****

**SSR COLLEGE OF ARTS, COMMERCE & SCIENCE**

**(Affiliated to Savitribai Phule Pune University)**

**Sayli,silvassa – 396230,D.& N.H.**

**Submitted to the partial fulfilment of**

**T.Y.B.Sc (Computer Science)**

**2017-2018**

**PROJECT WORK**

**“BUS RESERVATION SYSTEM”**

**Guided by: Submitted by:**

**Mrs. KAVITA SAPTASAGAR 1. ADARSH DIXIT**

**2. YASH PATEL**

****

**SSRCOLLEGE OF ARTS, COMMERCE & SCIENCE**

**(Affiliated to Savitribai Phule Pune University)**

**Sayli, Silvassa – 396230, D. & N.H.**

***Department of Computer Science***

**C E R T I F I C A T E**

This is to certify that Mr. ADARSH DIXIT and Mr. YASH PATEL

of T.Y.B.Sc. [Computer Science], Seat No.: 14079 & 14086 has completed project work in “Bus Reservation System” during the year 2017 – 2018.

In-charge H.O.D.

Internal External

Examiner Examiner Seal of the College

**DECLARATION**

We hereby declare that this project on “Bus Reservation System” is completed by our self under the guidance of **Mrs. KAVITA SAPTASAGAR**

This system has been designed by us on our own and this is not complete project and may require some modification in future as per user requirement. From practical implementation point of view flexibility is incorporated in package.

We are sure that we can do any kind of modification suggested by modifying file designed or necessary program code.

**ACKNOWLEDGEMENT**

I hear and forget, I see and remember, I do and I understand, so goes on the saying and how true it is!

Practical knowledge always has an upper hand over theoretical ones and this is what we understood during our project.

The satisfaction that accompanies with the successful completion of any task is always incomplete if we do not appreciate the people who have cooperated to make this project possible, whose constant guidance and encouragement crown all efforts with success.

Firstly, we would like to thank God for His grace and blessing.

We would like to thank the University for giving us the opportunity to apply our knowledge and skills in a practical environment as a part of curriculum for **B. Sc (Compute Science)**.

We are thankful to the staff of **Computer Science Department, SSR College of Arts, Commerce and Science, Sayli, Silvassa**, for all the direct and indirect help for providing Computer department with the necessary resources for the timely completion of our project. We are grateful to our project guide **Mrs. KAVITA SAPTASAGAR** for giving us inspiration and constructive suggestions that has helped us in preparing this project.

We will be ungrateful if we do not acknowledge our families and friends who have always been there for us. Last but not the least we also appreciate everyone who has helped us in some way or the other.

**ABSTRACT**

This project focuses on the “Bus Reservation System". The knowledge acquired proves to be helpful and effective, only if it is limited. The implement of the knowledge acquired is through project and case studies based on the real time scenarios.

The "Bus Reservation System" undertaken as a project is based on relevant technologies. The project is done individually, following the best coding standards and convection and adheres to project documentation standards.

The system provides different access rights through the administrator’s login. The proposed system helps administrator to configure the basic settings. These features help to display the details from the database. This project involve several modules to carry out each action.

We can also stores the details at run time. We can enter the bus details in database at run time. We can either store in a database or see details at the runtime by selecting the option view.

Hence this purposes system will maintain all the information in a standard database and will be able to generate reports as per the user requirements.

This project is capable of providing the user with the facility of booking tickets as well as knowing the status of the tickets and their availability.

**What is Travel management system?**

The term "Bus Reservation System" suggest that it is a system which assists in managing the company's travel related details, while commanding and controlling the implementation of procedures and changes, maintaining customer’s information and generating bills.

**The main functionalities of this system are as follows:**

* To maintain information about current availability of buses based on which it decides whether the requested ticket is to be booked or not.
* To display the information about buses and also add information about new buses.
* To maintain the basic information about company's personal details such as contact numbers, names, addresses.
* To maintain the information about customers who books ticket, also includes adding new customers their details.
* Enables to search or check the availability of bus from their desired from and to places.
* The system can be accessed by administrator.
* It also generates the report(bill) based on the maintained database, thereby helping the company to observe the progress.
* It also generates the ticket receipt.

**PROJECT SCOPE**

This document covers the requirement for the “Bus Reservation System". This software will provide a graphical representation in which the users of the system will be able to perform operations that are associated with storing, maintaining booking information.

The purpose of this guide has helped the developer in selecting a design that will be able to accommodate the full-scale application.

**PRODUCT PERSPECTIVE**

The project “Bus Reservation System" is an independent project and does not depend on any other project or system. The project will automate various tasks associated with handling bus bookings details and better organization of the information and optimum performance, thus helping the company to ensure smooth running of these processes.

**PRODUCT FEATURES**

The project “Bus Reservation System" will automate various tasks associated with handling bookings details of new and existing bus. Periodically it will be used by the travels company to store information in database such as storing customers and buses details.

And all this information is accessible to administrators to keep track of their booking progress of the product. Also the software has a provision that will allow administrators to print the required report of various products.

**System Requirement Specifications**

**Hardware & Software Features**

Front-end: Adobe Dreamweaver

Back-end: Xampp server

DBMS: MySQL

Operating System: Windows XP or above

Hardware: Pentium4 or above

512MB RAM or above

Colour monitor client system

40 GB Hard Disk or above

**Analysis**

**System Planning and Initial Investigation:-**

**Data Gathering:**

Study procedure or operation manuals. Evolution of current forms distribution to specify what details are required from the system and to whom the forms are distributed. Interview owner and operating staff if the analyst is sensitive to the needs and fear of the user, this technique can prove to be most important one in the data collection phase. Observe user as they perform their jobs to provide some insight into basic problem areas. Prepare, distribute and evaluate questionnaires to supplements information if interview and observation are inadequate or if more data is required.

**Feasibility Study:**

If the feasibility study is to serve as a decision document, it must answer three key questions.

1. Is the technology available to solve the problem in accommodating the new opportunity?

2. Would the system as proposed be accepted and used by intended user?

3. Are the benefit of developing the appropriate information system greater than cost?

Technical Feasibility(Available Technology)

Behavioural Feasibility (User Acceptance)

Economic Feasibility (Cost/Benefit Analysis)

**Data Analysis and Logical Design:**

The system analyst has gathered all pertinent data, they are analysed and the logical design of the system in developed. A logical design shows the flow of data through a system.

**Requirements Analysis Process - Requirements Definition & Gathering:**

After the project has been acquired and the contract has been signed, one of the first functions of the Analysis team is the process of Requirement Definition.

This is the stage where client requirements are gathered. This is done on the basis of information provided by the client in the form of documents, existing systems and process specs, on-site analysis interviews with end-users, market research and competitor analysis. This stage has the following steps:

High Level Requirements Specification

Requirements Definition

Requirements Analysis & Negotiation

Requirements Management

Requirements Specification

Requirements Elicitation

**Requirements Elicitation** - is the process of gathering customer needs. This involves asking the customers, users and others about the objectives of the system, what is to be accomplished, how the system fits into the needs of the business and finally how the system will be used.

**Requirements Analysis**- is the process of understanding the problem and the requirements for a workable solution. Once the requirements have been gathered they become the basis for "Requirements Analysis". Analysis categorizes requirements and organizes them into related subsets, explores each requirement in relationship to others, examines requirement for consistency, omissions and ambiguity, and prioritizes. Requirements based on the needs of the customer. Rough estimate of development are made and used to access the impact of each requirement on project cost and delivery time. Using an iterative approach, requirements are eliminated, combined, and/or modified so that each party achieves some measure of satisfaction. The requirements are used to generate Business Process Flows, Use Cases Modeling and Data Flow diagrams which facilitates a clearer understanding of the requirements and its solution, for both the customer and the developer.

**Requirements Specification** -is the process of describing what a system will do. It involves scoping the requirements so that it meets the customer's vision. Requirements Specification serves as a foundation software, hardware and database design. It describes the function (Functional and Non-Functional) and performance of the system and the constraints that will govern its development. It specifies the inputs and also describes the outputs. These specifications need to be:

1. Complete

2. Comprehensive

3. Testable

4. Consistent

5. Unambiguous

6. Writable

7. Modifiable

8. Implement able

**Requirements Management**- is the process that helps the project team identify, control, and track changes to the requirements at any time as the project proceeds. Requirements Verification, Validation and Traceability examine the specification to ensure that all system requirements have been stated unambiguously and that inconsistencies, omissions and errors have been detected and corrected.

Thus, ensuring that the work products confirms to the standards established for the **process**, the **project** and the **system**.

**The General Model:**

Software life cycle models describe phases of the software cycle and the order in which those phases are executed. There are tons of models, and many companies adopt their own, but all have very similar patterns. The general, basic model is shown below:

Implementation

Testing

Design

Requirements

Each phase produces deliverables required by then next phase in the life cycle. Requirements are translated into design. Code is produced during implementation that is driven by the design.

