMAS**

### P8.4.2 SEQUENCE DIAGRAM

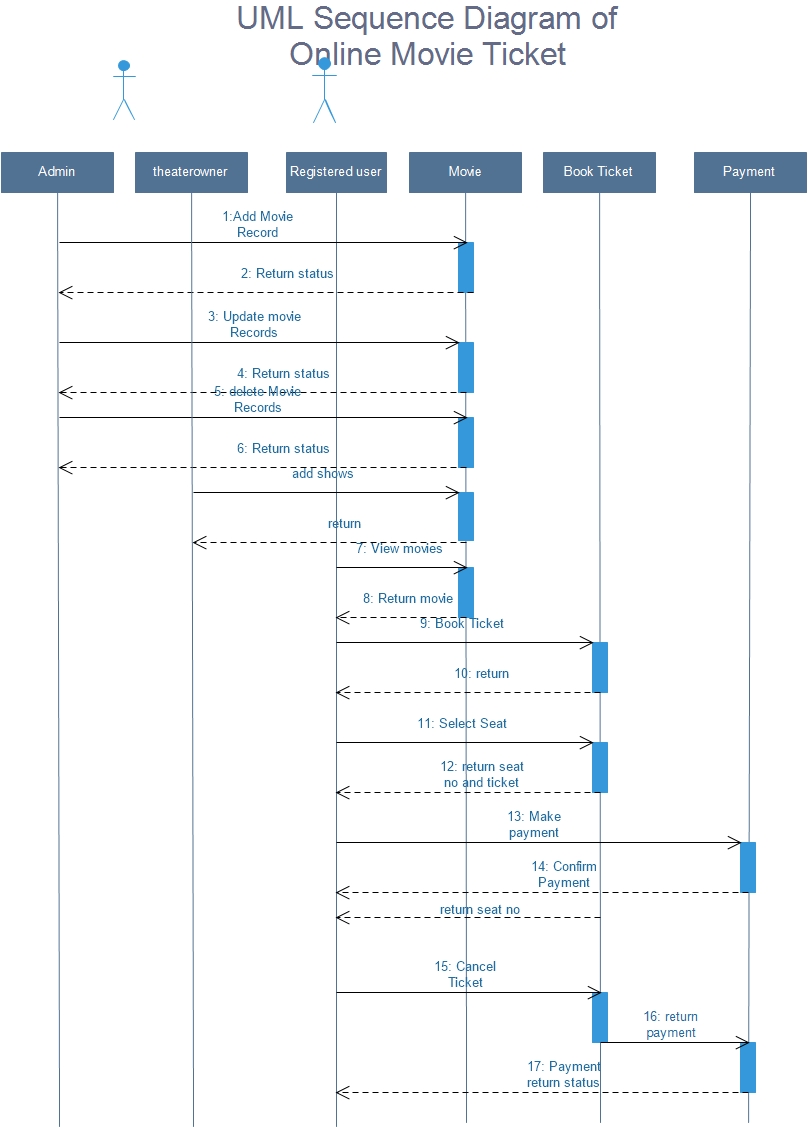
These diagrams are used by software developers and business people alike to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

## Sequence Diagram Applications

Sequence diagrams can be useful reference diagrams for businesses and other organizations. Try drawing a sequence diagram to:

* Represent the details of a UML use case.
* Model the logic of a sophisticated procedure, function, or operation.
* See how tasks are moved between objects or components of a process.

Plan and understand the detailed functionality of an existing or future scenario

****

**Figure 6 : Sequence diagram of UNIVERSAL CINEMAS**

#### **P8.5 SYSTEM SPECIFICATION**

**P8.5.1 Hardware Specification**

Processor - Pentium IV/AMD Dual core

RAM - 1 GB

Hard disk - 500 GB

**P8.5.2 Software Specification**

Front End - PHP

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, PHP, CSS

#### **P8.6 SOFTWARE DESCRIPTION**

**P8.6.1 PHP**

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994,the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for Personal Home Page,but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML or HTML5 markup, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification

**FEATURES**

It is most popular and frequently used worldwide scripting language, the main reason of popularity is; It is open source and very simple.

* Simple
* Faster
* Interpreted
* Open Source
* Case Sensitive
* Simplicity
* Efficiency
* Platform Independent
* Security
* Flexibility
* Familiarity

### Simple

It is very simple and easy to use, compare to other scripting language it is very simple and easy, this is widely used all over the world.

### Interpreted

It is an interpreted language, i.e. there is no need for compilation.

### Faster

It is faster than other scripting language e.g. asp and jsp.

### Open Source

Open source means you no need to pay for use php, you can free download and use.

### Platform Independent

PHP code will be run on every platform, Linux, Unix, Mac OS X, Windows.

### Case Sensitive

PHP is case sensitive scripting language at time of variable declaration. In PHP, all keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are NOT case-sensitive.

###### **P8.6.2 MySQL**

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

The MySQL Web site provides the latest information about MySQL software.

* **MySQL is a database management system.**
  1. database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.
* **MySQL databases are relational.**
  1. relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard.

We use the phrase“the SQL standard” to mean the current version of the SQL

Standard at any time.

* **MySQL software is Open Source*.***

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

* **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

* **MySQL Server works in client/server or embedded systems*.***

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

* **A large amount of contributed MySQL software is available*.***

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

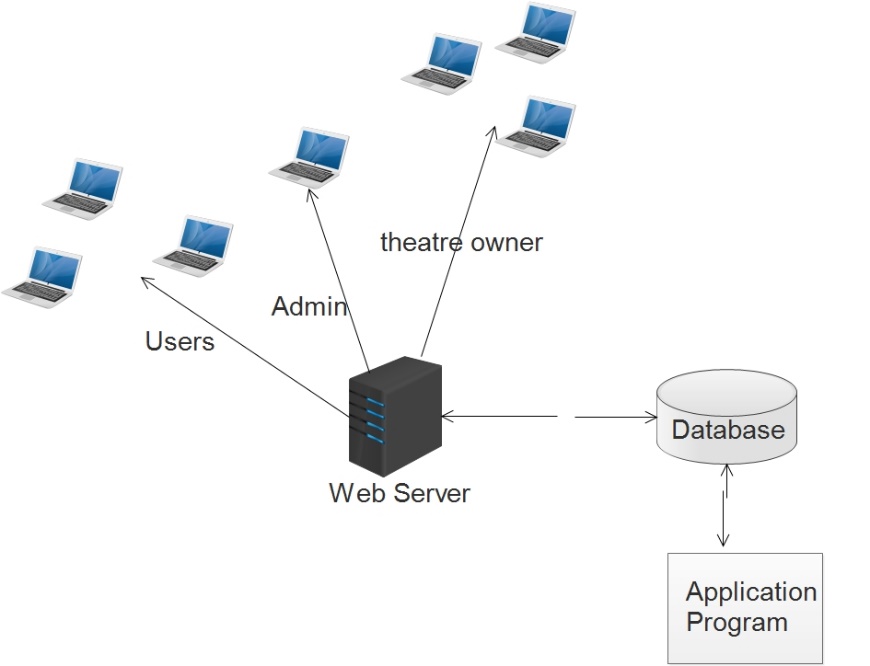
**P8.7 SYSTEM DESIGN**

## INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

#### **P8.7.1 ARCHITECTURAL DESIGN**

This section describes the components of Universal Cinemas.

****

**Figure 1: Architectural Design**

The registered user, admin, service provider can accesses the UNIVERSAL CINEMAS through internet using their Laptop, Smart Phone, Tablet or Desktop Computer. The System’s application program processes the user’s request and provides the required services by taking data from the system database.

#### **P8.7.2 MODULE DESIGN**

#### Admin Module

The admin is the overall controller of the system. The admin can add, edit, update, delete movies, theatre, theatre owners, categories, place. Also admin have the power to block and add theatre owners and registered users.

|  |  |
| --- | --- |
| Manage theater owners, add film, categories, place and language. | Block/Activate the registered users and theatre owners |

**Registered User Module**

After registration, user can search for a films,theatres, view his profile, book movie tickets avilable.

|  |  |
| --- | --- |
| User registration, login | Search movies |
| Booking movie tickets | Manage registration details |

**Theatre Owner Module**

After registration, theatre owners can view booking details, add shows, add online seats.

|  |  |
| --- | --- |
| theatre Registration ,OTP verification and login | Can add, edit, update profile details |
| Manage booking details. | Print report of all booking details |

#### **P8.7.3 DATA FLOW DIAGRAM**

Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. It’s a structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process oriented system flowcharts. When analysts prepare the Data Flow Diagram, they specify the user needs at a level of detail that virtually determines the information flow into and out of the system and the required data resources. This network is constructed by using a set of symbols that do not imply physical implementations. The Data Flow Diagram reviews the current physical system, prepares input and output specification, specifies the implementation plan etc.

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have some reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout the project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In our opinion, “efficient design of the data flow and context flow diagrams helps to design the system successfully without much major flaws within the scheduled time”. This is the most complicated part in a project. In the designing process, our project took more than the activities in the software life cycle. If we design a system efficiently with all the future enhancements, the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as way for expressing system requirements in graphical form. A data flow diagram also known as “bubble chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionally decomposes the requirement specification down to the lowest level. DFD depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

**Main symbols used in the data flow diagram are:**

1. Circle represents a process that transforms incoming data flows in to outgoing data flows.
2. A square defines a source and destination of system data.
3. Arrow identifies data in motion.
4. An open rectangle defines a data store, data at rest or temporary repository of data.

