**SHREE SHAMBHUBHAI V. PATEL COLLEGE OF COMPUTER**

**SCIENCE AND BUSINESS MANAGEMENT**

**VEER NARMAD SOUTH GUJARAT UNIVERSITY, SURAT.**

**PROJECT REPORT**

**ON**

**“Gujarat RTO”**

AS PARTIAL REQUIREMENT FOR THE DEGREE

OF

BACHELOR OF COMPUTER SCIENCE

**[B.C.A]**

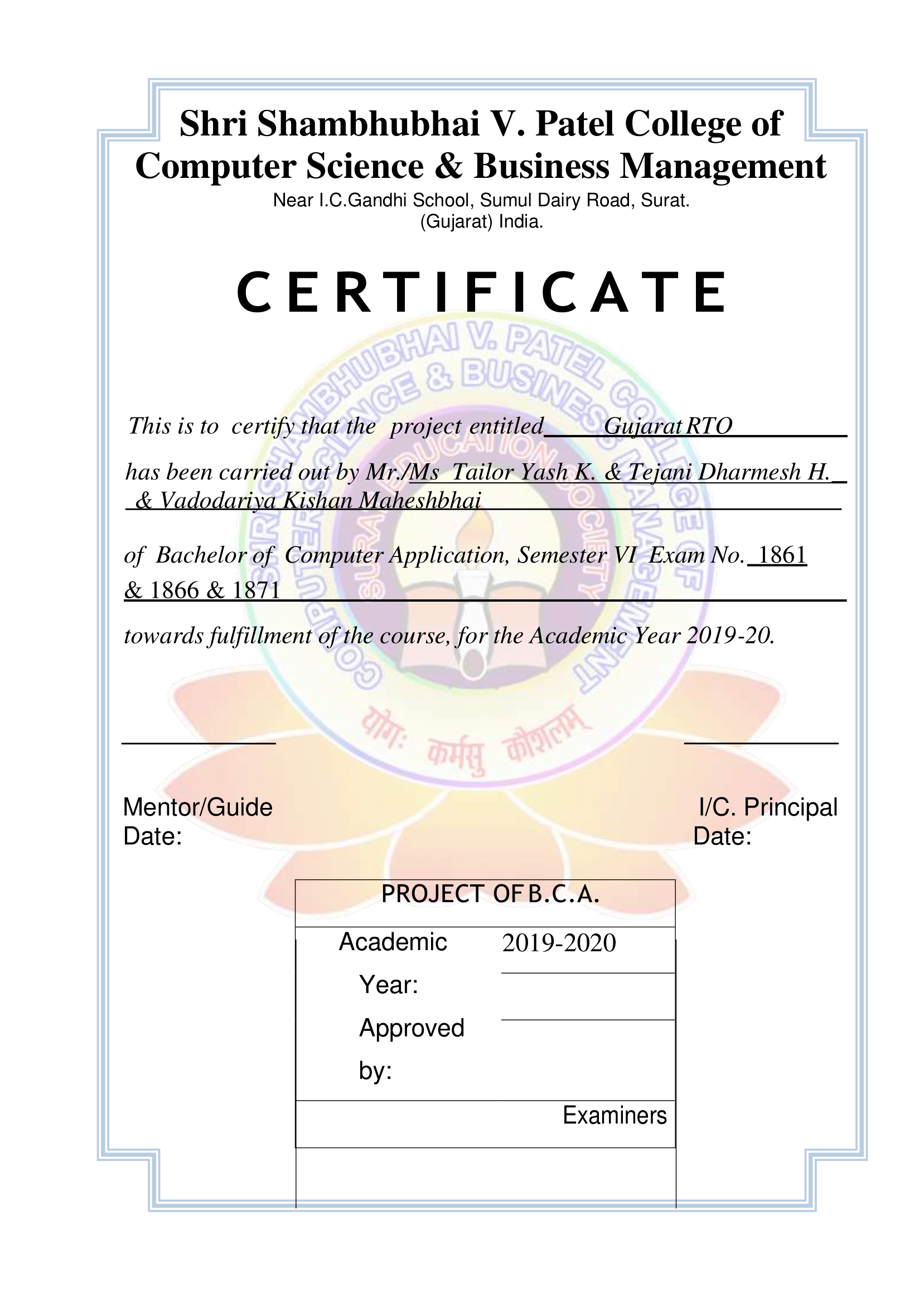
**YEAR – 2019 – 2020**

**SUBMITTED BY: GUIDED BY:**

**Tailor Yash K. (1866) Mrs. Binal Naik**

**Tejani Dharmesh H. (1871)**

**Vadodariya Kishan M. (1876)**

****

# ACKNOWLEDGMENT

First of all, I am indebted to the GOD ALMIGHTY for giving me an opportunity to excel in our efforts to complete this Project on time.

I am extremely grateful to **MR. CHINTU THAKKAR,** Principal of **SHRI SHAMBUBHAI V.PATEL COLLEGE OF COMPUTER SCIENCE & BUSINESS MANAGEMENT** and **PROF. RIDDHI VYAS,** Head of Department, Department of Computer Science, for providing all the required resources for the successful completion of our project .

My heartfelt gratitude to our project guide **PROF BINAL NAIK,** for valuable suggestion and guidance in the preparation of the project.

We would be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred to in this project.

.

Table of Contents

[ACKNOWLEDGMENT I](#_Toc43147083)

[ABSTRACT IV](#_Toc43147084)

[1. Introduction 1](#_Toc43147085)

[2. SYSTEM ENVIRONMENT 3](#_Toc43147086)

[**2.1 Hardware and Software Requirement** 4](#_Toc43147087)

[3. Problem Specification 5](#_Toc43147088)

[**3.1 Introduction, Objective and purpose** 6](#_Toc43147089)

[**3.2 System Model** 7](#_Toc43147090)

[3.3 Feasibility study 9](#_Toc43147091)

[3.3.1 Economic Feasibility 9](#_Toc43147092)

[3.3.2 Technical Feasibility 10](#_Toc43147093)

[3.3.3 Behavioral Feasibility 10](#_Toc43147094)

[4. Risk Identification 11](#_Toc43147095)

[And Management 11](#_Toc43147096)

[4.1 Risk Monitoring 12](#_Toc43147097)

[5. Technology Used 14](#_Toc43147098)

[For Development 14](#_Toc43147099)

[Characteristics of PHP 15](#_Toc43147100)

[6. Planning 20](#_Toc43147101)

[6.1 System Planning 21](#_Toc43147102)

[6.2 Time Line Chart 23](#_Toc43147103)

[7. System Analysis 24](#_Toc43147104)

[7.1 Database Design 25](#_Toc43147105)

[7.2 ER-Diagram 26](#_Toc43147106)

[7.3 Data Flow Diagram 27](#_Toc43147107)

[**7.4 Data Dictionary** 34](#_Toc43147108)

[8. Designing 59](#_Toc43147109)

[8.1 User Interface 60](#_Toc43147110)

[9. Testing 69](#_Toc43147111)

[9.1 Software Testing 70](#_Toc43147112)

[9.2 Unit Testing 71](#_Toc43147113)

[9.3 System Testing 72](#_Toc43147114)

[10. Reference 74](#_Toc43147115)

# ABSTRACT

Gujarat RTO project is prepared for RTO office to maintain all records like 2 wheeler registration, 3 wheeler registrations, LMV, HMV, learning license and driving license, changing of address, renewal form and much more. These are the main activities of RTO office. The administrator is a power user, he has the power to verify the data and provide appropriate solutions. By introducing the new system we have been organized some striking facilities. Registration of vehicle through online. Fancy number selection of vehicles through online. Issues of information about the licence, which includes application forms and license test and other information. In the existing system of RTO office perform functions such as registration, license and fitness. Regional transport office is the organization of the Indian government responsible for maintaining a database of vehicles for various states of India. The RTO issues driving licenses organizes a collection of vehicle excise duty and sells personalized registrations. These are the main activities of RTO office; we developed this software application with a fully computerized method to manage all the data. At present all records are maintained manually.