**Requirements:**

Business requirements are gathered in this phase. This phase is the main focus of the project managers (owner).Meetings with owner and users are held in order to determine the requirements. Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system? These are general questions that get answered during a requirements gathering phase. This produces a nice big list of functionality that the system should provide, which describes functions the system should perform, business logic that processes data, what data is stored and used by the system, and how the user interface should work.

The overall result is the system as whole and how it performs, not how it is actually going to do it.

**Design**

The software system design is produced from the results of the requirements phase. Architects have the ball in their court during this phase and this is the phase in which their focus lies. This is where the details on how the system will work is produced. Architecture, including hardware and software, communication. Software design (UML is produced here) are all part of the deliverables of a design phase.

**Implementation**

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation my overlap with both the design and testing phases. Many tools exists(CASE tools) to actually automate the production of code using information gathered and produced during the design phase.

**Testing**

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole.

So in a nutshell, that is a very basic overview of the general software development life Cycle model. Now let's delve into some of the traditional and widely used variations.

**Waterfall Model**

This is the most common and classic of life cycle models, also referred to as a linear-sequential life cycle model. It is very simple to understand and use.

In a waterfall model, each phase must be completed in its entirety before the next phase can begin. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. Here the phases do not overlap.

**Waterfall Life Cycle:**

Requirements

Design

Implementation and unit testing

Integration and System Testing

Operation

**Advantages**

* Simple and easy to use.
* Easy to manage due to the rigidity of the model - each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* **Disadvantages**
* Adjusting scope during the life cycle can kill a project
* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty
* Poor model for complex and object-oriented projects.

**System Design**

System Design is the most important part of software development. It requires a careful planning and logical thinking on the part of the software are going to meet the software means to plan how the various part of the software are going to meet the desired goals.

It should be done with the almost care because if the phase contain any error then it is going to effect the performance of the system as result it may take more processing time ,more response time, more coding and extra workload.

**Design Requirement and Objectives**

*The following goals were kept in mind while designing the new system:*

* Replace the manual system with a computerized system, which will be more users friendly and easy to use.
* A common database and automation for resource planning and reporting.
* To make the system completely menu-driven and hence user friendly this was necessary so that even non-programmer could use the system effectively system could act as catalyst in achieving objectives.
* To design the system in such a way that it reduces redundant information.
* To make the system reliable, understandable and cost effective.

**Design overview**

The design system take the final specification of the system from analysis and find the best way of fulfilling them, given the technical environment and previous decision on required level of automation on the basic of previous investigation the main features to be incorporated in the tool are finalized.

**Database Dictionary**

**Table name: admin**

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Type** | **Description** |
| ***Adminid*** | **bigint(20)** | **PRIMARY KEY** |
| **Adminname** | **varchar(50)** | **NOT NULL** |
| **Adminaddr** | **varchar(50)** | **NOT NULL** |
| **Adminmob** | **bigint(20)** | **NOT NULL** |
| **Adminmail** | **varchar(50)** | **NOT NULL** |
| **Adminpsswrd** | **varchar(300)** | **NOT NULL** |

**Table name: bill**

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Type** | **Description** |
| ***Billid*** | **bigint(20)** | **PRIMARY KEY** |
| **Adminid** | **bigint(20)** | **NOT NULL** |
| **Custid** | **bigint(20)** | **NOT NULL** |
| **Busid** | **varchar(20)** | **NOT NULL** |
| **Busfrom** | **varchar(50)** | **NOT NULL** |
| **Busto** | **varchar(50)** | **NOT NULL** |
| **Ondate** | **Date** | **NOT NULL** |
| **Timestart** | **Time** | **NOT NULL** |
| **Bustype** | **varchar(50)** | **NOT NULL** |
| **Seatno** | **varchar(20)** | **NOT NULL** |
| **Price** | **bigint(20)** | **NOT NULL** |
| **Billdate** | **Date** | **NOT NULL** |

**Table name:bus**

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Type** | **Description** |
| **Driverid** | **bigint(50)** | **NOT NULL** |
| **Drivername** | **varchar(50)** | **NOT NULL** |
| **Drivermob** | **bigint(20)** | **NOT NULL** |
| **Drivermail** | **varchar(50)** | **NOT NULL** |
| **Driversalary** | **int(50)** | **NOT NULL** |
| ***Busid*** | **varchar(50)** | **PRIMARY KEY** |
| **Busfrom** | **varchar(50)** | **NOT NULL** |
| **Busto** | **varchar(50)** | **NOT NULL** |
| **Timestart** | **Time** | **NOT NULL** |
| **Bustype** | **varchar(50)** | **NOT NULL** |
| **Priceprst** | **bigint(20)** | **NOT NULL** |
| **Seat** | **int(20)** | **NOT NULL** |
| **Date** | **int(2)** | **NOT NULL** |

**Table name:customer**

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Type** | **Description** |
| **Custid** | **bigint(20)** | **NOT NULL** |
| **Custname** | **varchar(50)** | **NOT NULL** |
| **Custaddr** | **varchar(50)** | **NOT NULL** |
| **Custmob** | **bigint(20)** | **NOT NULL** |
| **Custmail** | **varchar(50)** | **NOT NULL** |

**Table name:route**

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Type** | **Description** |
| **Busid** | **varchar(20)** | **NOT NULL** |
| **Busfrom** | **varchar(50)** | **NOT NULL** |
| **Busto** | **varchar(50)** | **NOT NULL** |
| **Bustype** | **varchar(50)** | **NOT NULL** |
| **Priceprst** | **int(20)** | **NOT NULL** |
| **Timestart** | **Time** | **NOT NULL** |
| **Date** | **Date** | **NOT NULL** |
| **Seat** | **varchar(100)** | **NOT NULL** |

**Use Case Diagram**

**Admin Customer**

**Class Diagram**

Admin

Adminid Adminname Adaminaddr Adminmob Adminmail Adminpsswrd

Addadmin() Viewadmin()

Login

id

passwowrd

check id()

check password()

1 1

1

1

\*

1

Bus driverid drivername drivermob drivermail driversalary **busid**busfrom busto bustype Priceprst timestart seat date

Addbusdriver() viewbusdriver()

\* \*

Route

Busid busfrom busto bustype priceprst timestart date seat

Insertroute() View route() Updateroute()

Customer

Custid Custname Custaddr Custmob Custmail

Addcust() viewcust() pay()

1

1

PAY

Pay()

Bill

**billid**

adminid custid busid busfrom busto ondate timestart bustype seatno price billdate

gettotal() viewbill()

1

1

**TESTING**

**Test methods:-**

**Unit Testing-**Unit test case design after the high level design is approved by a Peer Technical Review. The Unit test cases shall be designed to test the validity of the programs correctness. White box testing will be used to test the modules and procedures that support the modules. The white box testing technique ignores the function of the program under test and will focus only on its code and the structure of that code. To accomplish this, a Statement and Condition technique shall be used. Test case designers shall generate cases that only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once:

Each decision statement in the program shall take on a true value and a false value at least once during testing.

Each condition shall take on each possible outcomes at least once during testing.

Each program shall be tested for complexity based.

An Equivalence Partitioning/Boundary Value technique shall be implemented. The four primary types of inputs will be:

**A Simple Test-** A test value that establishes the basic correctness of the process.

This shall be the starting point of all system and maintenance tests.

**Valid Input Values-** Test values within the boundaries of the specification equivalence classes. This shall be input data the program expects and is programmed to transform into usable values.

**Invalid Input Values-**Test equivalence classes outside the boundaries of the specification. This shall be input data the program may be presented, but that will not produce any meaningful output.

**Special Cases-** Input cases that are not identified by the specification. Such as nonsensical data(i.e. negative numbers or characters), depressing several input keys at once, or any other test case the test designer feels has a good chance of exposing an error.

The Equivalence Partitioning Technique is a test case selection technique in which the test designer examines the entire input space defined for the unit under test and seeks to find sets of input that are, or should be processed identically. Included in the input space are legal situations that have both syntactic and semantic meaning to the program, and illegal situations that are either syntactically or semantically meaningless to it. These sets of input are equivalence classes similarly, in the Boundary Value techniques, the test designer first identifies the range of expected inputs(e.g. equivalence classes), then tests the input values inside and outside that range.

**Testing Scope-** Outlined below are the main test types that will be performed for this release. All system test plans and conditions will be developed from the functional specification and the requirements catalogue.

**Functional Testing**

The objective of this test is to ensure that each element of the application meets the functional requirements of the business as outlined in the:

Requirements Catalogue

Business Design Specification

Other functional documents produced during the course of the project i.e. resolution to issues/change requests/feedback.