**Steps to Construct Data Flow Diagrams:-**

Four steps are commonly used to construct a DFD:

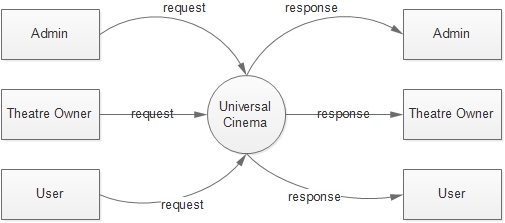
* + Process should be named and numbered for easy reference. Each name should be representative of the process.
  + The destination of flow is from top to bottom and from left to right.
  + When a process is exploded in to lower level details they are numbered.
  + The names of data stores, sources and destinations are written in capital letters.

**Rules for constructing a Data Flow Diagram**

* + Arrows should not cross each other.
  + Squares, circles and files must bear names.
  + Decomposed data flow squares and circles can have same names.
  + Draw all data flow around the outside of the diagram.

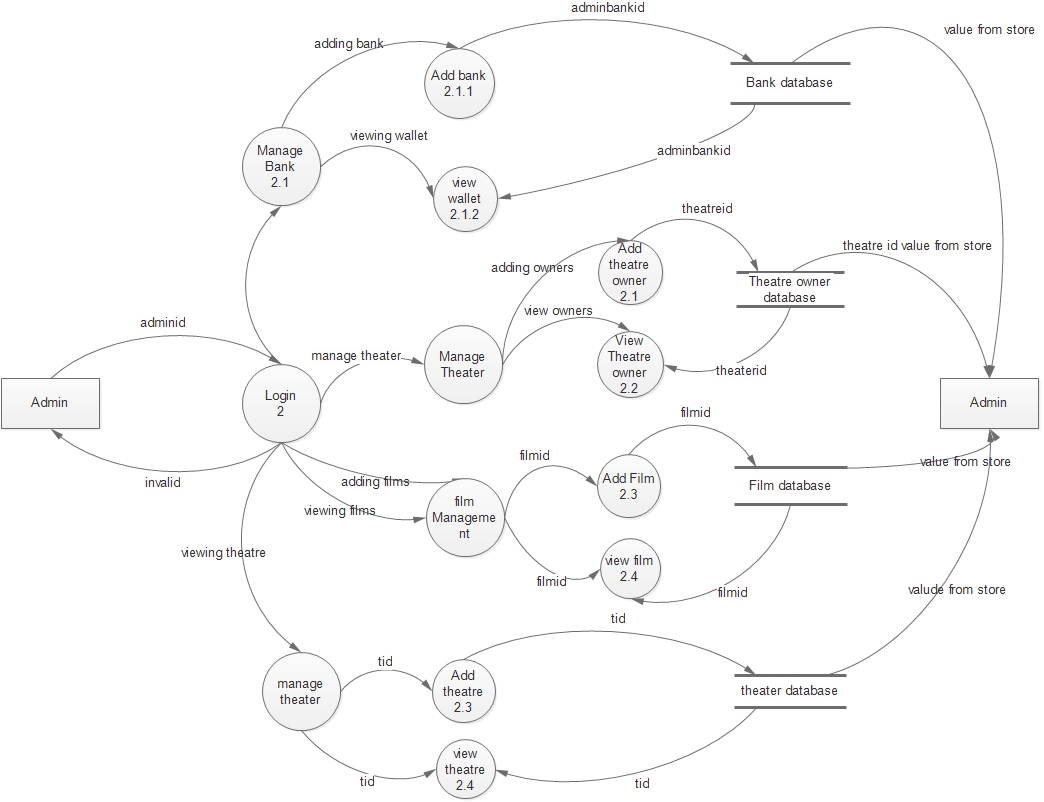
**Data Flow Diagrams of Universal Cinemas**

**Level -0**

****

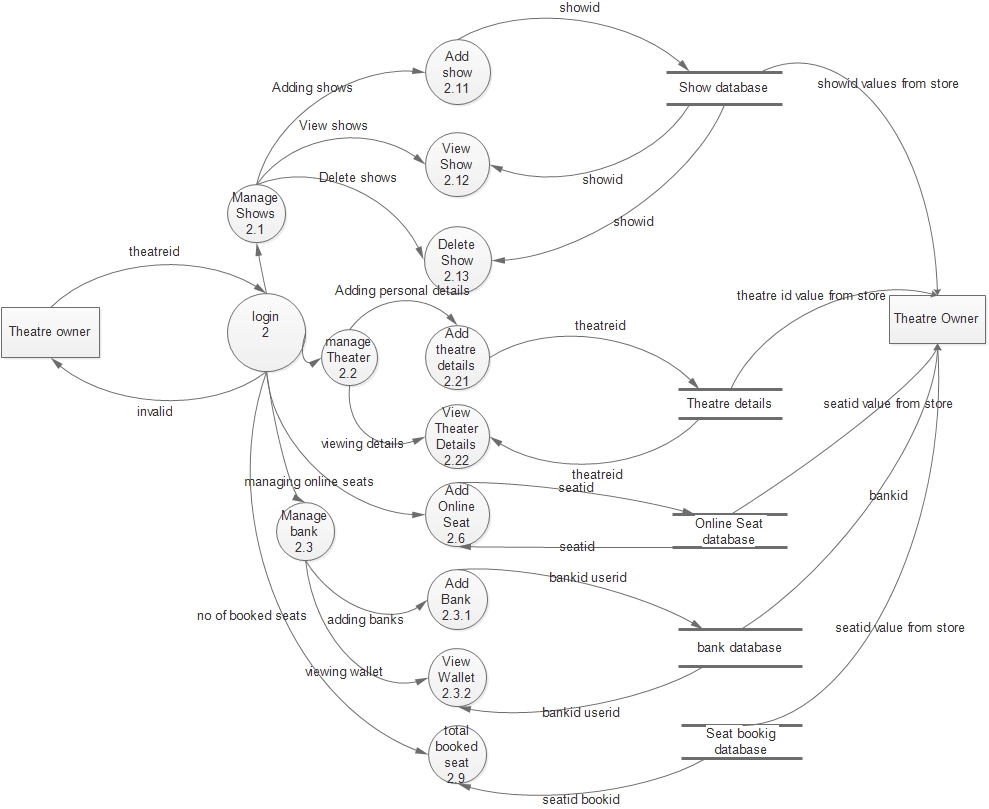
**Figure 2 : Level 0 data flow diagram**

**Admin Level-1**

****

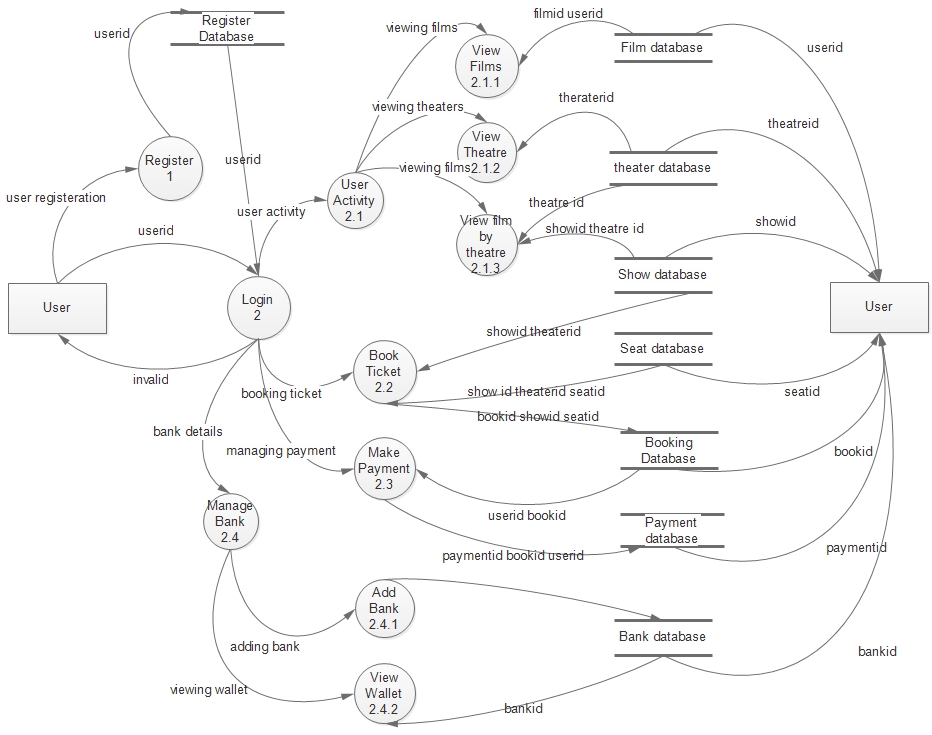
**Figure 3 : Level 1 data flow diagram for admin**

**Theatre Owner Level 1**

****

**Figure 4 : Level 1 data flow diagram for TheaterOwner**

**Registered User Level-1**

****

**Figure 5 : Level 1 data flow diagram for Registered user**

#### **P8.7.4 Database Design**

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

* Data Integrity
* Data independence

**Relational Database Management System (RDBMS)**

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

**Relations, Domains & Attributes**

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

**Relationships**

* Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
* Entity Integrity enforces that no Primary Key can have null values.
* Referential Integrity enforces that no Primary Key can have null values.
* Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

**Normalization**

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

* Normalize the data.
* Choose proper names for the tables and columns.
* Choose the proper name for the data.