# 1. Introduction

|  |  |
| --- | --- |
| Project Title | RTO.com |
| Project Definition / Objectives | To design, Develop, Test and implement the “RTO” in php core |
| Project Guide | Mrs. Binal Naik |
| Team Size | 3 Member |
| Team Member | Tejani Dharmesh  Vadodariya Kishan  Tailor Yash |
| Language | PHP Core |
| User Interface | Core(7.2) |
| Back End | MySQL |
| Documentation Generation Tool | Microsoft Word 2013 |
| Project Duration | 3 Months |
| Submitted To | Shri Shambhubhai V. Patel College of Computer Science and Business Management |

# 2. SYSTEM ENVIRONMENT

## **2.1 Hardware and Software Requirement**

**Hardware Requirement**

|  |  |
| --- | --- |
| MicroProcessor | Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz 2.00 GHz |
| Harddisk | 1 TB |
| RAM | 4 GB |
| Screen | 15” |

**Minimum Hardware Configuration**

|  |  |
| --- | --- |
| MicroProcessor | 850 MHz Intel Celeron |
| Harddisk | 1 GB |
| RAM | 64 MB |
| Keyboard | Simple Keyboard |
| Mouse | Simple Mouse |
| Monitor | 14” Monitor |
| Operating System | Windows 7 |

**Software Requirement**

|  |  |
| --- | --- |
| Operating System | Windows 7 |
| Front-End | Bootstrap,HTML |
| Back-End | PHP Core |
| Other | Sublime v3.2.2 |

# 3. Problem Specification

## **3.1 Introduction, Objective and purpose**

**Introduction:-**

RTO stands for Regional Transport Office. It is an Indian Government bureau which is responsible for registration of vehicles. It is situated in every state of India which issues driving licenses and vehicle registration cards. Almost every state of India has the Regional Transport Offices in most of the districts.

This system is also interacting with exotic & offbeat destinations in world. There are two modules involves with in this system.

1) Admin side Handling

2) User side Handling

**Purpose:-**

* The proposed system is aimed to automate the major processes in the Regional Transport Office
* Online RTO Management System for different License and Vehicle Registration requirement

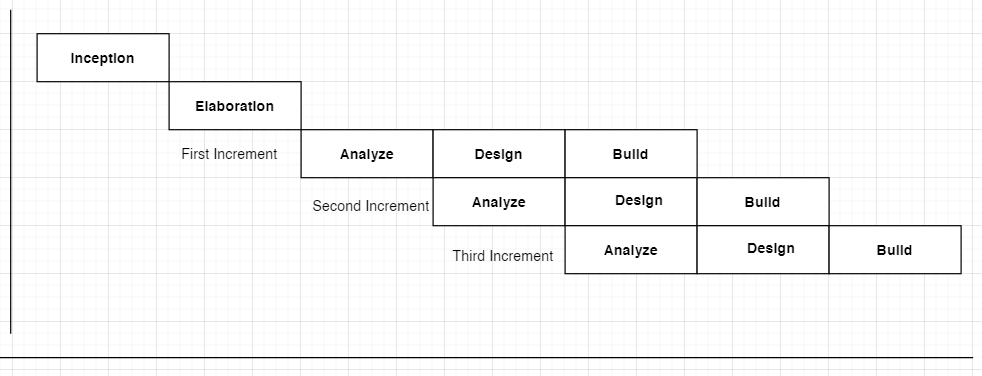
**Objective:-**

* + The main objective of the project RTO, user is to book Online Exams Date, Vehicle Registration number, license, RC Book.
  + The Administrator is providing for authentication purpose as well as it handles all the database of RTO and manages all the process
  + It help to reduce the RTO work manually and it helps to save the time of the user
  + This process intends to help the customer in saving their time
  + It also helps RTO officials to maintain records systematically and reduce a lot of paper work

## **3.2 System Model**

**Requirement Gathering:**

**Incremental Model:**



The incremental approach attempts to combine the waterfall sequence with some of the advantages of prototyping. Incremental model in software engineering is a one which combines the elements of waterfall modal in an iterative manner. It delivers a series of releases called increments which provide progressively more functionality for the client as each increment is delivered.

**Inception:**

In the incremental modal of software engineering, is repeatedly applied in each increment. The incremental modal applies linear sequences in a required pattern as calendar time passes. Each linear sequence produces an increment in the work.

**Elaboration:**

The first increment is often a core product where the necessary requirements are addressed, and the extra features are added in the next increments. The core product is used and evaluated by the client. Once the customer assesses the core product, there is plan development for the next increment. Thus is every increment, the needs of the client are kept in mind, and more features and functions are added, and the core product is updated. This process continues till the complete product is produced.

The increments earlier to the main increment are called as versions of the final product. These increases form a base for customer evaluation. On this basis, the client can suggest new requirements if required.

## 3.3 Feasibility study

All projects are feasible time. But in the real world that can be bare minimum. Especially provided that unlimited resources and infinite computer-based systems are likely to be bounded by the limited resources as well time. Feasibility and risk analysis is related in many ways. If the project risk is great, then feasibility of producing quality software reduced.

During system engineering, we concentrated our tension on three primary areas of interest.

Feasibility test basically focuses on answer to, how well our project does satisfy the four dimensions:

3.3.1 Economic Feasibility

3.3.2 Technical Feasibility

3.3.3 Behavioural Feasibility

Answer to these questions could be very easy for projects in established area where we get lots of time and resources to use and as the risks increases the feasibility study would result negative.

### 3.3.1 Economic Feasibility

Economic analysis is the most usually used method for estimating the effectiveness of a client system. In this feasibility study below points were discussed.

How to provide good quality of Service in less cost?

How to gain more income by keeping reasonable fees from member?

How can build proposed system in less cost with effective GUI?

How to separate the business logic from the interface?

### 3.3.2 Technical Feasibility

In Technical and system feasibility study we discussed on below points.

🡪Analyzing about performance of server compare to proposed system.

🡪Which technology to use, as a Font-End & Back-End, for proposed system?

🡪Which operating system if more efficient? Or Generalize OS for Portability.

🡪Security of data in server.