This stage will also include:

**Validation Testing**-which is intensive testing of the new Front end fields and screens. Windows GUI Standards, valid, invalid and limit data input, screen & field look and appearance, and overall consistency with the rest of the application.

The third stage includes **Specific Functional testing**- these are low-level tests which aim to test the individual processes and data flows.

**Integration Testing-**This test proves that all areas of the system interface with each other correctly and that there are no gaps in the data flow. Final Integration Test proves that system works as integrated unit when all the fixes are complete.

**Business (User) Acceptance Test**

This test, which is planned and executed by the Business Representative(s) , ensures that the system operates in the manner expected, and any supporting material such as procedures, forms etc are accurate and suitable for the purpose intended. It is high level testing, ensuring that there are no gaps in functionality.

**Performance Testing**

These tests ensure that the system provides acceptable response times (which should not exceed 4 seconds).

**Regression Testing**

A Regression test will be performed after the release of each Phase to ensure that-There is no impact on previously released software, and to ensure that there is an increase in the functionality and stability of the software.

The regression testing will be automated using the automated testing tool.

**Bash & Multi-User Testing**

Multi-user testing will attempt to prove that it is possible for an acceptable number of users to work with the system at the same time. The object of Bash testing is an ad-hoc attempt to break the system.

**Technical Testing**

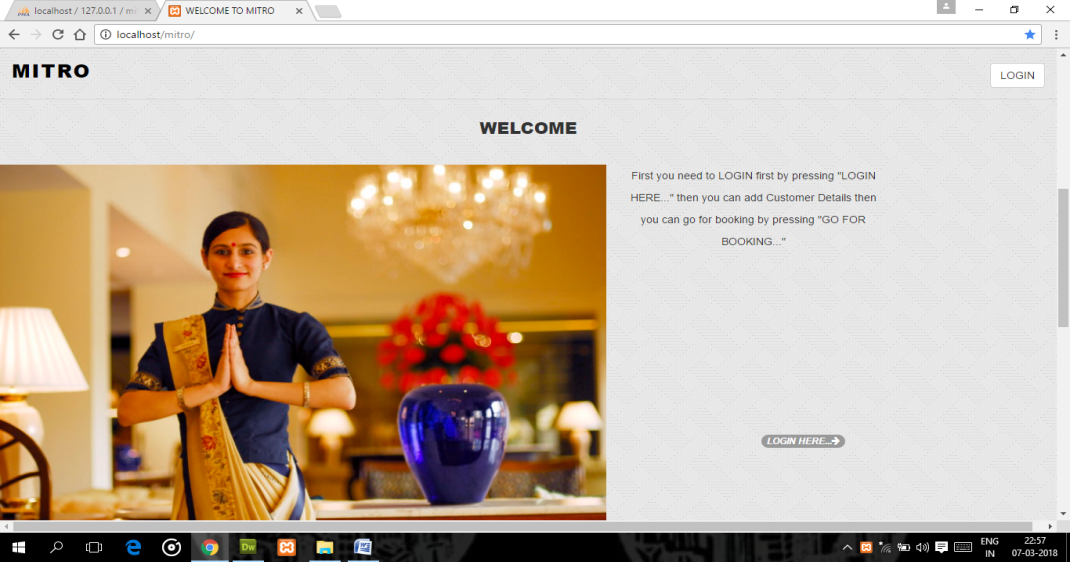
Technical Testing will be the responsibility of the Development Team.