**First Normal Form**

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words 1NF disallows “relations within relations” or

“relations as attribute values within tuples”. The only attribute values permitted by

1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data.A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

**Second Normal Form**

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

**Third Normal Form**

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

**TABLES**

**Table No 1 :**

**THEATER OWNER REG TABLE**

PRIMARY KEY:ADMIN\_ID

FOREIGN KEY: THEATER\_ID

This table is used to register the details of the theater owners and staff members

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELDNAME | DATATYPE | DESCRIPTION |
| 1 | ADMIN\_ID | NUMBER | Admin login id unique |
| 2 | USERNAME(EMAIL) | VARCHAR | Username of the theater owner and staff |
| 3 | PASSWORD | VARCHAR | Password |
| 4 | FIRST\_NAME | VARCHAR | First name |
| 5 | LAST\_NAME | VARCHAR | Last name |
| 6 | AGE | NUMBER | Age |
| 7 | CONTACT | NUMBER | Contact no |
| 8 | ADDRESS | VARCHAR | Address for communication |
| 9 | ADMIN TYPE | VARCHAR | Theater owner/staff |
| 10 | THEATER\_ID | NUMBER | Theater id |

**Table No:2**

**USER REGISTERATION**

PRIMARY KEY: USER\_ID

This table is used to register the details of the user`

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | USER\_ID | NUMBER | Userid od user |
| 2 | USERNAME(EMAIL) | VARCHAR | Username |
| 3 | PASSWORD | VARCHAR | Password |
| 4 | FIRSTNAME | VARCHAR | Firstname |
| 5 | LASTNAME | VARCHAR | Lastname |
| 6 | AGE | NUMBER | Age |
| 7 | SEX | VARCHAR | Gender |
| 8 | ADDRESS | VARCHAR | Address |
| 9 | CONTACT | NUMBER | Contact no |

**Table No:3**

THEATER REG

PRIMARY KEY: THEATER ID

This table is handled by super Admin It is use to register the details of theater owners

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | THEATER\_ID | NUMBER | Theater id |
| 2 | THEATER\_NAME | VARCHAR | Theatername |
| 3 | LOCATION | VARCHAR | Location of the theater |
| 4 | ADDRESS | VARCHAR | Address of the theater |
| 5 | CONTACT | NUMBER | Contact no of theater |
| 6 | EMAIL | VARCHAR | Email of the theater |

**Table No: 4**

SCREEN DETAILS

PRIMARY KEY: SCREEN\_ID

FOREIGN KEY: THEATER\_ID comes from theater registeration

FOREIGN KEY: THEATER\_ID

This table is used to register the details of theater

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | SCREEN\_ID | NUMBER | Screen id |
| 2 | THEATER\_ID | NUMBER | Theaterid |
| 3 | THEATER\_MODE | VARCHAR | Mode of theater whether 3k or 4k |
| 4 | NO\_OF\_SEAT | NUMBER | Total no of seats |

**Table No:5**

FILM REG

PRIMARY KEY: FILM\_ID

This table is used to add film details it is handled by super admin

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATATYPE | DESCRIPTION |
| 1 | FILM\_ID | NUMBER | Film id |
| 2 | FILM NAME | VARCHAR | Film name |
| 3 | ACTOR NAME | VARCHAR | Actor name |
| 4 | ACTRESS NAME | VARCHAR | Actress name |
| 5 | PRODUCER | VARCHAR | Producers name |
| 6 | SCRIPT WRITTER | VARCHAR | Script writer name |
| 7 | CATEGORY | VARCHAR | Category of name |

**Table No: 6**

SHOW REG

PRIMARY KEY: SHOW\_ID

FOREIGN KEY: SCREEN\_ID comes from screen reg table

FOREIGN KEY: FILM\_ID comes from film reg table

This table is register the show details to a theater it is done by the theater owner

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATATYPE | DESCRIPTION |
| 1 | SHOW\_ID | NUMBER | Showid |
| 2 | SCREEN\_ID | NUMBER | Screen id |
| 3 | FILM\_ID | NUMBER | Film id |
| 4 | SHOW TIME | VARCHAR | Show times |

**Table No:7**

BANK DETAILS

PRIMARY KEY: BANK\_ID

FOREIGN KEY: LOGID

This table contain the details of the the bank

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | BANK\_ID | NUMBER | BANK ID |
| 2 | BANK NAME | VARCHAR | BANK NAME |
| 3 | IFSC CODE | VARCHAR | IFSC CODE OF THE BANK |
| 4 | BRANCH NAME | NUMBER | BRANCH NAME OF THE OFFICE |
| 5 | LOGINID | NUMBER | LOGIN ID OF THE USER |
| 6 | BALANCE | NUMBER | ACCOUNT BALANCE |
| 7 | STATUS | NUMBER | ACTIVE OR NOT |

**Table No: 8**

FEEDBACK TABLE

PRIMARY KEY: FED\_ID

FOREIGN KEY: LID

This table is used to enter the feedback it is used by customer

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELDNAME | DATATYPE | DESCRIPTION |
| 1 | FEED\_ID | NUMBER(10) | FEEDBACK ID |
| 2 | LID | VARCHAR(40) | EMAIL ID OF THE CUSTOMER |
| 3 | REASON | VARCHAR(1000) | REASON |

**Table No:9**

ADD PLACE

PRIMARY KEY: COUNTRY\_ID

This table contain the details of the the bank

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | PLACE\_ID | NUMBER | PLACE ID |
| 2 | PLACE\_NAME | VARCHAR | PLACE NAME |

**Table No: 10**

ADD LANGUAGE

PRIMARY KEY: LID

This table contain the details of the the Language

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | LID | NUMBER | LANGUAGE ID |
| 2 | LANGUAGE | VARCHAR | LANGUAGE |

**Table No: 11**

ADD CATEGORY

PRIMARY KEY: CAT\_ID

This table contain the details of the the Category of films

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | CAT-ID | NUMBER | CATEGORY\_ID |
| 2 | CATNAME | VARCHAR | CATEGORY NAME |

**Table No: 12**

LOGIN

PRIMARY KEY: LID

This table contain the details of the the bank

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | LID | NUMBER | LOGIN ID |
| 2 | USERNAME | VARCHAR | USERNAME OF THE USERS |
| 3 | PASSWORD | VARCHAR | PASSWORD OF THE USERS |
| 4 | STATUS | NUMBER | CHECK THERE THE USER IS LOGED IN OR NOT |
| 5 | ROLE | NUMBER | KNOW WHETHER USER, THATRE OWNER OR ADMIN |

Table No 13

RATING

PRIMARY KEY: RID

FOREIGN KEY: FID ,USERID

This table contain the details of rating movie

|  |  |  |  |
| --- | --- | --- | --- |
| SLNO | FIELD NAME | DATA TYPE | DESCRIPTION |
| 1 | RID | NUMBER | RATING ID |
| 2 | MSG | VARCHAR | AWARD GIVEN BY USER |
| 3 | RATE | VARCHAR | RATING OF THE FILM |
| 4 | FID | NUMBER | FILM ID |
| 5 | USERID | NUMBER | USER ID |

# P8.8 SYSTEM TESTING

#### 5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Validation **:** Are we doing the right job? Verification **:** Are we doing the job right?

Software testing should not be confused with debugging. Debugging is the process of analysing and localizing bugs when software does not behave as expected. Although the identification of some bugs will be obvious from playing with the software, a methodical approach to software testing is a much more thorough means for identifying bugs. Debugging is therefore an activity which supports testing, but cannot replace testing.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behaviour of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

* A good test case is one that has high possibility of finding an undiscovered error.
* A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

* For correctness
* For implementation efficiency
* For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

#### P8.8.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

* Unit testing
* Integration Testing
* Data validation Testing
* Output Testing

**P8.8.2.1 Unit Testing**

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm’s execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

**P8.8.2.2 Integration Testing**

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop.

After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

**P8.8.2.3 Validation Testing or System Testing**

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

**P8.8.2.4 Output Testing or User Acceptance Testing**

The system considered is tested for user acceptance; here it should satisfy the firm’s need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

* Input Screen Designs,
* Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

**P8.9 IMPLEMENTATION**

#### **INTRODUCTION**

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

* Careful planning.
* Investigation of system and constraints.
* Design of methods to achieve the changeover.
* Training of the staff in the changeover phase.

#### P8.9.1 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

* The active user must be aware of the benefits of using the new system.
* Their confidence in the software is built up.
* Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won’t take place.

**P8.9.2 User Training**

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

**P8.9.3 Training on the Application Software**

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

**P8.9.4 System Maintenance**

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

**P8.10 CONCLUSION AND FUTURE SCOPE**

#### P8.10.1 FUTURE ENHANCEMENT

* The system is designed in such a way that the payment can be returned when the user cancel the ticket

#### P8.10.2 CONCLUSION

The software reduces the time consumption and the manual efforts of searching for movie and booking ticket for movie finding theaters.

The benefits, we can obtain from the new system are:

* Timely and accurate information will be available
* Reduced data loss
* The access time and process time is highly reduced
* Quick data view
* Error free output

The proposed system is expected to replace manual system and provide more efficient performance and services.