### 3.3.3 Behavioral Feasibility

#### The new system can be beneficial only if it satisfies the organization requirements; in such a way that resource utilization and optimum outcome is justified. A new system should not only be robust but should also be able to work simultaneously with other systems. Operational feasibility means that new system should not affect any existing system during the development phase or even in the implementation phase. Following are some points underlying the operational feasibility of the system- As the development proceeded many doubts got cleared out.

#### Our project guide –Prof. Binal Naik guided us to take proper care and check for the operational feasibility of the system.

#### Efforts were made to optimize the human efforts in data collection, storage, retrieval, security and presentation.

#### The proposed system made best efforts in achieving necessary function and performance, as required by the user and keeping in mind some infrastructure constraints.

# 4. Risk Identification

# And Management

## 4.1 Risk Monitoring

After gathering basic requirement, feasibility study and studying functionality of proposed system, we assessed collected project data for possible risks and found following possible risks for project.

**POSSIBLE RISKS:**

As all team members are not much experienced in professional software development, our team might have difficulties in building elegant solution. We might have specified or decided to provide some requirement that we are not sure of fulfilling and that may lead to confusion and schedule problems during later stages in project. Project might suffer from lack of schedule due to possible breaks in schedule caused by other activity like seminar preparation etc.

**ACTION TAKEN:**

We are trying to study more and more professionally developed online applications so we can include maximum features. We have decided to evaluate specifications at regular intervals for feasibility and refine it if necessary. We have prepared project schedule in such way that we get enough time for project work along with other preparations.

**RISK TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risks** | **Probability** | **Impact** | **Type** |
| Website may not be well protected from piracy | 70% | High | Known Risk |
| Website may not fit environment for which it is design | 50% | High | Known Risk |
| Required resources may not be availability | 20% | Medium | Known Risk |
| Technical knowledge defect | 35% | Medium | Known Risk |
| Requirement gathering defect | 25% | High | Known Risk |
| Project not complete in given time | 10% | Medium | Known Risk |

# 5. Technology Used

# For Development

**PHP Core**

**PHP** started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

**PHP** is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning PHP:

* PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
* PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
* It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
* PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
* PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
* PHP is forgiving: PHP language tries to be as forgiving as possible.
* PHP Syntax is C-Like.

Characteristics of PHP

Five important characteristics make PHP's practical nature possible −

* Simplicity
* Efficiency
* Security
* Flexibility
* Familiarity

**PHP**

**PHP** stands for: HyperText Preprocessor. The food was originally named Personal Home Page Tolls. PHP is a server side scripting language, which can be embedded in HTML or used as standalone binary. Sometimes it’s also called as Open source.

**Java-Script**

JavaScript is a programming language that can be included on web pages to make them more interactive. You can use it to check or modify the content of forms, change image, open new windows and write dynamic page content. You can even use it with CSS to make DHTML (Dynamic HyperText Markup Language). This ickd the page.

**Ajax**

The term Ajax is used to describe a set of technologieo is that allows browsers to provide users with a more natural browsing experience. Before **Ajax,** web sitesforced their users into the submit/wait/redisplay paradigm, where the users actions were always synchronized with the server’s “think time”. Ajax provides the ability to communicate with the server asynchronously, thereby freeing the user experience from the request/response cycle. With Ajax, when a user clicks a button, you can use JavaScript and DHTML to immediately update the UI, and spawn an asynchronous request returns, you can then use JavaScript and CSS to update the UI accordingly without referencing the entire page. Most importantly, users don’t even know your code is communicating with the server, the website feels like it’s instantly responding.

**XAMPP**

XAMPP stands for Cross-Platform (X), Apache (A), MySQL (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing purposes. Everything you need to set up a web server - server application (Apache), database (MySQL), and scripting language (PHP) - is included in a simple extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac and Windows. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server is extremely easy as well.

**CSS**

Cascading Style Sheets (CSS) is a W3C standards for defining the presentation of web documents. Presentation refers to the way a document is displayed or delivered to the user, whether it’s on a computer monitor, a cell phone display, or read aloud by a screen reader. This book focuses primarily on the visual aspects of presentation, such as typography, colors, alignment, layout, and so on. CSS is the mechanism for providing these types of style instruction to elements in a document that has been marked up with XHTML, HTML, or any XML language. Most important, CSS keeps these presentation instructions separate from the content and its structural and semantic markup.

**The Benefits of CSS**

* **Greater typography and page layout controls**
* With style sheets, you can specify traditional typography features that you could never do with HTML alone (even with its presentational extensions).
* **Less work**
* Not only can you format all similar elements in a document with a single style rule, external style sheets make it possible to edit the appearance of an entire site at once with a single style sheet edit.
* **Potentially smaller documents**
* Redundant font tags and nested tables make for bloated documents. Stripping presentational HTML out of the document saves on file size.

MySQL



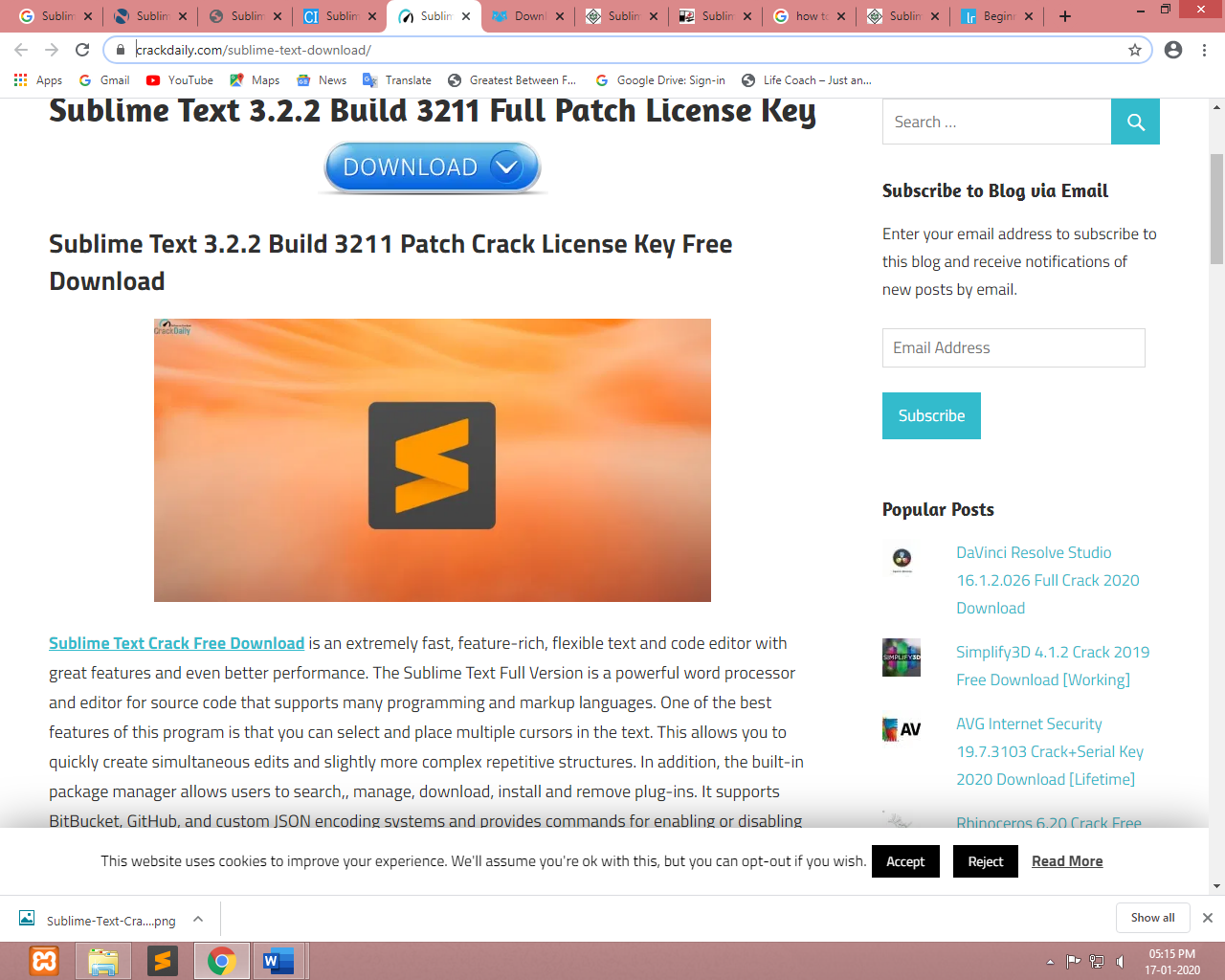
MySQL is a database system used on the web. Basically, a MySQL database allows you to create a relational database structure on a web-server somewhere in order to store data or automate procedures. If you think of it in comparison to Microsoft Access , MySQL is what holds all of your tables , PHP acts as your queries(among other things),and your forms are basically web pages with fields in them. With all of this combined, you can create truly spectacular projects on the web.