6.2.8 Operations Acceptance Testing (OAT) the SIS team will define their own testing criteria, and carry out the tests.

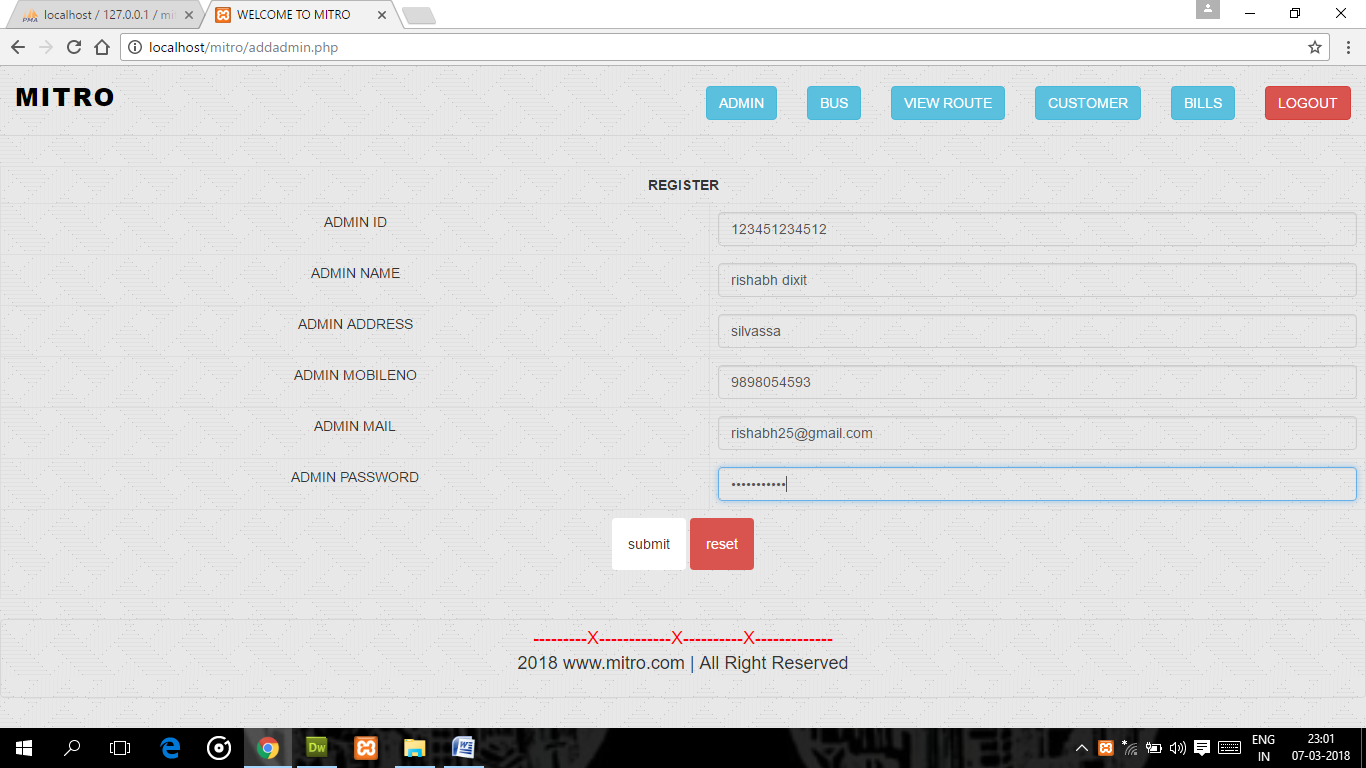
**SCREENSHOTS**

**Home Page:-**

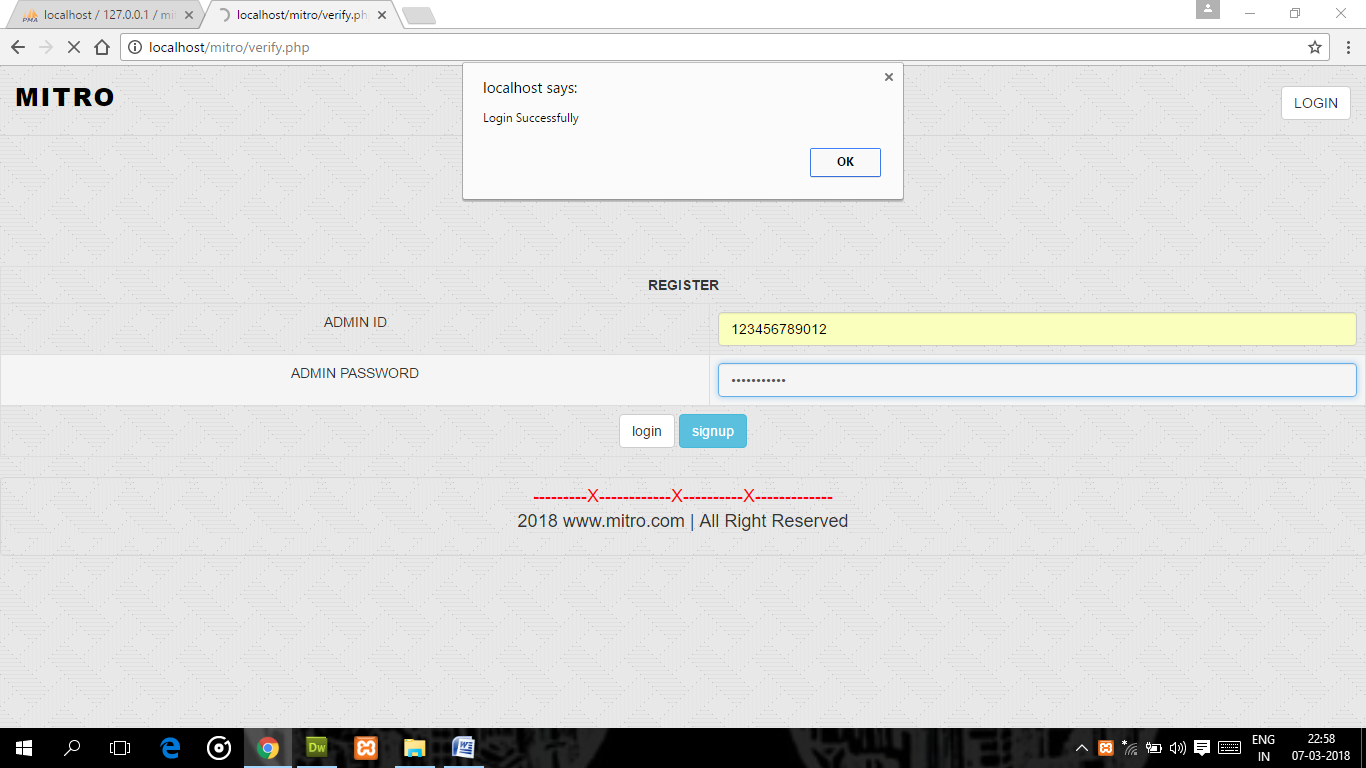




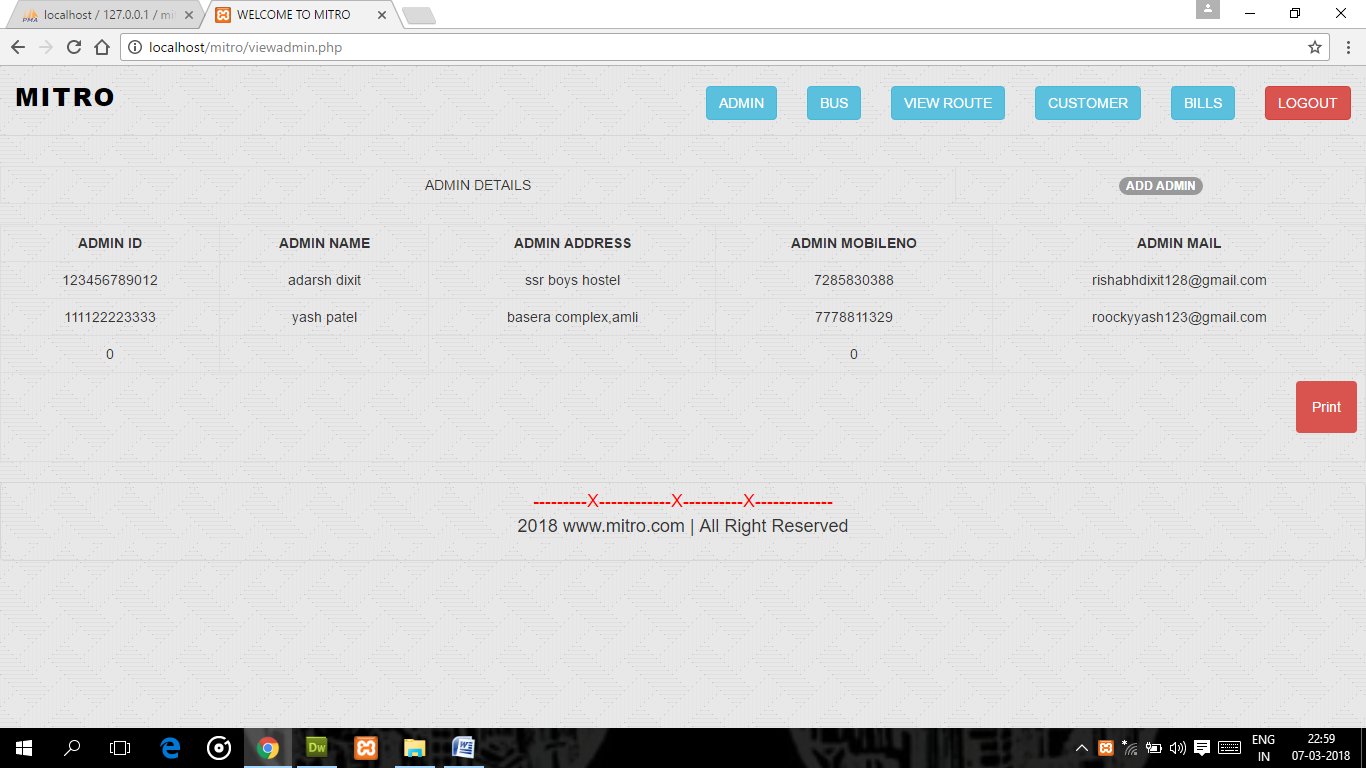