**P8.11 BIBLIOGRAPHY**

**BOOKS/REFERENCES:**

* Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
* Roger S Pressman, “*Software Engineering*”, 1994.
* PankajJalote, “So*ftware engineering*: a precise approach”, 2006.
* James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
* IEEE Std 1016 Recommended Practice for Software Design Descriptions.

**WEBSITES:**

* www.w3schools.com
* www.jquery.com
* http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
* www.agilemodeling.com/artifacts/useCaseDiagram.htm

**P8.12 APPENDIX**

#### **P8.13Sample Code**

**User Reg**

<!doctype html>

<?php

include 'connection.php';

if(isset($\_POST['submit']))

{

$a=$\_POST["email"];

echo $a;

$b=$\_POST["pass"];

echo $b;

$g=$\_POST["cpass"];

$c=$\_POST["fname"];

echo $c;

$d=$\_POST["lname"];

echo $d;

$e=$\_POST["phone"];

echo $e;

$f=$\_POST["address"];

if($b==$g)

{

$sql1="INSERT INTO `login`(`username`, `password`, `status`, `role`) VALUES ('$a','$b',0,1)";

$result1=mysqli\_query($con,$sql1);

$sql2="select max(lid) as id from login";

$res=mysqli\_query($con,$sql2);

$ar=mysqli\_fetch\_array($res);

$mid=$ar['id'];

echo $mid;

//$sq="INSERT INTO `screen`(`theaterid`, `tmode`, `noscreen`, `seatno`) VALUES ('$mid','$m','$n','$o')";

//$ress=mysqli\_query($con,$sq);

$sql="INSERT INTO `userlogin`(`fname`, `lname`, `phone`, `address`, `status`, `lid`) VALUES ('$c','$d','$e','$f',0,'$mid')";

$result=mysqli\_query($con,$sql);

$to = $a;

$subject = "Universal Cinema";

$message ="<html>

<head>

<title>User registeration</title>

</head>

<body>

<p>your username and password is below</p>

<table>

<tr>

<th>UserName</th>

<th>password</th>

</tr>

<tr>

<td>'$a'</td>

<td>'$b'</td>

</tr>

</table>

</body>

</html>";

$headers = "MIME-Version: 1.0" . "\r\n";

$headers .= "Content-type:text/html;charset=UTF-8" . "\r\n";

$headers .= 'From: <anands@mca.ajce.in>' . "\r\n";

$headers .= 'Cc: anands@mca.ajce.in' . "\r\n";

mail($to,$subject,$message,$headers);

}

else

{

echo("enter correct password");

}

}

?>

<html>

<head>

<script src="validation.js"></script>

<script src="reg.js"></script>

<link rel="stylesheet" href="reg.css" />

</head>

<body>

<div class="menu">

<table style="color: #fff;

font-size: 2em;

font-weight: 200;

font-family: 'Georgia', cursive;

align="right";

>

<tr><td><a class="links" style="color: #DC6180;text-decoration: none;" href="index.html">

HOME&nbsp|&nbsp

</a></td>

<td><a class="links" style="color: #FFFFFF;text-decoration: none;" href="index.html">

LOGIN &nbsp&nbsp

</a></td></tr>

</table>

</div>

<form name="treg" id="msform" method="post" action="#">

<fieldset>

<table>

<tr><th style="align:center;"><h2 class="fs-title">Create your account</h2></th></tr>

<tr><td><input type="text" name="email" id="email" placeholder="Email" onchange="em()" required /></td></tr>

<tr> <td><input type="password" name="pass" id="pass" placeholder="Password" onchange="ps()" required /></td></tr>

<tr> <td><input type="password" name="cpass" placeholder="Confirm Password" required /></td></tr>

<tr style="align:center;"><td>Personal Details</td></tr>

<tr><td><input type="text" name="fname" id="fname" placeholder="First Name" onchange="n()"required /></td>

<td><input type="text" name="lname" id="lname" placeholder="Last Name" onchange="ln()" required /></td></tr>

<tr><td><input type="text" name="phone" id="phone" placeholder="Mobile no" onchange="p()" required /></td></tr>

<tr><td><textarea name="address" placeholder="Address" required ></textarea></td></tr>

<tr><td><input type="submit" name="submit" class="submit action-button" value="Register" /></td>

</tr>

</table>

</fieldset>

</form>

</body>

</html>

**Seat booking php code**

<?php

include 'connection.php';

if(!(isset($\_SESSION['username'])))

{

header('location:index.php');

}

if(isset($\_POST['num'])){

$a=$\_POST['num'];

$n=$\_SESSION['arc'];

$\_SESSION['cnt'][$n]=$a;

//$c[$n]=$a;

//$c[2]=$a;

for($i=0;$i<$n+1;$i++)

{

print\_r($\_SESSION['cnt'][$i]);

}

$n=$n+1;

$\_SESSION['arc']=$n;

header("location:demo.php");

}

if(isset($\_POST['add'])){

$tid=$\_SESSION['tid'];

$uid=$\_SESSION['lid'];

$n=$\_SESSION['arc'];

$dt=$\_SESSION['dt'];

$shid=$\_SESSION['shid'];

$price=0;

for($i=0;$i<$n;$i++)

{

$price=$price+120;

}

$price;

$sql9="SELECT \* FROM `staffreg` WHERE `theaterid`='$tid'";

$res=mysqli\_query($con,$sql9);

$row4=mysqli\_fetch\_array($res);

$q=$row4['lid'];

//echo $q;

$sql4="SELECT \* FROM `wallet` WHERE `logid`='$uid'";

$result4=mysqli\_query($con,$sql4);

$row4=mysqli\_fetch\_array($result4);

$r1=$row4['balance'];

$sql8="SELECT \* FROM `wallet` WHERE `logid`='$q'";

$result8=mysqli\_query($con,$sql8);

$row8=mysqli\_fetch\_array($result8);

$r2=$row8['balance'];

//echo $r2;

$ttl=$r1-$price;

$fprice=$r2+$price;

//echo $fprice;

if($price>$r1)

{

echo "<script>if(confirm('no balance!!!!')){document.location.href='userhome.php'}else{document.location.href='userhome.php'};</script>";

}

else

{

for($i=0;$i<$n;$i++)

{

$sid=$\_SESSION['cnt'][$i];

//echo($sid);

//echo($tid);

$sql6="UPDATE `wallet` SET `balance`='$ttl' WHERE `logid`='$uid'";

$result6=mysqli\_query($con,$sql6);

$sql7="UPDATE `wallet` SET `balance`='$fprice' WHERE `logid`='$q'";

$result7=mysqli\_query($con,$sql7);

//echo($uid);(`bookid`, `userid`, `theaterid`, `seatid`, `show\_num`, `date`, `paystatus`)

$sql="INSERT INTO `seat\_book`(`userid`, `theaterid`, `seatid`, `show\_num`, `date`, `paystatus`) VALUES ('$uid','$tid','$sid','$shid','$dt','0')";

$result=mysqli\_query($con,$sql);

//echo $sql;

}

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

unset($\_SESSION['cnt']);

unset($\_SESSION['arc']);

$\_SESSION['cnt']=array();

$\_SESSION['arc']=0;

header("location:demo.php");

}

}

?>

Design of seat booking

<html>

<head>

<style>

.topnav {

overflow: hidden;

background-color: #333;

}

.topnav a {

float: left;

display: block;

color: #f2f2f2;

text-align: center;

padding: 14px 16px;

text-decoration: none;

font-size: 17px;

}

.topnav a:hover {

background-color: #ddd;

color: black;

}

.topnav a.active {

background-color: #4CAF50;

color: white;

}

</style>

</head>

<body>

<div class="topnav">

<a class="active" href="userhome.php">Home</a>

<a href="profileview.php">Profile</a>

<a href="#">Book My Show </a>

<a href="userviewfilms.php">View films </a>

<a href="newrelease.php">New Release </a>

<a href="wal.php">Add bank</a>

<a href="mywallet.php">view my Wallet</a>

<a href="logout.php">Logout</a>

<!--<?php $mid=$\_SESSION['fname'];?> -->

<p style="color:white; margin-left:90%;">hai...!!!<?php echo $mid;?></p>

</div>

<form name="f1" action="#" method="post">

<select name="loc">

<option value=0>Show time</option>

<option value="1">Morning show</option>

<option value="2">Noon show</option>

<option value="3">Mattini show</option>

</select>

<span>select date:</span><input type="date" name="date2" >

<input type="submit" name="ok" value="ok">

</form>

<form action="dd.php" method="post" >

<?php

include 'connection.php';

if(isset($\_POST['submit']))

{

$theaterid=$\_POST['sid'];

$\_SESSION['tid']=$theaterid;

}

if(isset($\_POST['ok']))

{

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

unset($\_SESSION['cnt']);

unset($\_SESSION['arc']);

$\_SESSION['cnt']=array();

$\_SESSION['arc']=0;

date\_default\_timezone\_set('Asia/Kolkata');

$date3 = date('Y-m-d', time());

$time2=date('h:i:s a', time());

$v=date("H:i", strtotime($time2));

$show=$\_POST['loc'];

$\_SESSION['shid']=$show;

$d2=$\_POST['date2'];

$\_SESSION['dt']=$d2;

$date="$date3 $v";

//echo $date3;

}

if(isset($\_SESSION['tid']) and isset($\_SESSION['shid']))

{

date\_default\_timezone\_set('Asia/Kolkata');

$date3 = date('Y-m-d', time());