MySQL is open source in that it’s free and falls under the GNU General Public License(GPL).Chances are , if you are getting your own web-pages or already associated with(through not limited to) Unix/Linux based servers. If by chance you are considering getting your own page and want MySQL and PHP support, check out Dream host-We’ve been using them for years and they absolutely can’t be beat.

Interacting MySQL database is a little weird as you don’t have the tried and true WYSIWYG interface that something as easy Microsoft Accords affords, PHP My Admin gives you an easy-to-use interface that you to create at allows you to create tables and run queries by filling in a little bit of information and then having the tables created for you. This is good if you’re either lazy, or don’t feel like bothering with big and complicated SQL statements.

* MySQL is a database server.
* MySQL is ideal for both small and large applications
* MySQL supports standards SQL
* MySQL compiles on a number of platform.
* MySQL is free to download and use.

**Sublime (IDE 3.2.2):**



* [**Sublime Text**](https://crackdaily.com/sublime-text-download/)  is an extremely fast, feature-rich, flexible text editor for editing local files or a code base with great features and even better performance. The Sublime Text Full Version is a powerful word processor and editor for source code that supports many programming and markup languages. One of the best features of this program is that you can select and place multiple cursors in the text. This allows you to quickly create simultaneous edits and slightly more complex repetitive structures. In addition, the built-in package manager allows users to search, manage, download, install and remove plug-ins. It supports BitBucket, GitHub, and custom JSON encoding systems and provides commands for enabling or disabling packets.
* Sublime Text editor is used as an Integrated Development Editor (IDE) like Visual Studio code and NetBeans. The current version of Sublime Text editor is 3.2.2 and is compatible with various operating systems like Windows, Linux and MacOS.
* Go to anything (switch files).
* Code reshaping and line merging.
* Automatic storage and customizable key connections.
* Code Generation (for specific languages).
* Autofill (depends on input).
* Supports MacOS, Windows and Linux.
* The direct access key / selection can be assigned.
* Repeat last action (simultaneously).
* Syntax reporting and eye relief.

# 6. Planning

## 6.1 System Planning



* **System Planning:-**

In this particular phase of the project we have explained the problem definition along with its requirements in detail. All the research work and background study prior the project initialization has been conducted this phase. The work breakdown structureand scheduling chart is prepared after conducting the feasibility study and risk analysis.

**Requirement Gathering:-**

Before developing elegant Website,best way to identify basic needs and functionality isto take interview of director and management who are working in that specific area addressed by proposed application.

For gathering basic needs and functionalities, we had meeting with chairman personally to understand required functionalities, also noted down some useful features to be included By analyzing the information we understand following process requirement for security system.

Here we have tried to gather all the requirements pertaining program code, user interface and database design. The detailed flow of data in the system is depicted by data flow diagrams which is assisted along with process specification and data dictionary. We have also shown the relationships between the entities of data through entities relationship diagrams. This particular phase leads milestone.

**User Interface Designing:-**

In website without login customer only sow information of RTO (Regional Transport Office). If customer has an account then no need to registration otherwise Customer must be registration.

**Implementation and Testing:-**

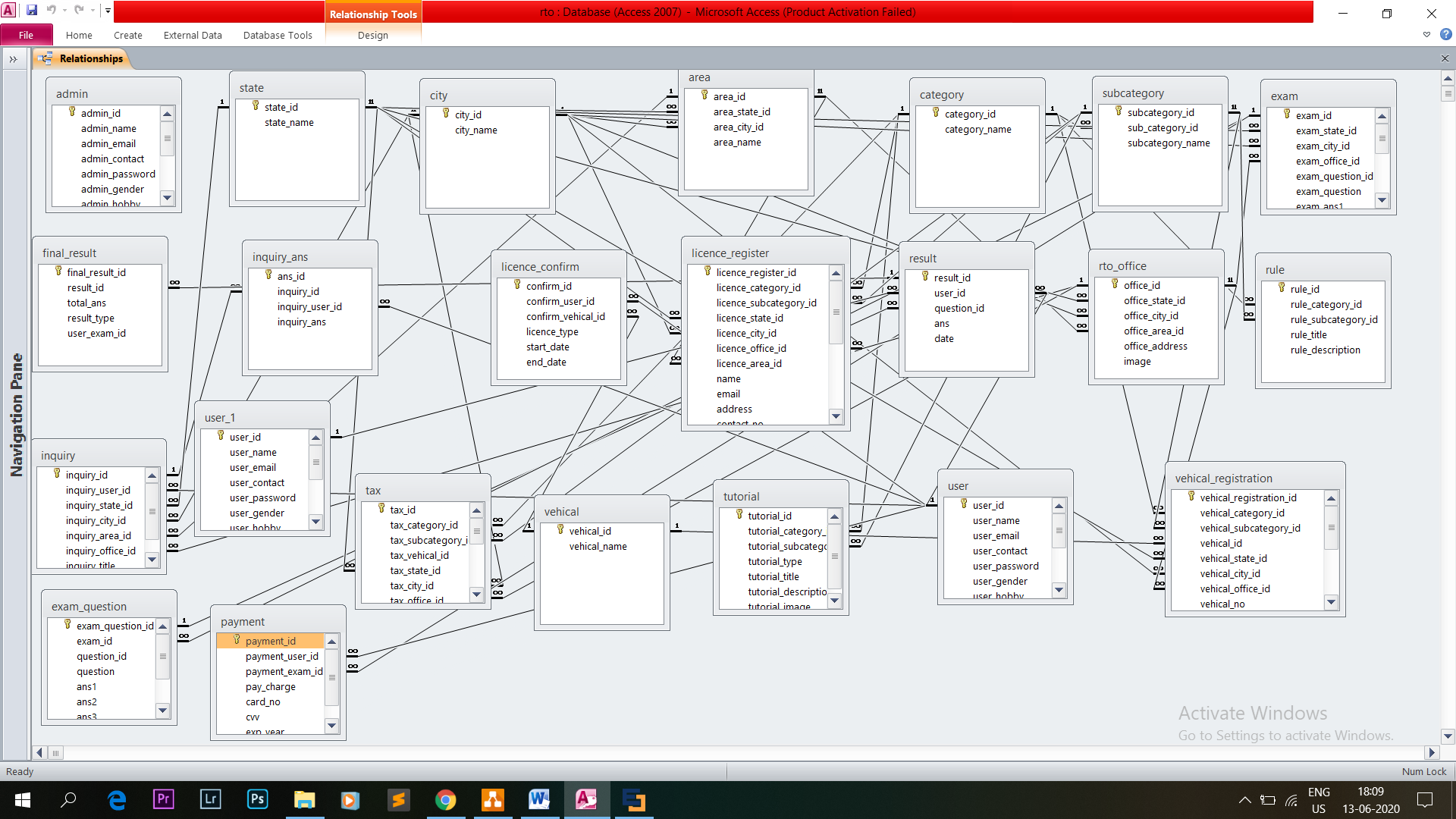
Depending upon the constraints and assumptions specified in the scope, we have implemented the system using latest PHP technology.