**Sign Up page:-**

**ge:**

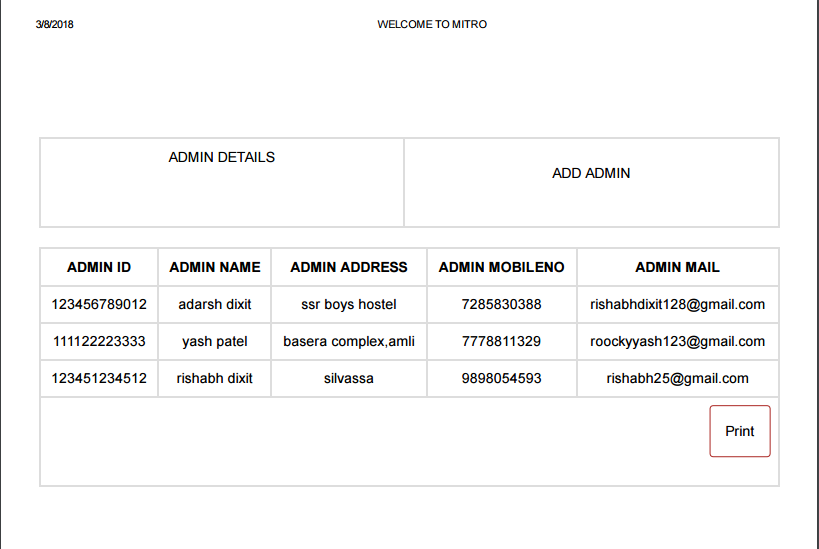
**Login page:-**



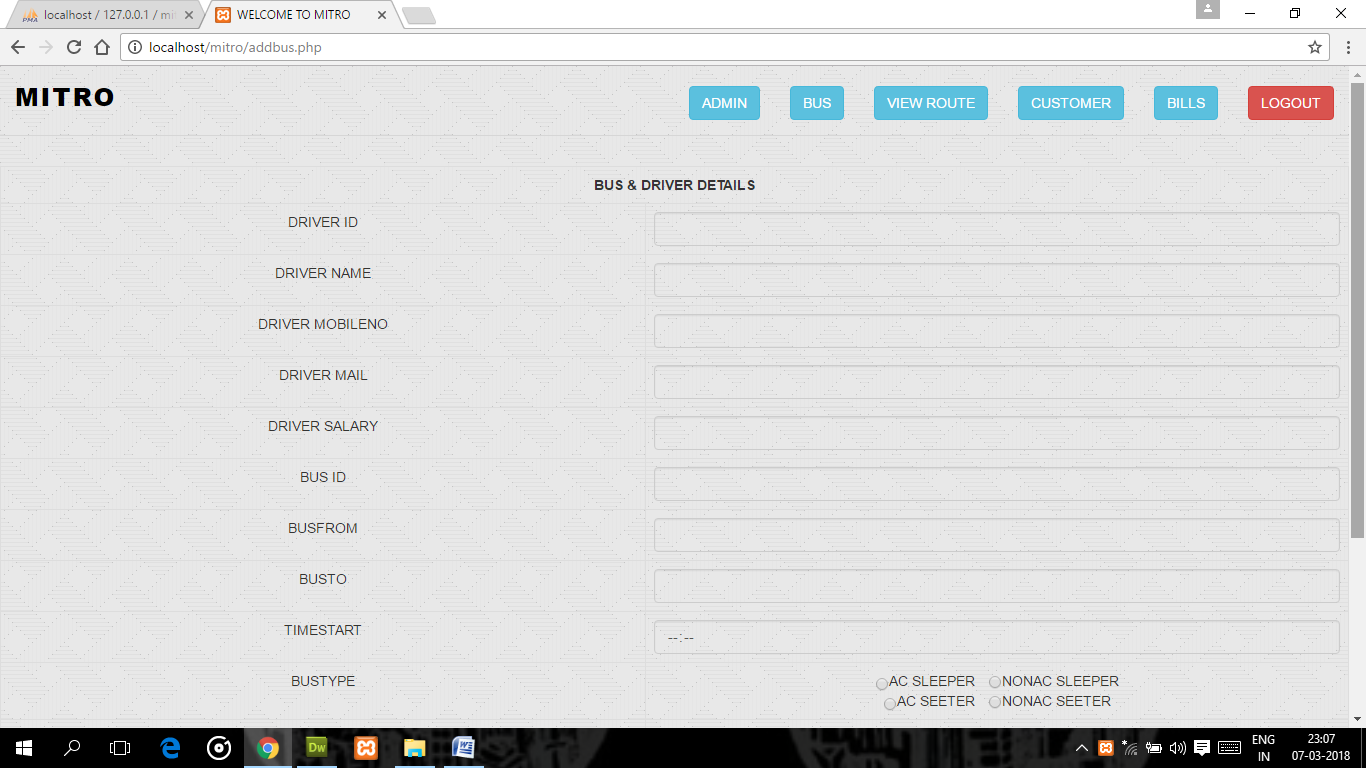
**View Admin page:-**

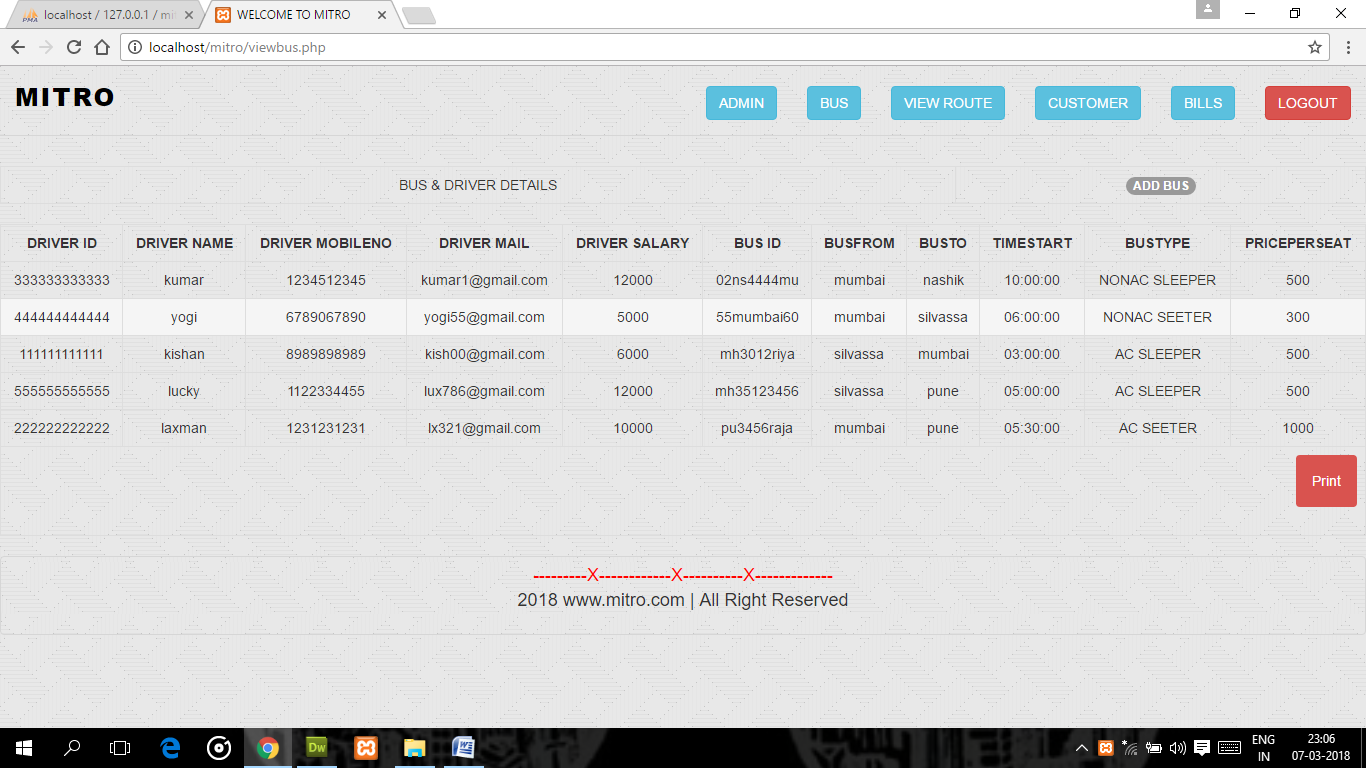


**Print The Admin Details:-**

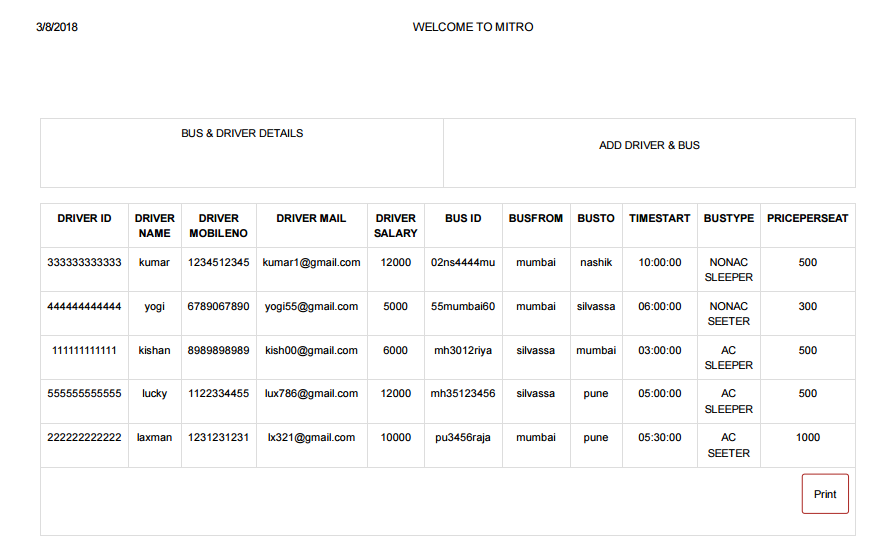