$time2=date('h:i:s a', time());

$v=date("H:i", strtotime($time2));

$date="$date3 $v";

$shid=$\_SESSION['shid'];

$tid=$\_SESSION['tid'];

$dt=$\_SESSION['dt'];

if($shid==1)

{

$date2="$dt 10:30";

if($date2<$date)

{

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

echo "<script>if(confirm('enterd date is not valid!!!')){document.location.href='demo.php'}else{document.location.href='demo.php'};</script>";

}

}

elseif($shid==2)

{

$date2="$dt 14:15";

if($date2<$date)

{

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

echo "<script>if(confirm('enterd date is not valid!!!')){document.location.href='demo.php'}else{document.location.href='demo.php'};</script>";

}

}

elseif($shid==3)

{

$date2="$dt 21:00";

if($date2<$date)

{

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

echo "<script>if(confirm('enterd date is not valid!!!')){document.location.href='demo.php'}else{document.location.href='demo.php'};</script>";

//echo "<script>alert('enterd date is note valid!!!');</script>";

}

}

else

{

unset($\_SESSION['dt']);

unset($\_SESSION['shid']);

echo "<script>if(confirm('select show!!!')){document.location.href='demo.php'}else{document.location.href='demo.php'};</script>";

//echo "<script>alert('select show!!!');</script>";

}

//echo $tid;

//SELECT \* FROM `seat\_book` WHERE `bookid``userid``theaterid``seatid``show\_num``date``paystatus`

$tid=$\_SESSION['tid'];

//echo($theaterid);

$sql="SELECT \* FROM `seat` WHERE `theaterid`='$tid'";

$result=mysqli\_query($con,$sql);

$row=mysqli\_fetch\_array($result);

$row1=$row['row1'];

$col1=$row['col1'];

$count=1;

$st=$row['startno'];

$end=$row['endno'];

$n=$\_SESSION['arc'];

//$shid=$\_SESSION['shid'];

$k=0;

for($i=0;$i<$row1;$i++)

{

for($j=0;$j<$col1;$j++)

{

if($count==$st&&$count<=$end)

{

$st++;

$t=false;

$h=false;

$sql="SELECT `seatid` FROM `seat\_book` WHERE `theaterid`='$tid' and `date`='$dt' and `show\_num`='$shid' and `seatid`='$count'";

$result=mysqli\_query($con,$sql);

$row4=mysqli\_fetch\_array($result);

$siid1=$row4['seatid'];

if($count==$siid1)

{

$h=true;

}

for($k=0;$k<$n;$k++)

{

if($count==$\_SESSION['cnt'][$k])

{

$t=true;

}

}

if($t==true || $h==true)

{

?>

<input type="button" value="<?php echo($count);?>" disabled style="width:4%; border-color:yellow;">

<?php

$k=$k+1;

}

else{

?>

<input type="submit" name="num" value="<?php echo($count);?>" style="width:4%; border-color:blue;">

<input type="hidden" name="co" value="<?php echo($count);?>">

<?php

}

}

else

{

?>

<input type="button" value="<?php echo($count);?>" disabled style="width:4%; border-color:red;">

<?php

}

$count++;