## 6.2 Time Line Chart

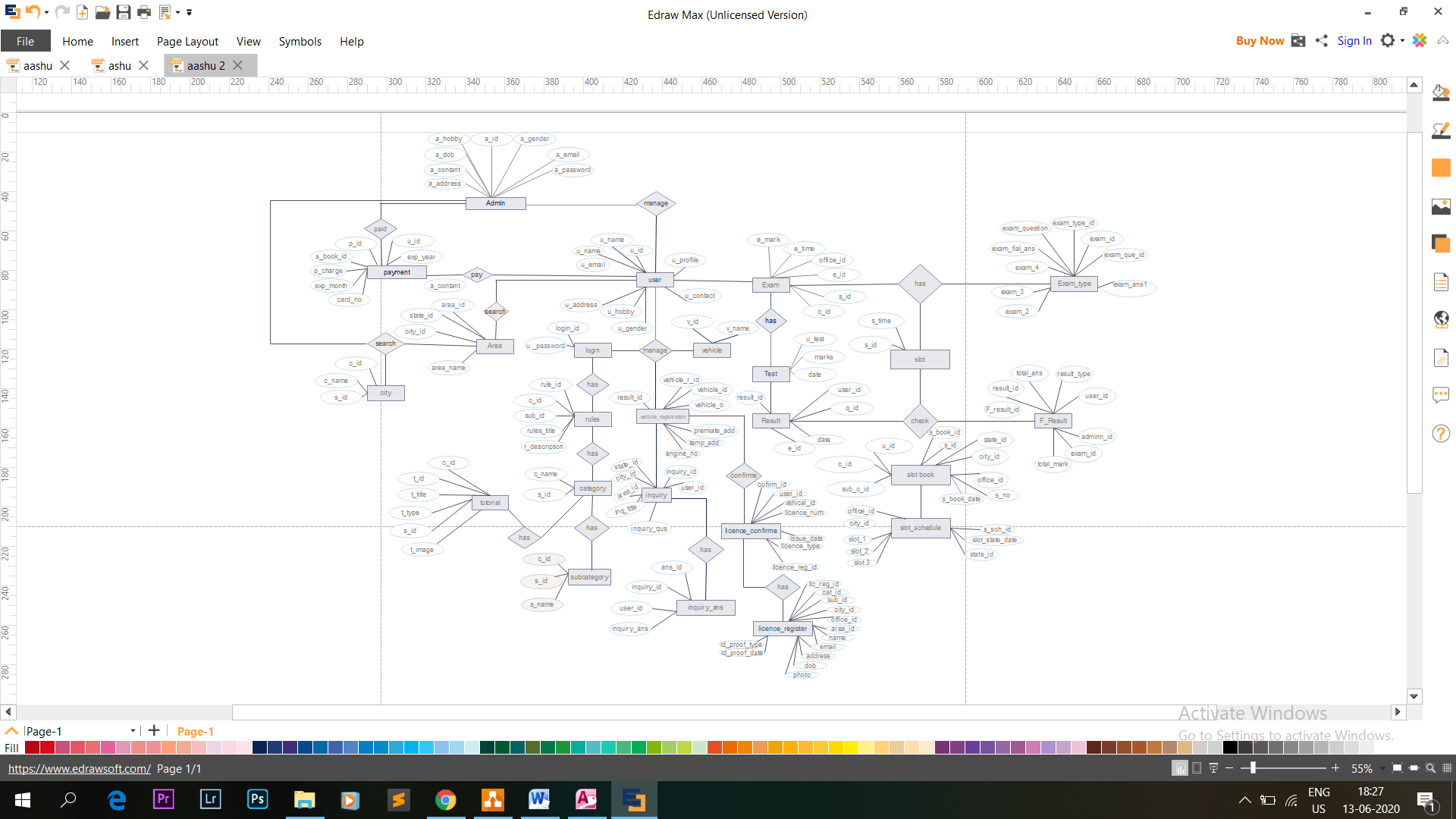
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Week | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | | 17 | | 18 | 19 | 20 | 21 |
| Month/  Week | December | | | | January | | | | February | | | | March | | | | | | April | | | |
| Requirement  Gathering |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Analysis |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Coding |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Designing |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |

# 7. System Analysis

## 7.1 Database Design



## 7.2 ER-Diagram



## 7.3 Data Flow Diagram

A Graphical Representation that depicts information flow and the transformation those are applied as data moves from input to output known as Data Flow Diagram. The DFD is also known as bubble chart the DFD may be partitioned into levels that represent increasing information flow and functional details. The 0th level DFD represent the Entire software elements as single bubble. As the 0th level DFD is partitioned, additional process are brought out to level more details.

**--:Symbols:--**

**Input and Output:**

Input and output are source and destination of information data. Entities are represented by a rectangles with their respective names.