**Add bus detail and driver detail page:-**

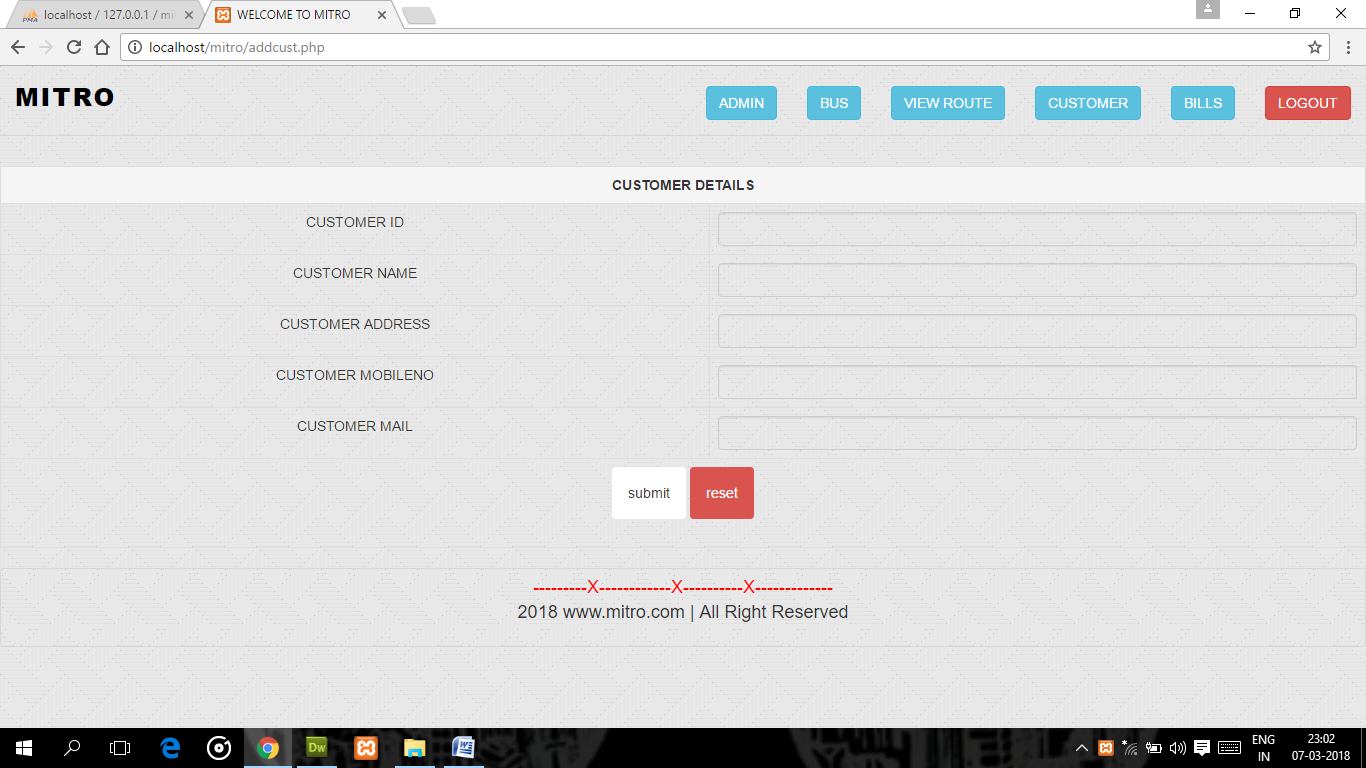


**View bus and driver detail page:-**

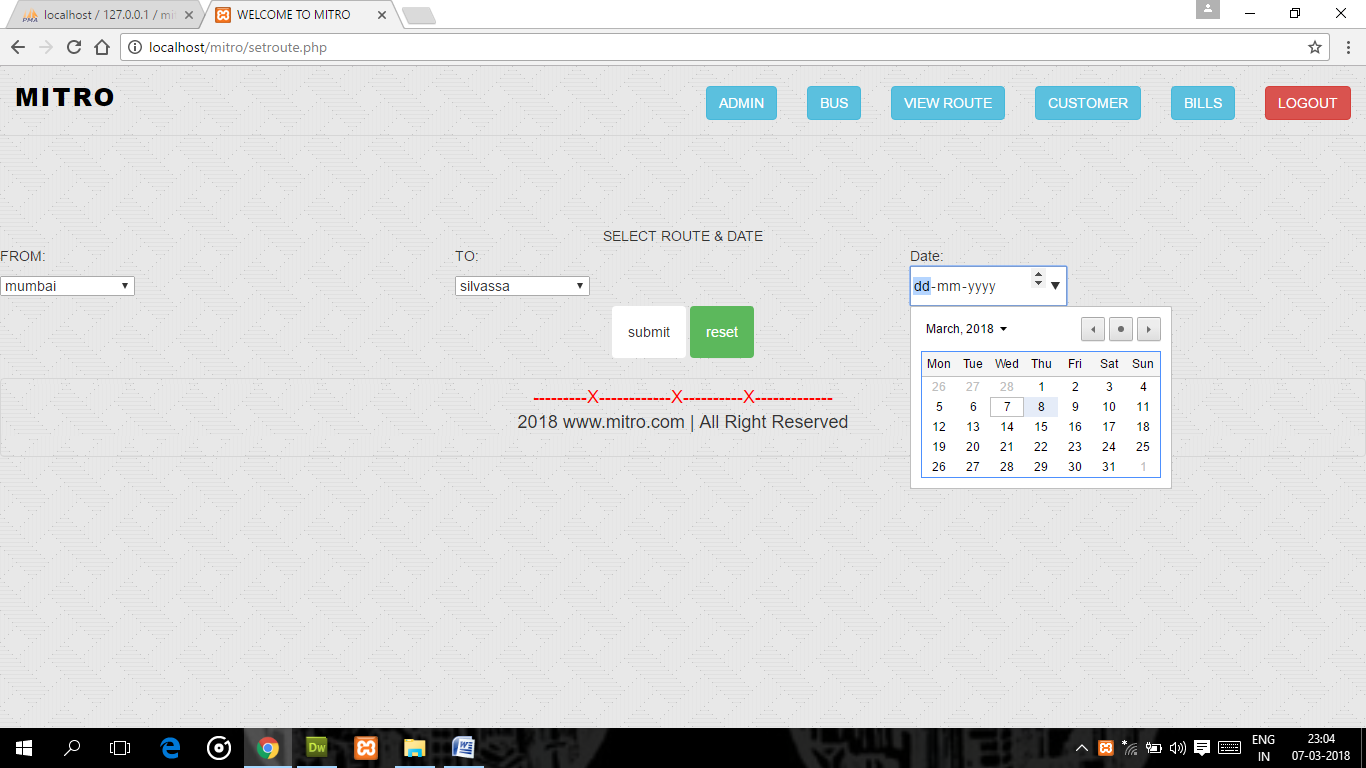
**Print The Bus& Driver Details:-**



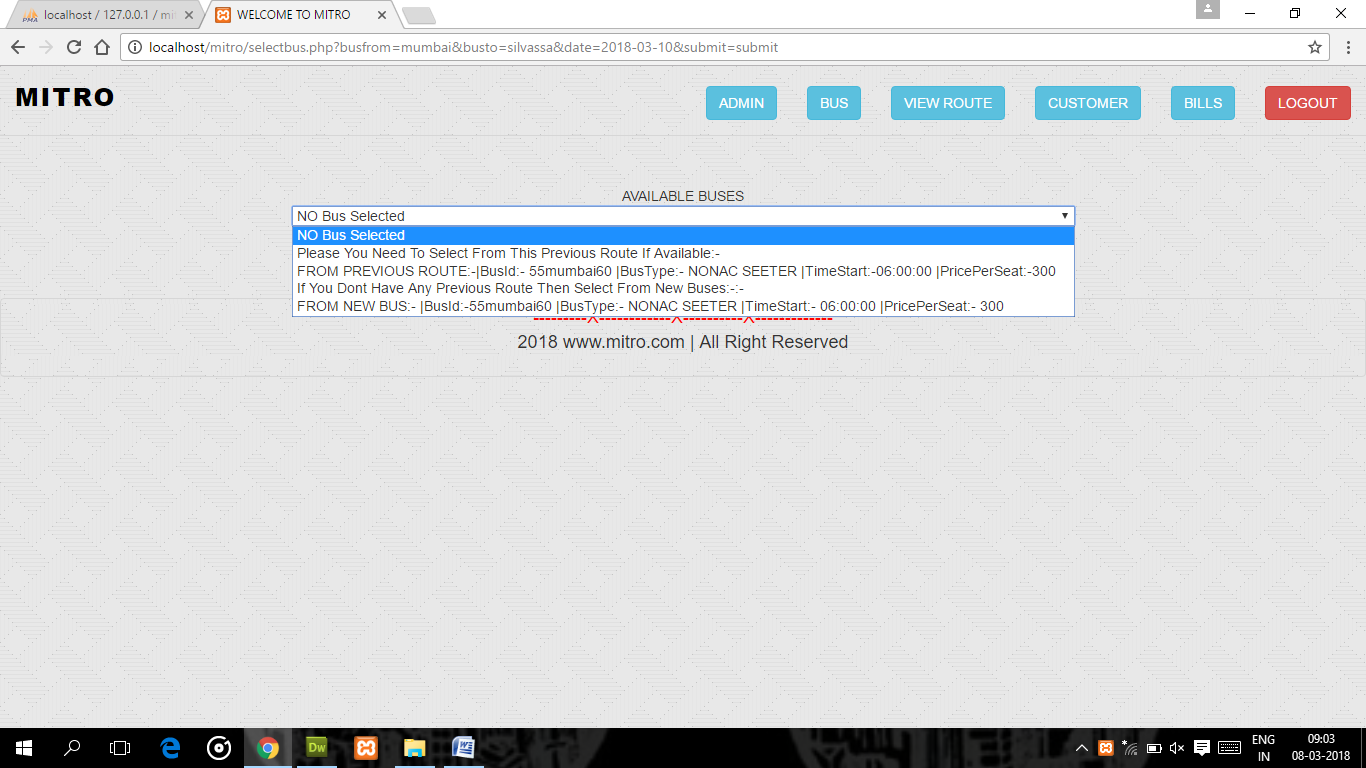
**Now we are going for Booking & First Add customer page:-**