}

?>

</br>

<?php

}

}

?>

</form>

<form action="dd.php" method="post">

<input type="submit" name="add" value="BOOK NOW" style="width:8%; color:red;">

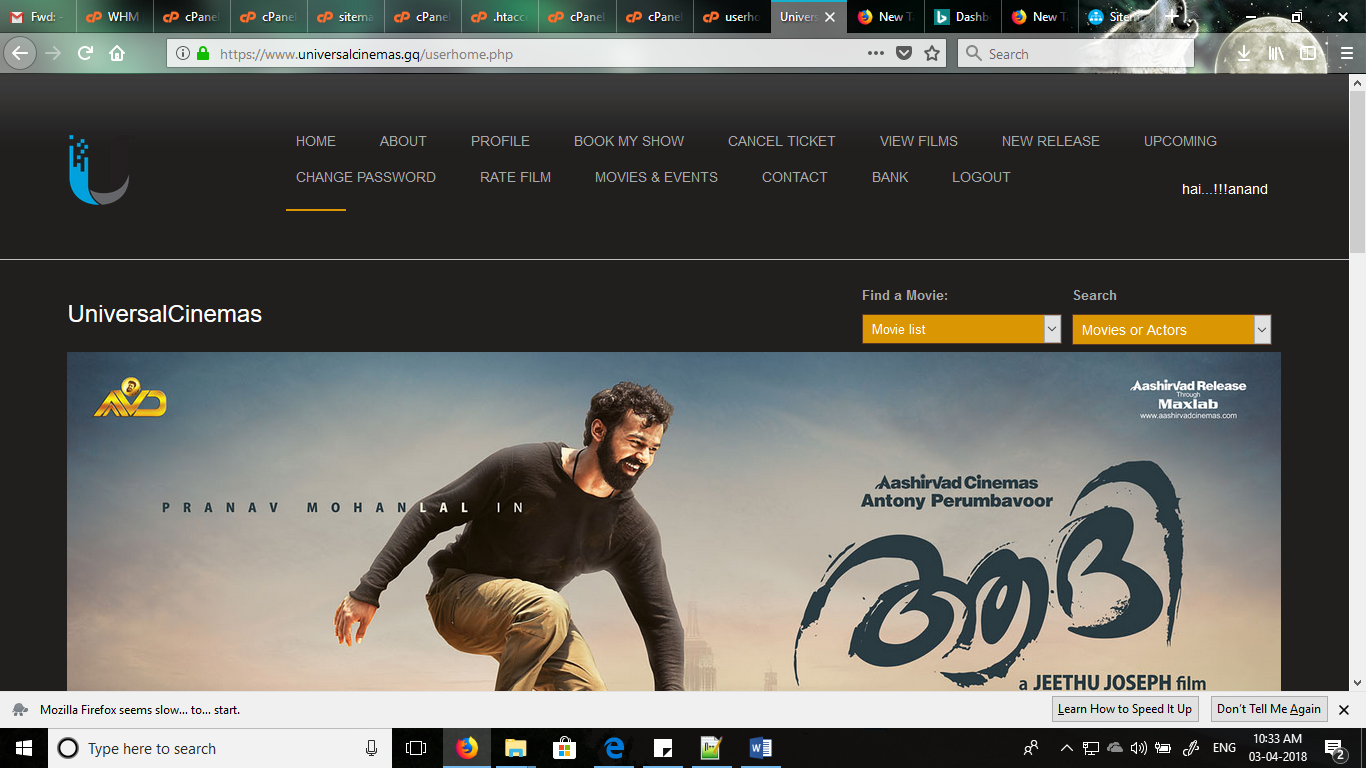
</form>

</body>

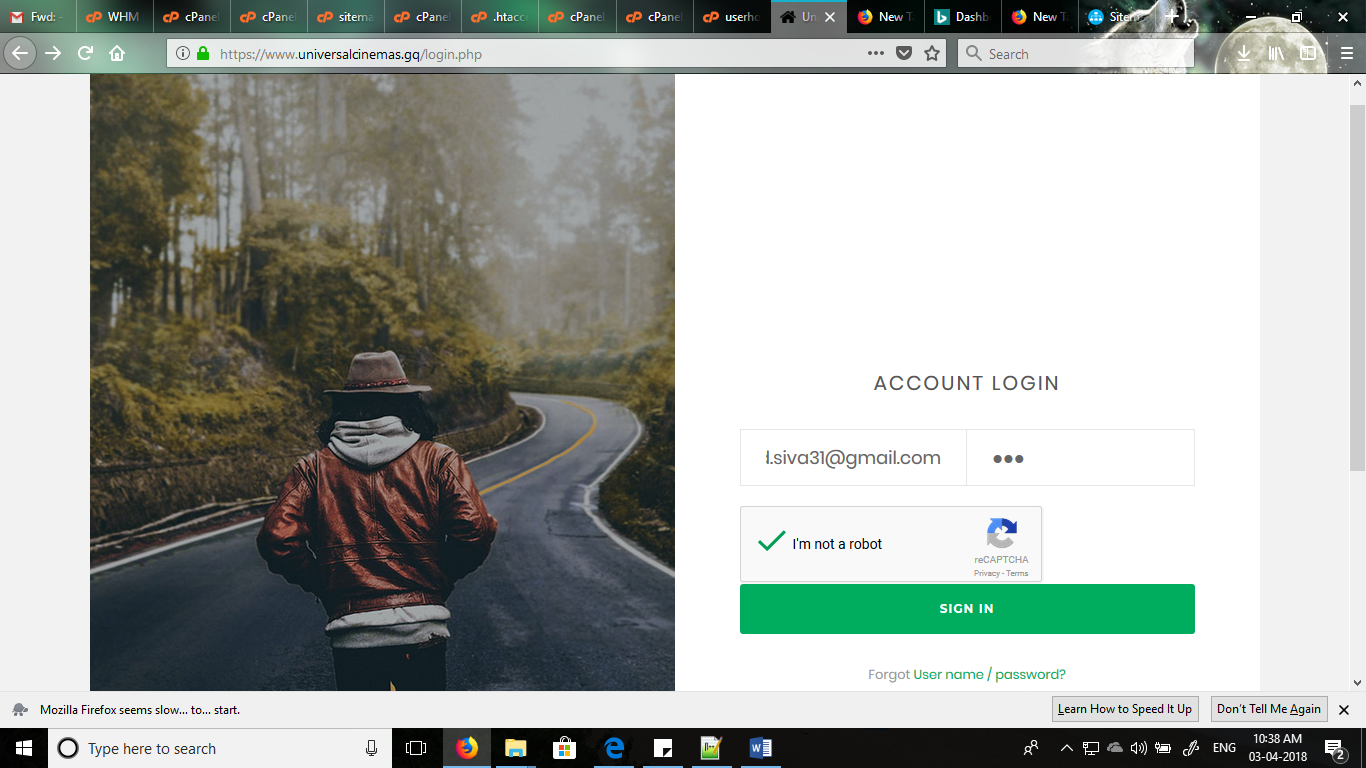