**Processing:**

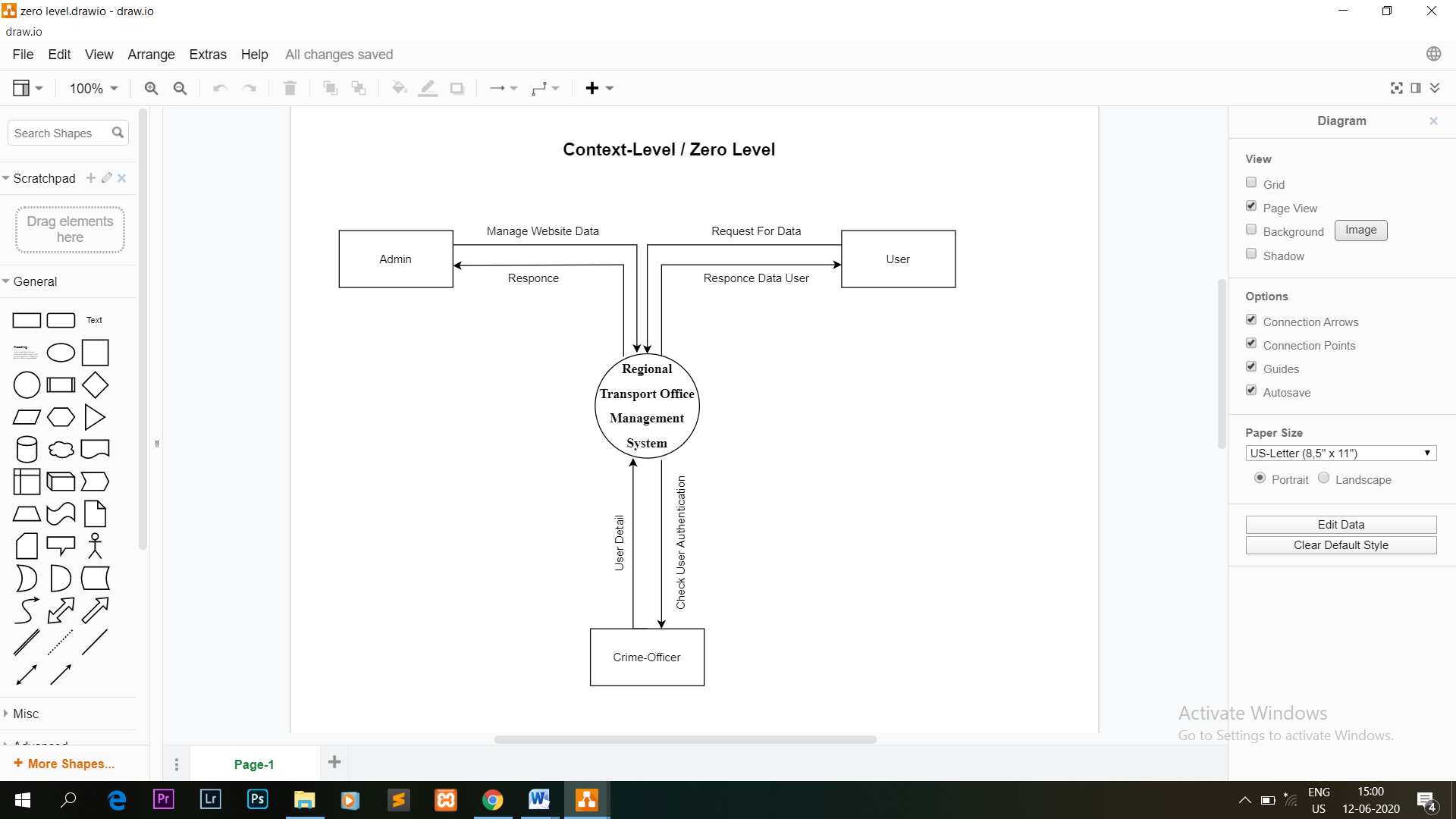
Activities and action taken on the data are represented by Circle or Round-edged rectangles.

**Flow of Data:**

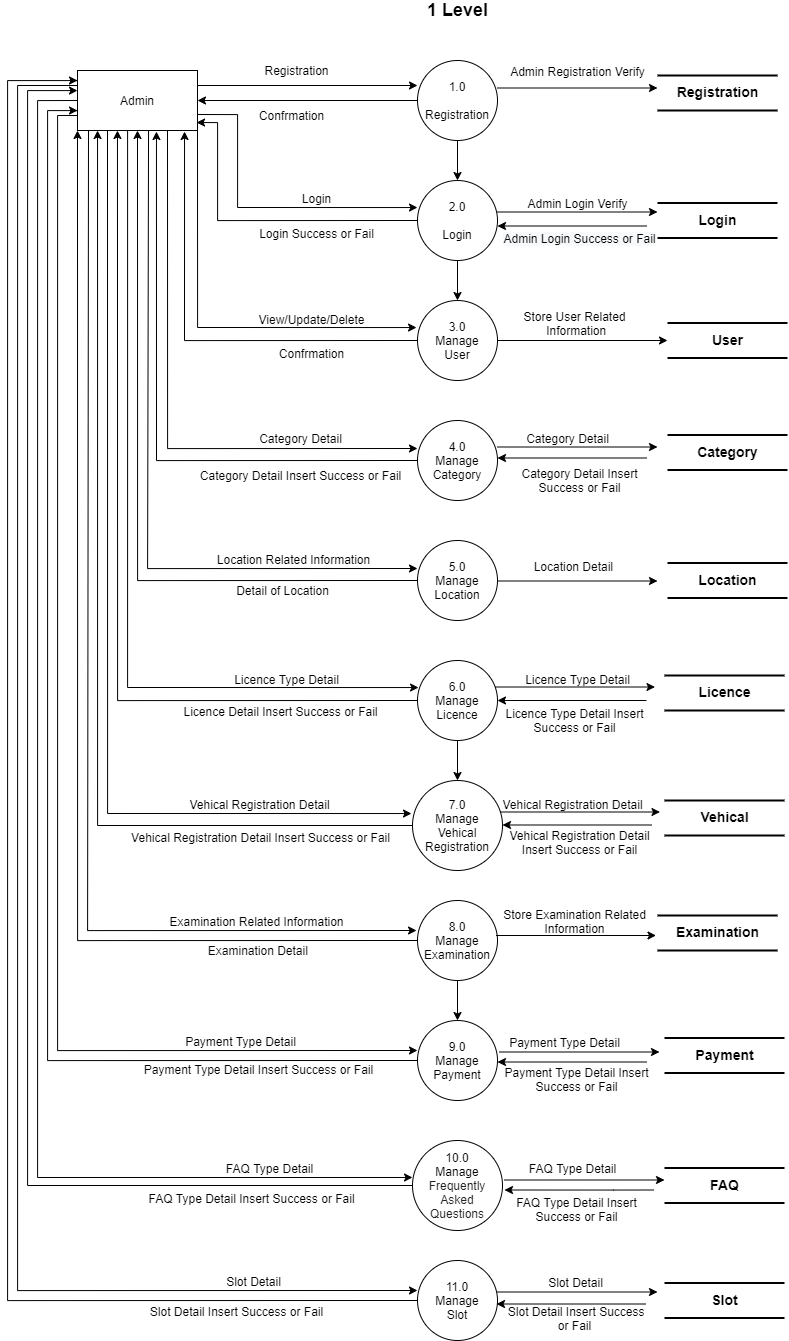
Movement of data is shown by pointed arrows. Data movement is shown from the base of arrow as its source towards head of the arrow as destination.