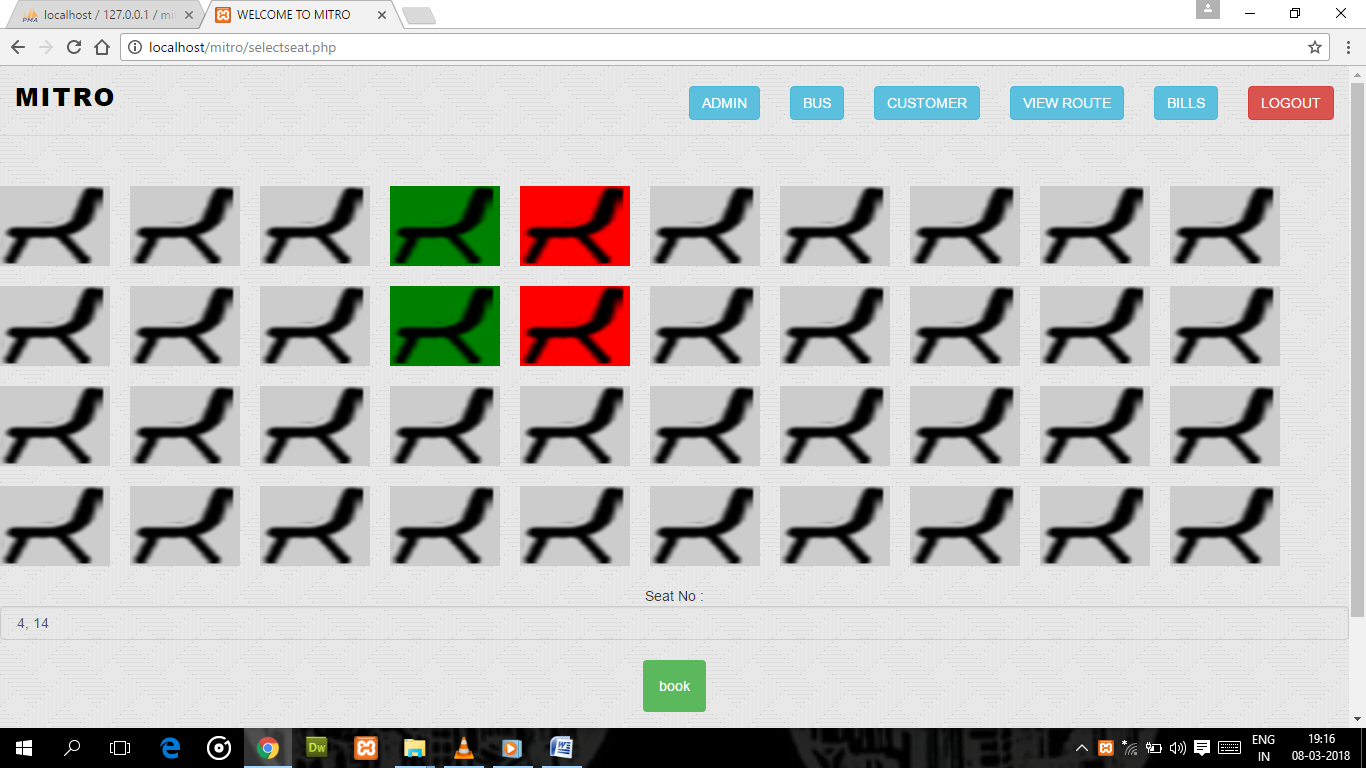
**Route selection page:-**



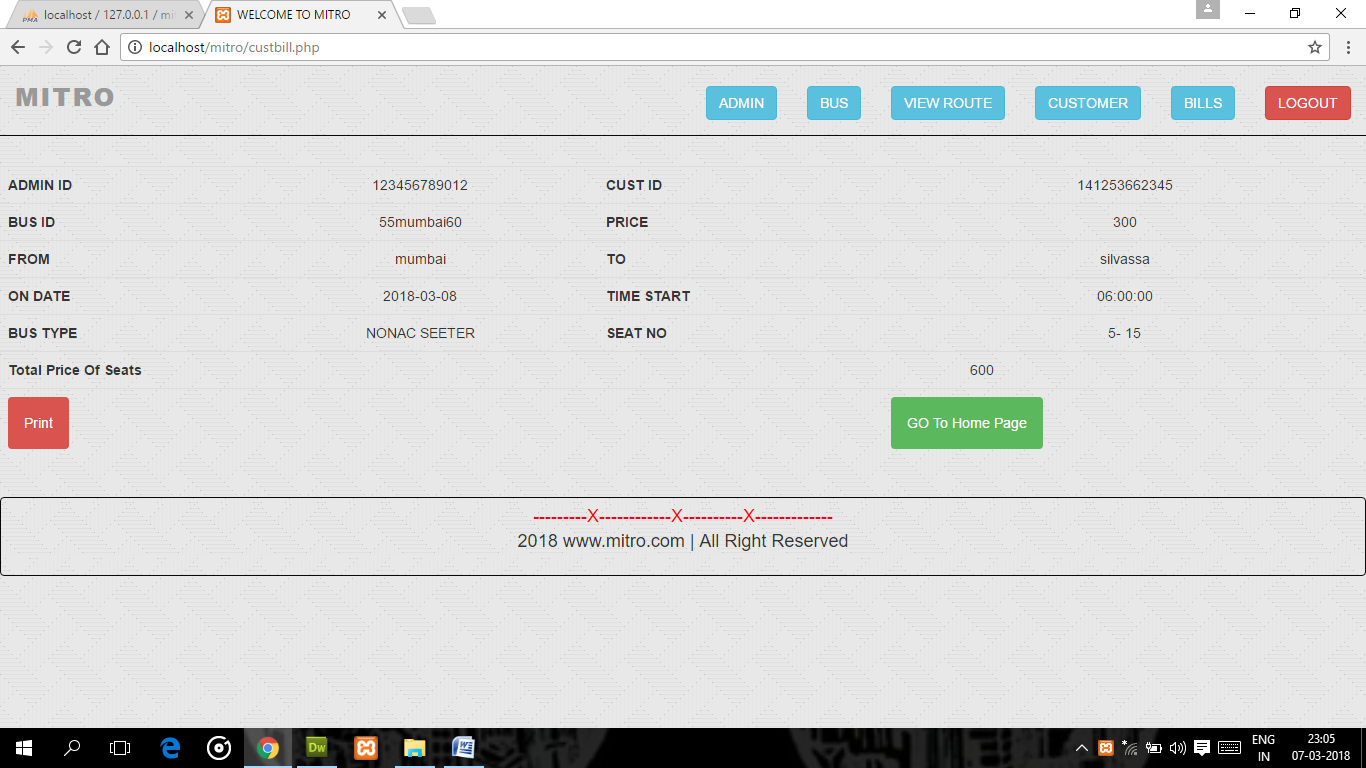
**Bus selection page:-**



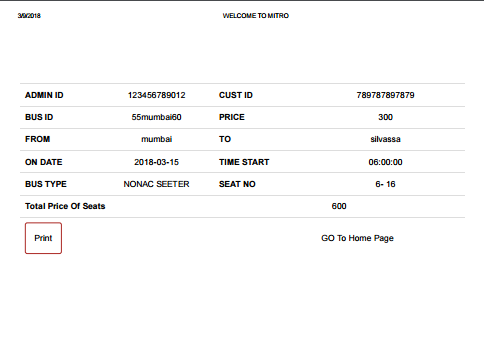
**View Booked Seat OR Seat selection page:-**