</html>

#### Screen Shots

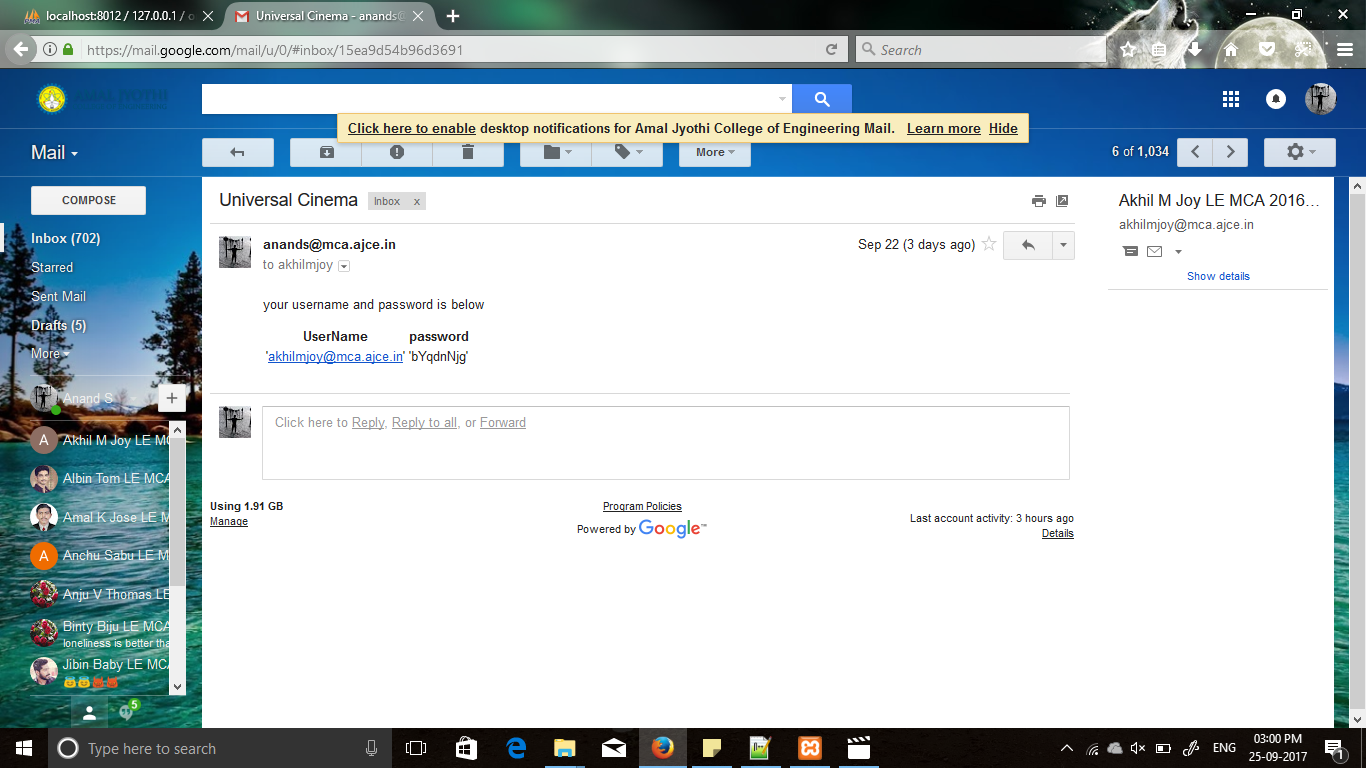
Home Page



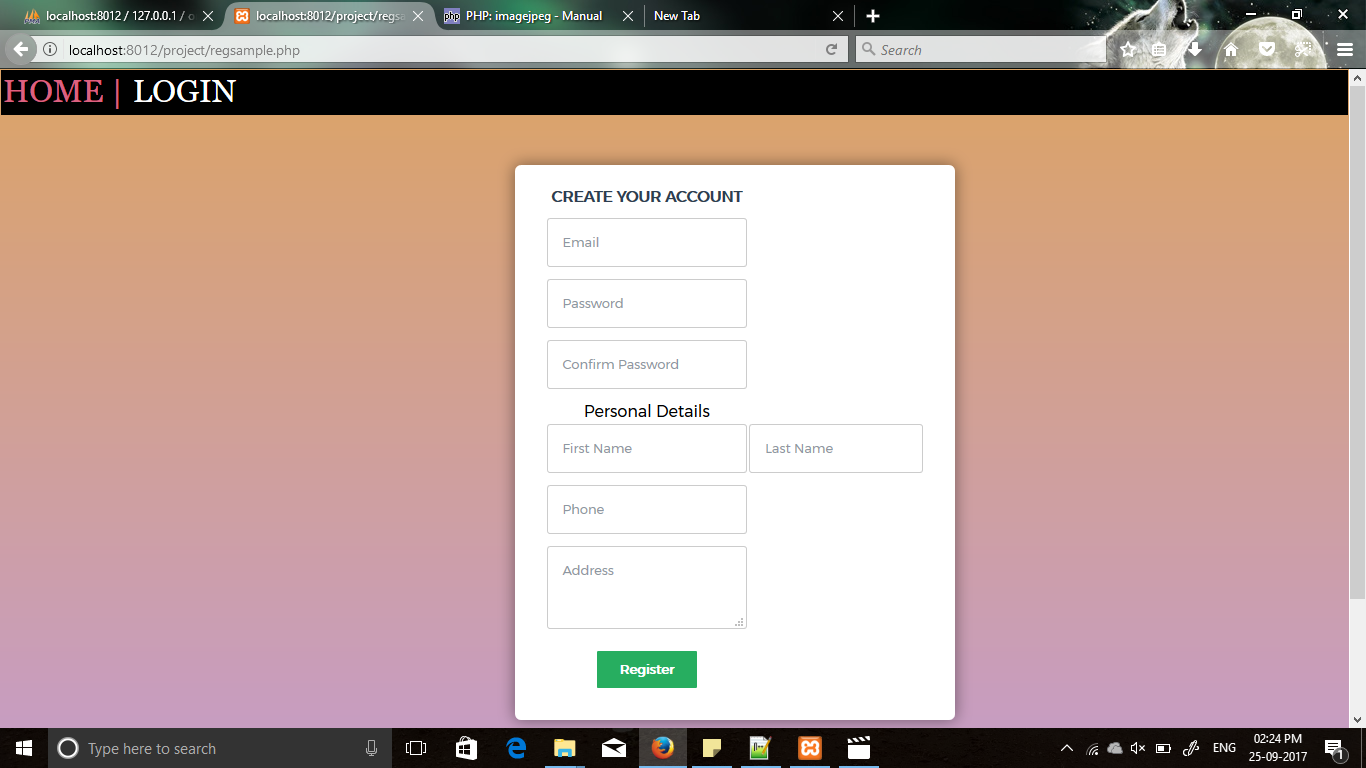
Login



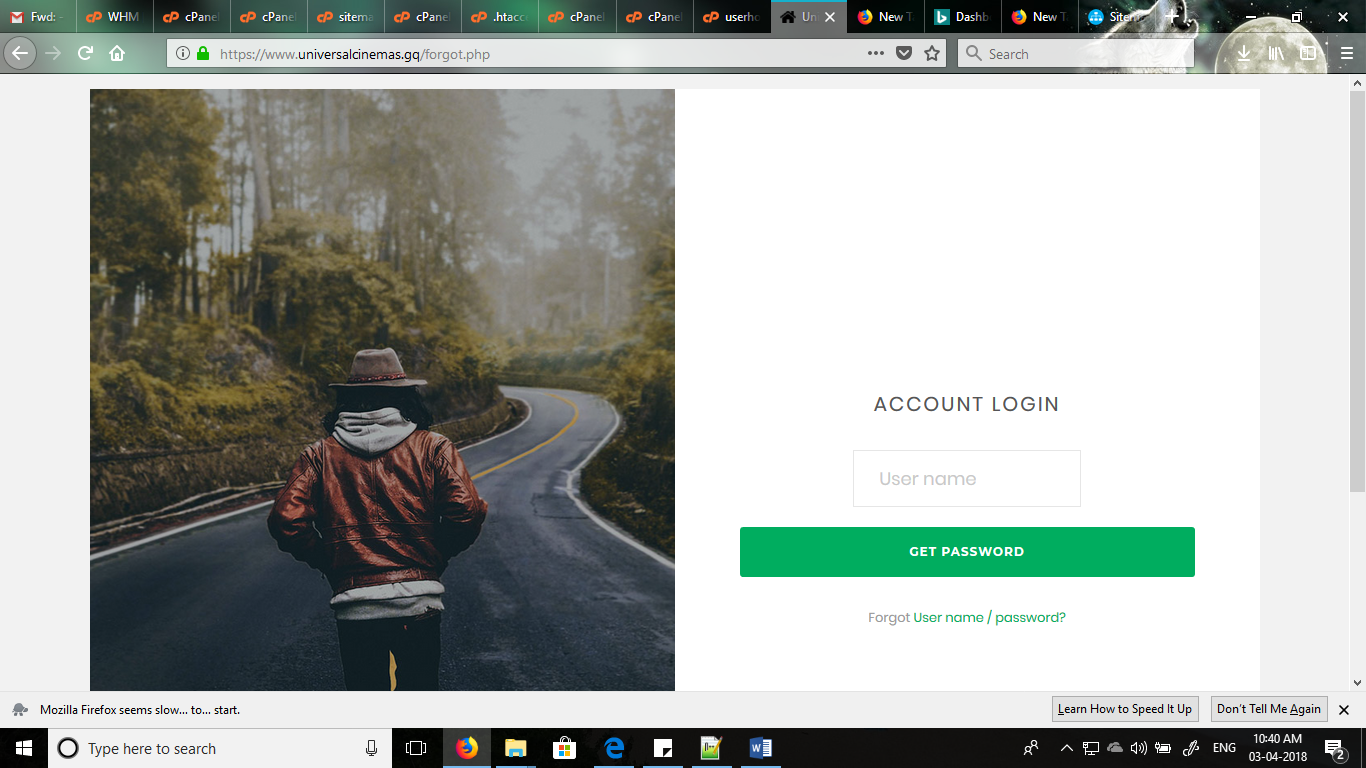
Confirmation Mail



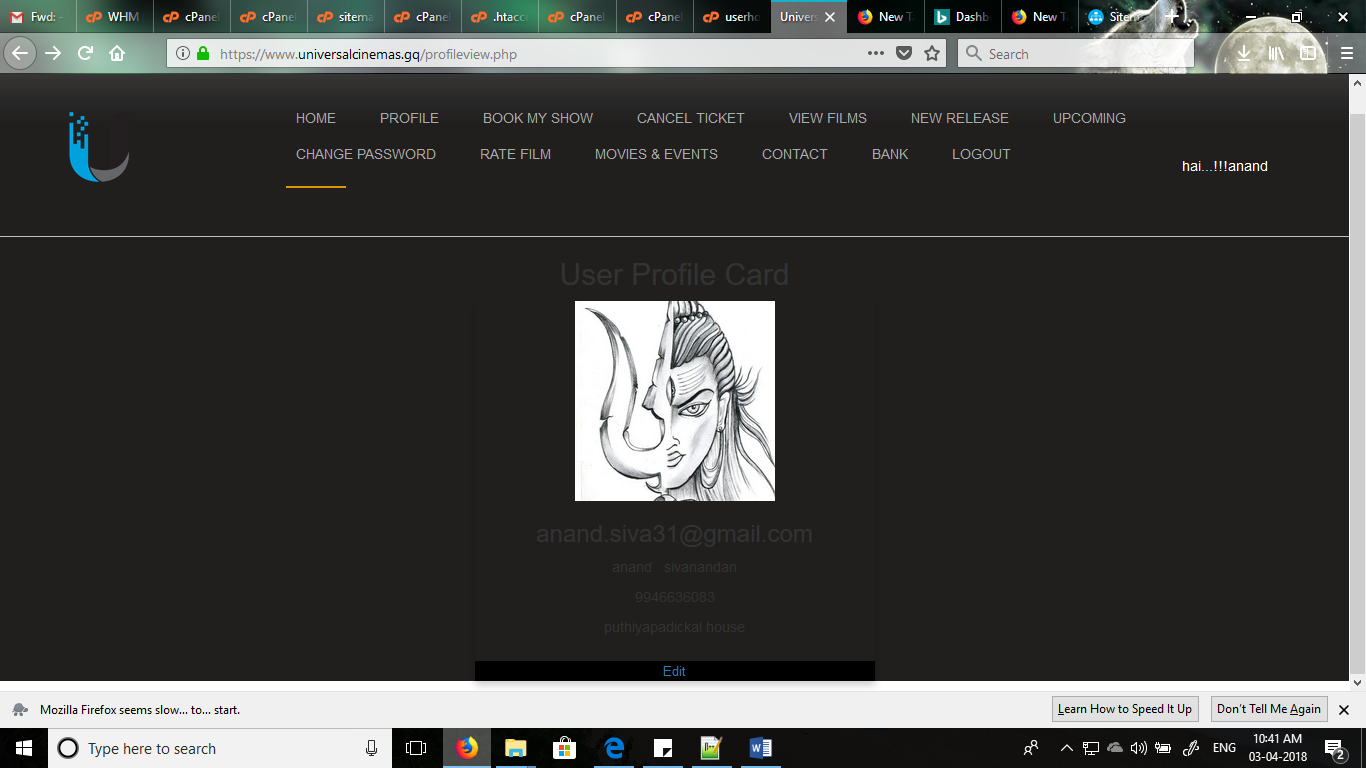
User Reg