**Data Storage:**

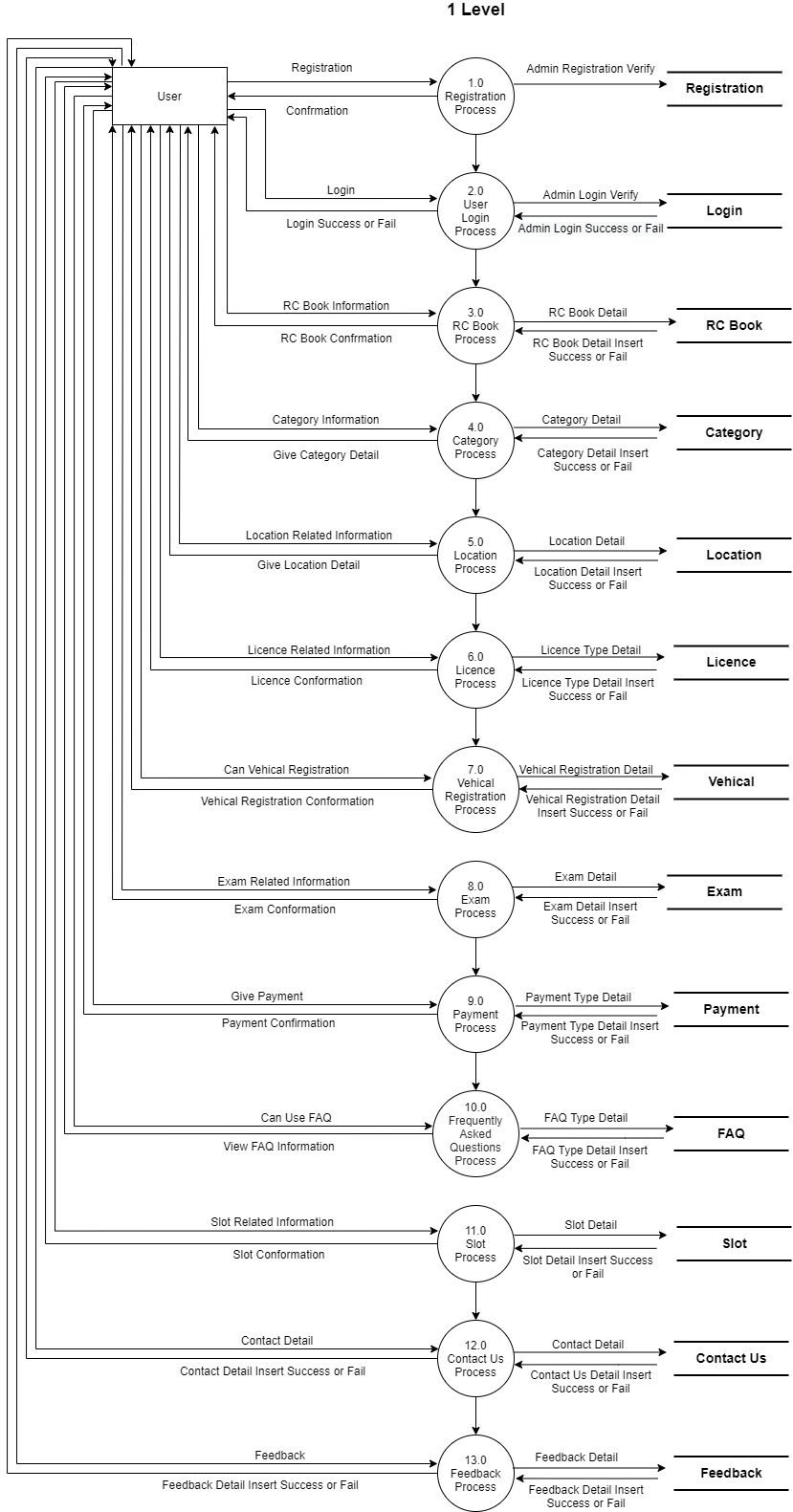
There are two variants of data storage - it can either be represented as a rectangle with absence of both smaller sides or as an open-sided rectangle with only one side missing.