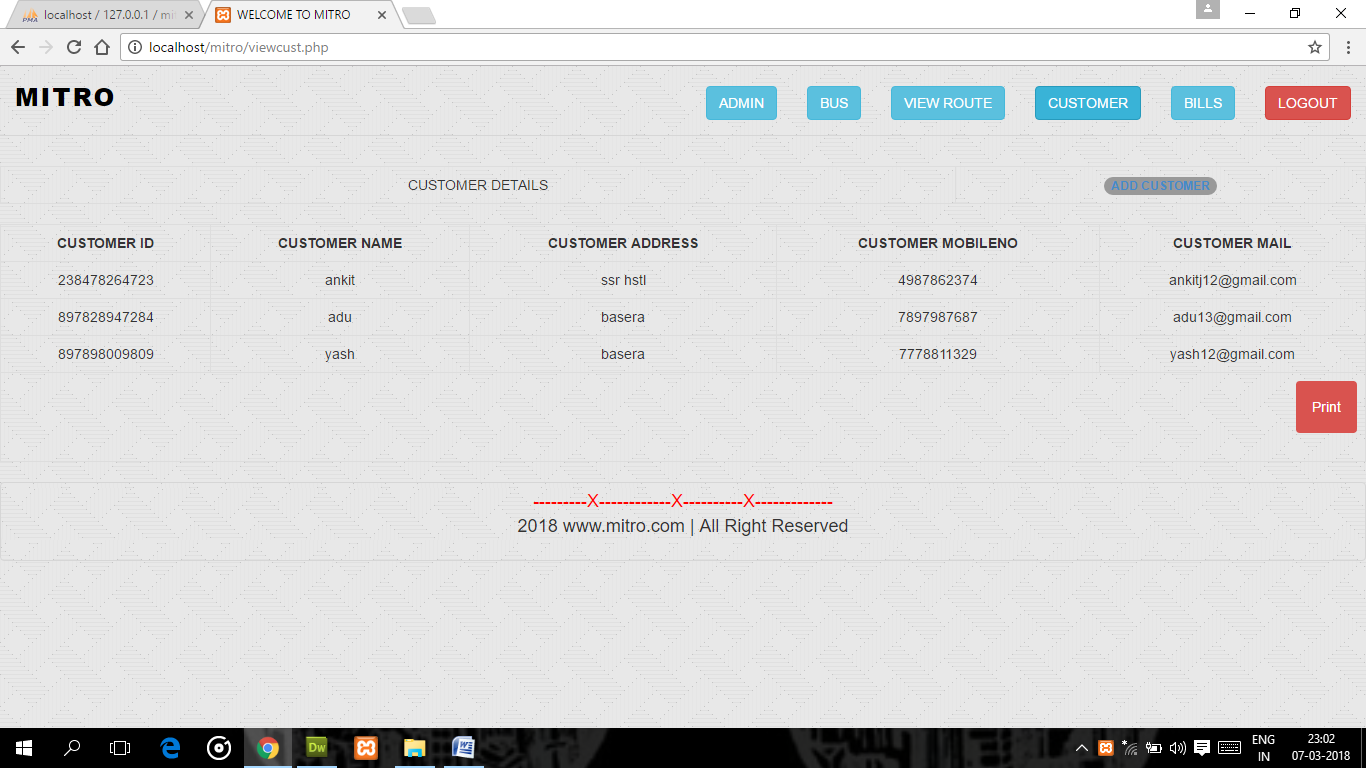


**Customer bill page:-**

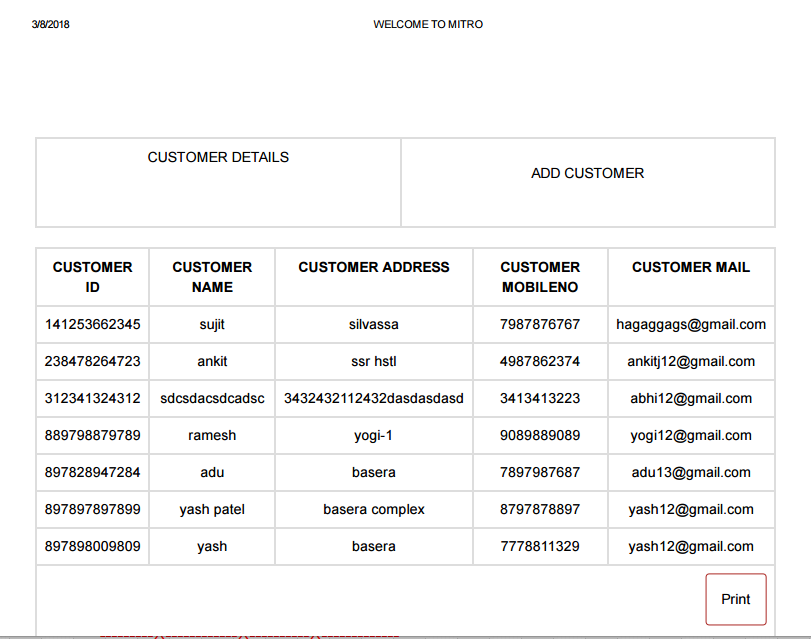


**Print OF Customer Bill:-**

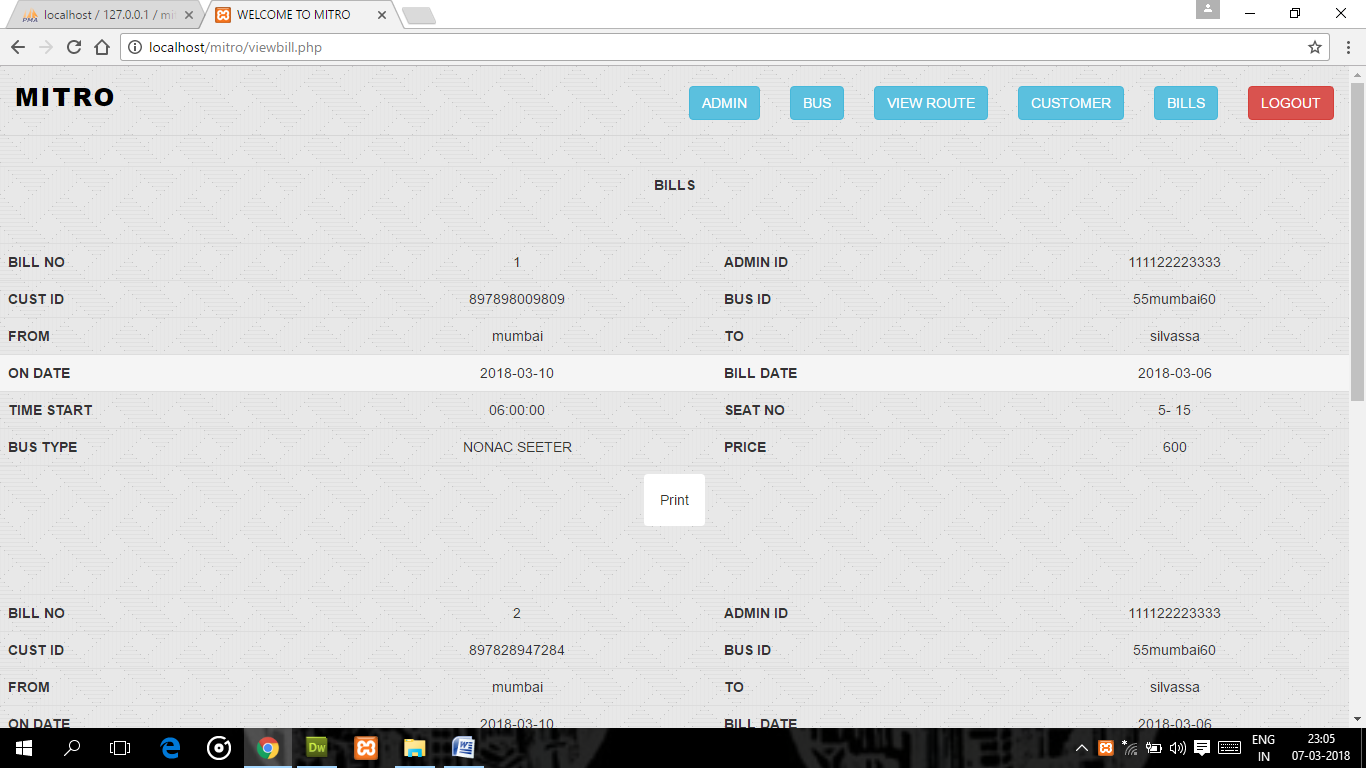
****

**View Customer Details page:-**

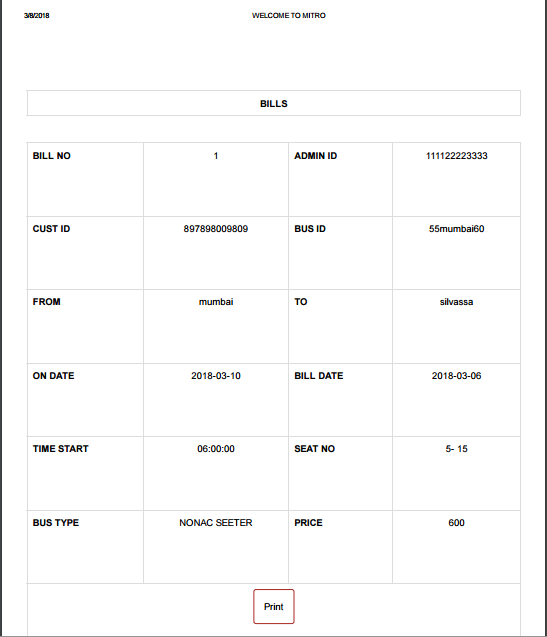
**Print Customer Details:-**

****

**View Bill Details page:-**



**Print Of Bill Details page:-**

****

**CONCLUSION**

Hereby we conclude that it was a nice experience to work with the Professors. This will be helpful when we will work in industry & educational field there we can put it in our practice.

We sincerely think that the knowledge & information made available was excellent, and it will definitely be useful for us in upcoming days.

As a student we got theoretical as well as Practical knowledge about the Software & Hardware subject, different software’s used, maintenance which was available. Last but not the least the different software’s used helped us in adding & improving our practical knowledge about what we have studied in our syllabus.

**BIBLIOGRAPHY**

For the creation and development of this project, we had to refer and make a keen study on many information sources.

So, we are really thankful to our lecturers who had been a great source of knowledge for our project.

**SOME OTHER SITES:**

* [**www.stackoverflow.com**](http://www.stackoverflow.com)
* [**www.youtube.com**](http://www.youtube.com)
* [**www.w3school.com**](http://www.w3school.com)
* **Software Engineering : A Practitioner’s Approach**

THANK YOU