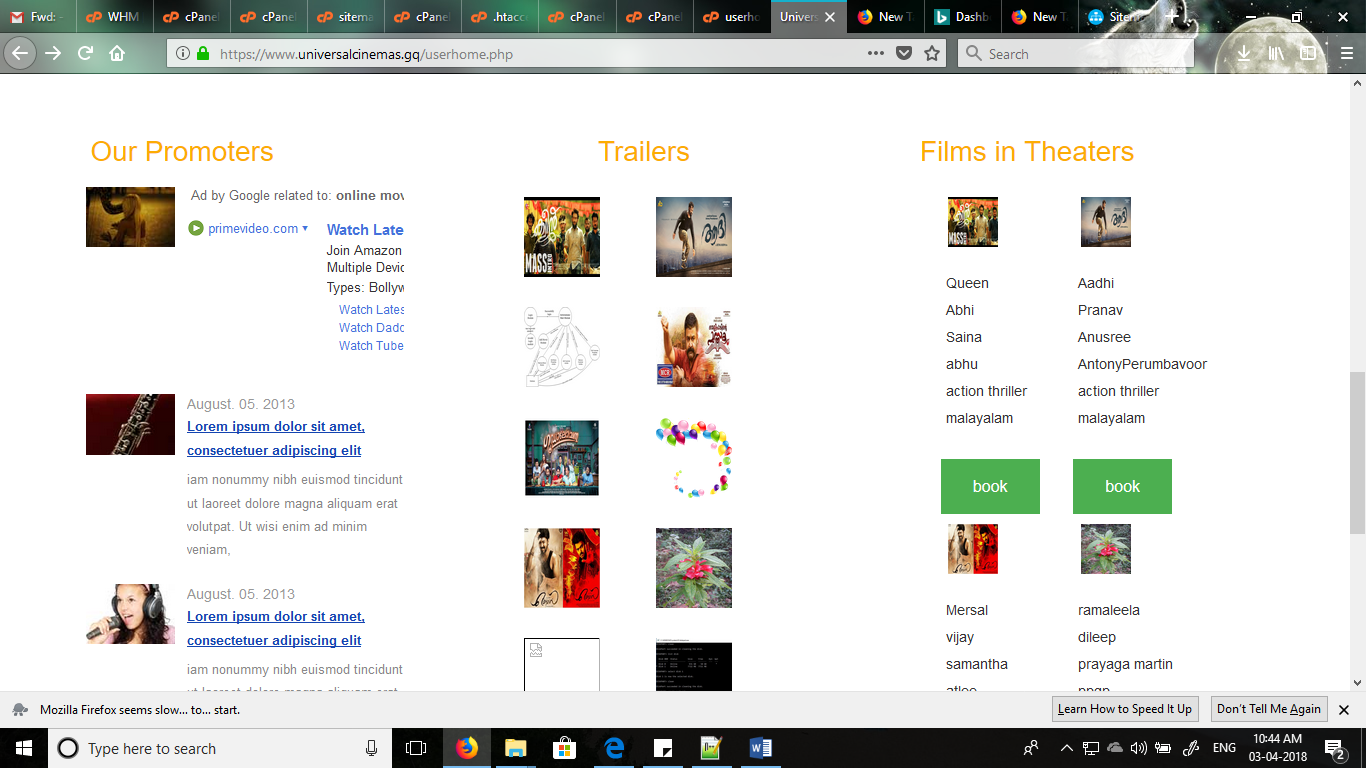
Forgot Password

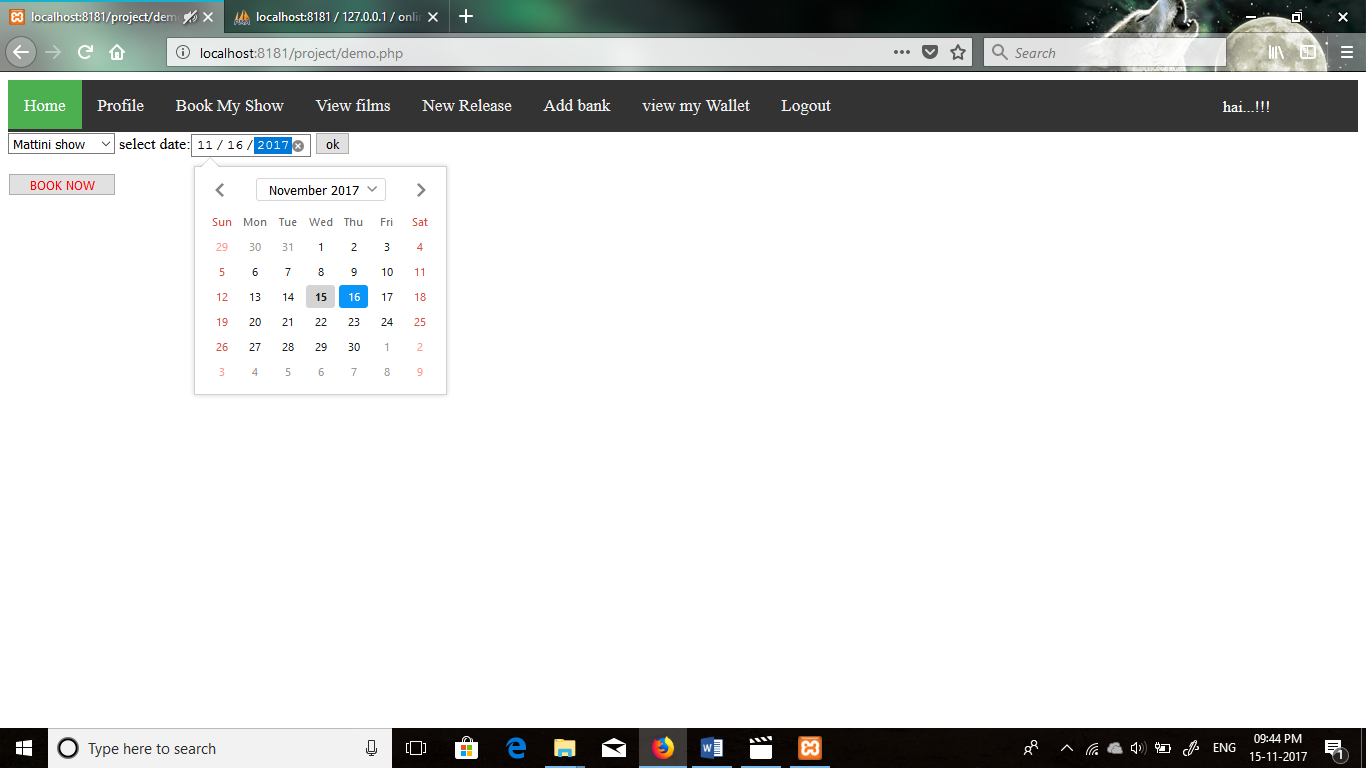


Profile View



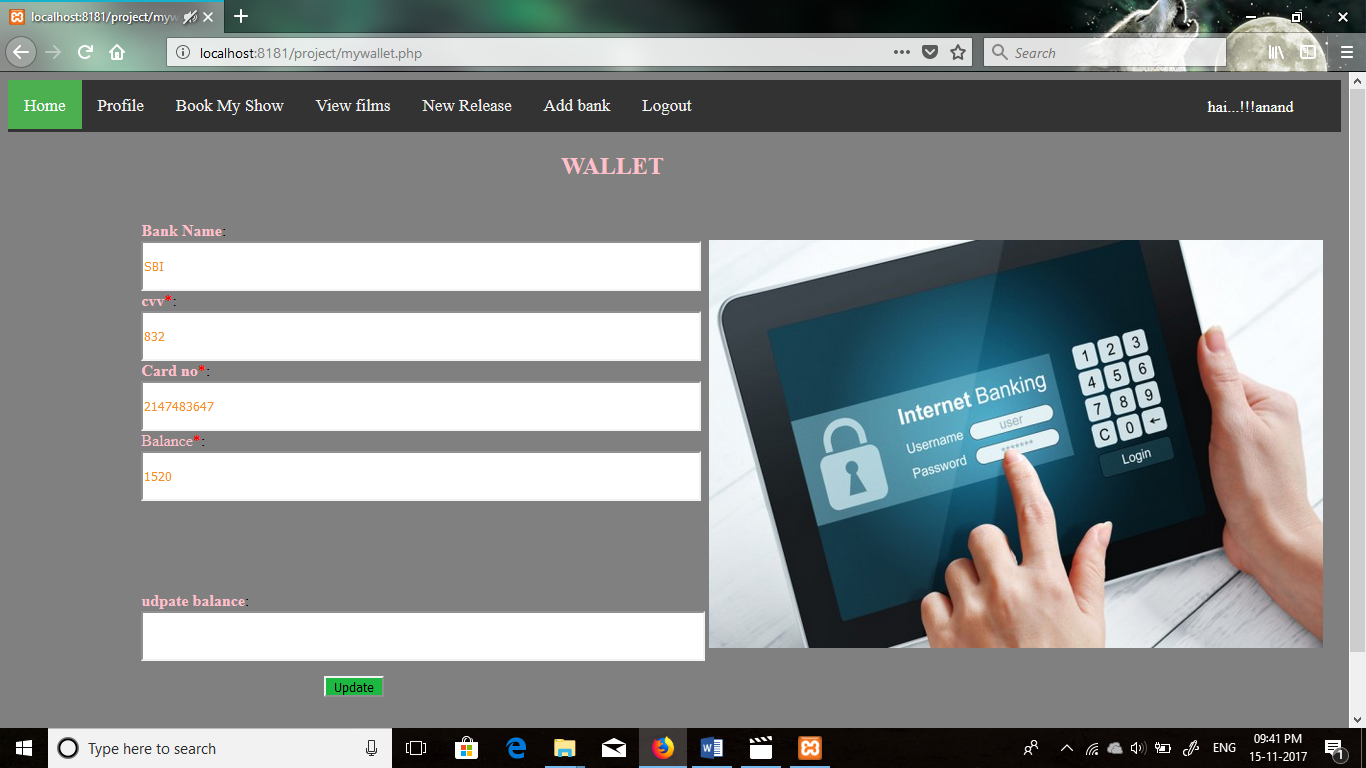
Show Booking



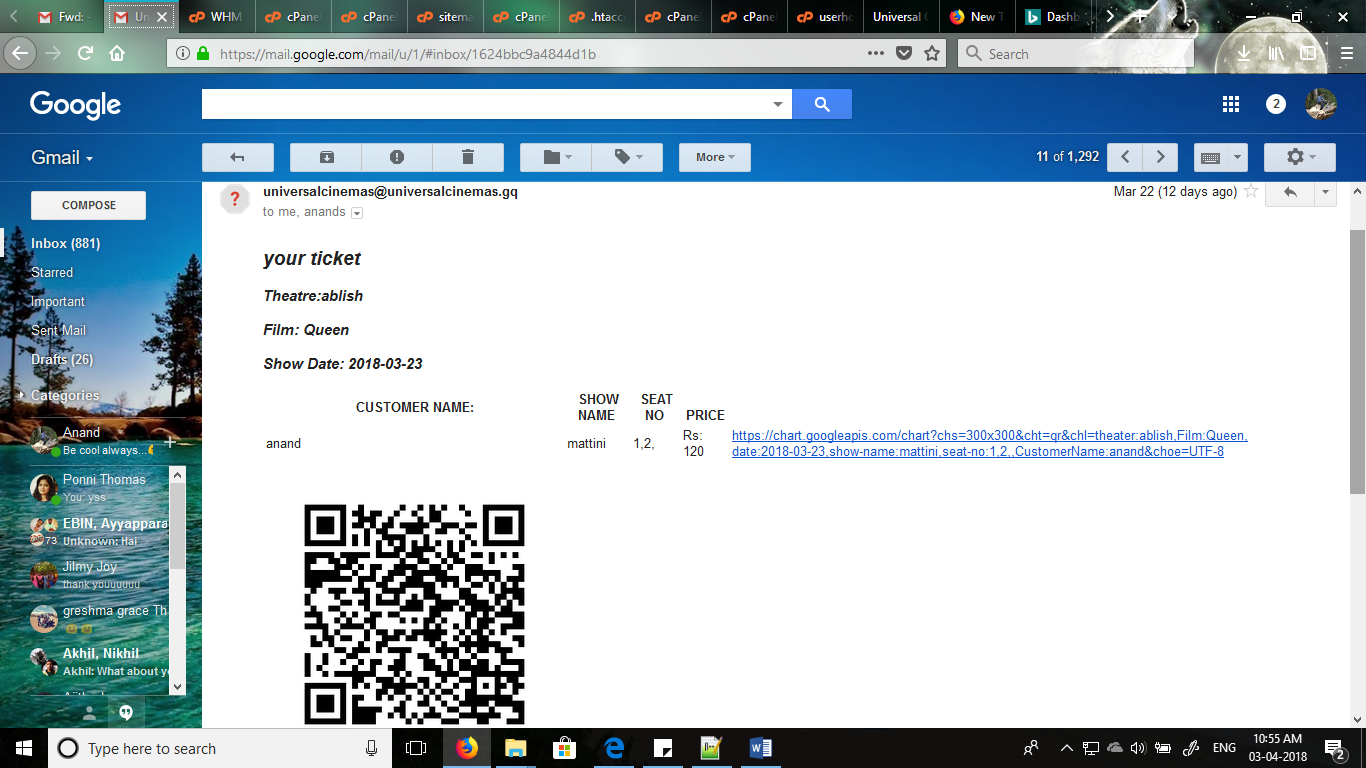




Wallet



Show Ticket



Ticket Cancellation