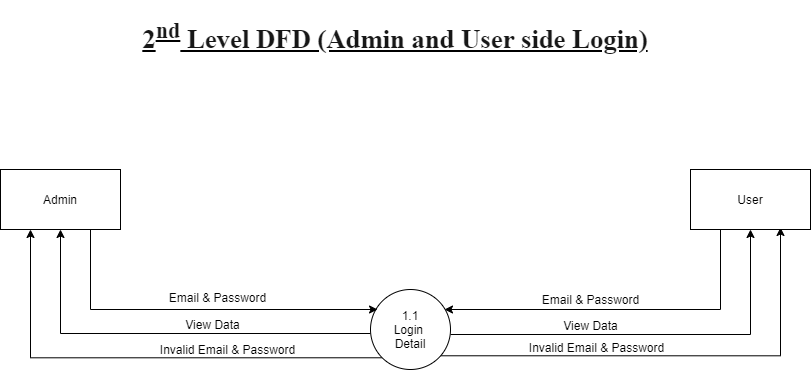
**Context Level**

****

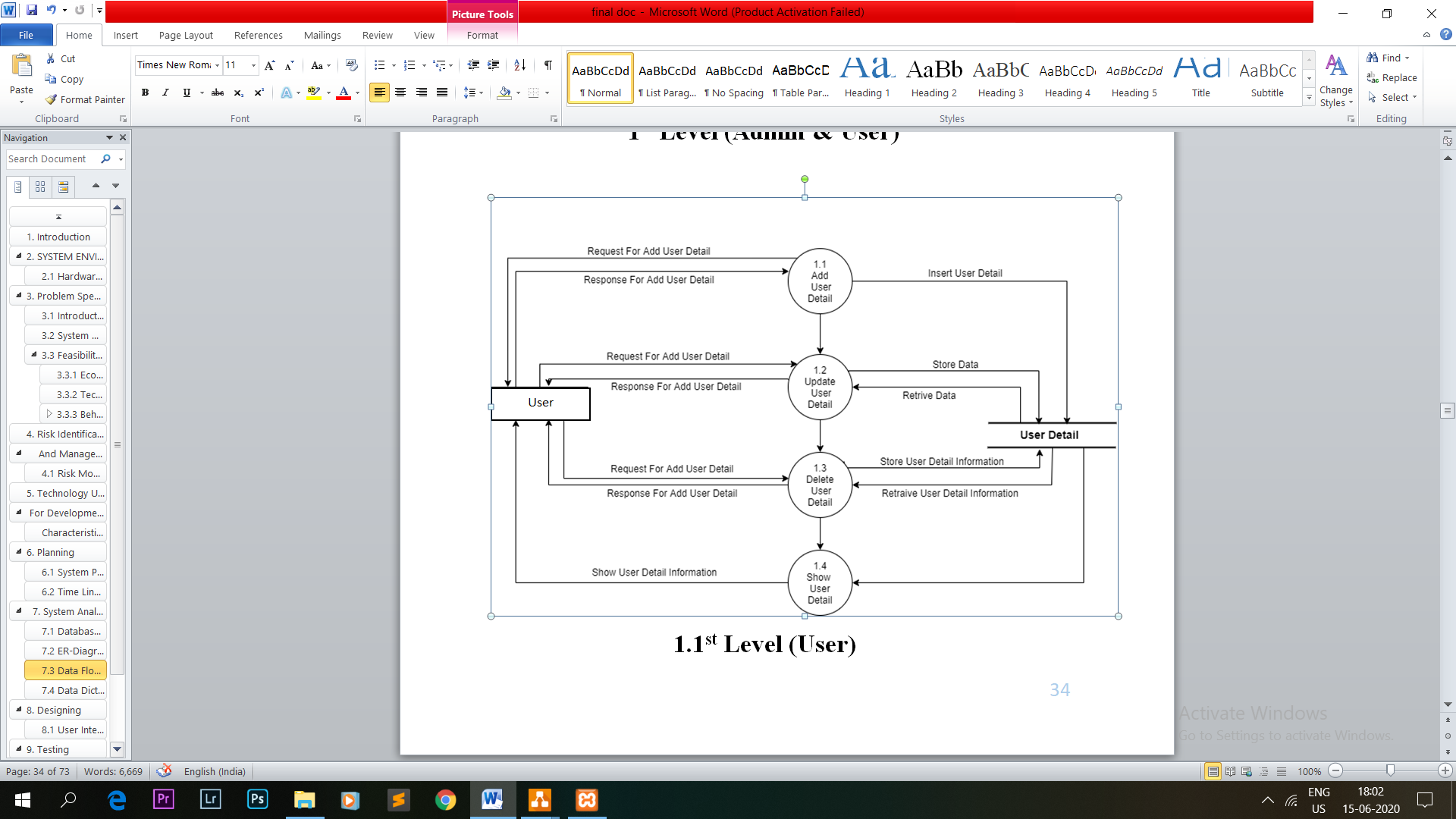
**Zero Level(Admin)**

****

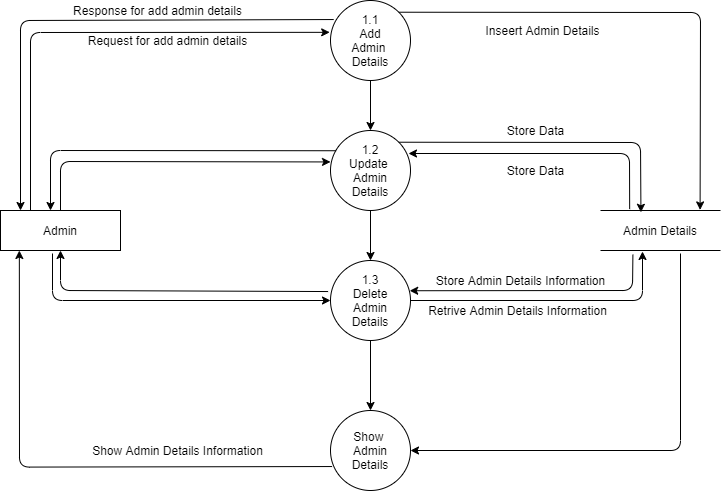
**Zero Level(User)**



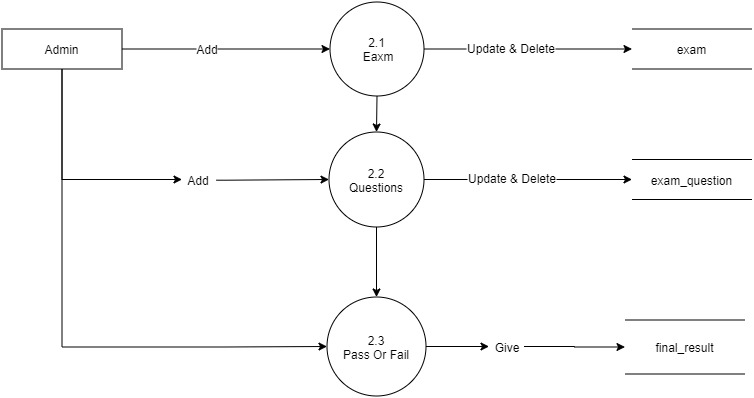
**1st Level (Admin & User)**



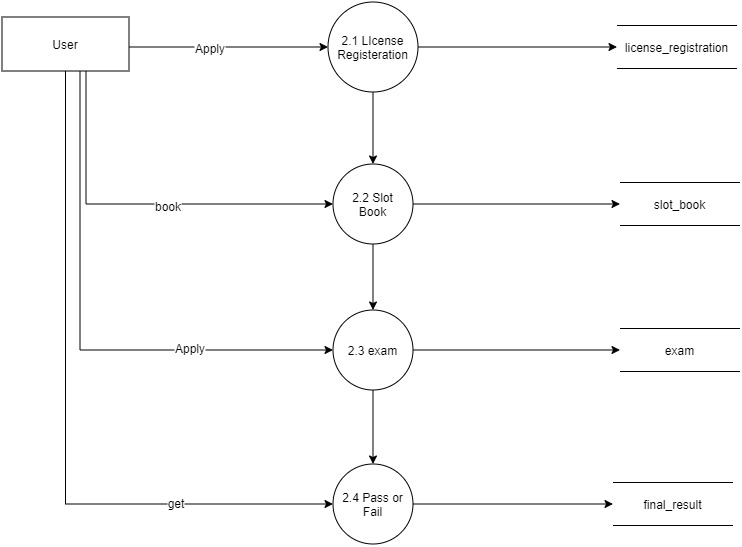
**1.1st Level (User)**

****

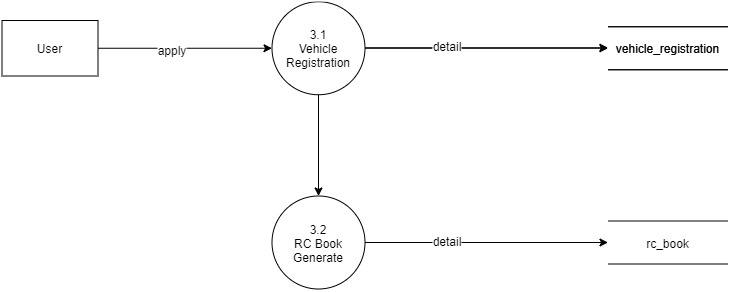
**1.1st Level (Admin)**

****

**2nd Level(Admin)**

****

**2nd Level(User)**

****

**3rd Level(User)**

## **7.4 Data Dictionary**

1. **Admin**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Admin\_Id** | Int | Primary Key | It Indicates the Id of the Admin table. |
| **Admin\_Name** | Varchar(50) | Not Null | It Contains UserName for the Admin table. |
| **Admin\_Email** | Varchar(50) | Not Null | It Contains Email Id for the Admin table. |
| **Admin\_Contact** | Varchar(13) | Not Null | It Contains Contact for the Admin table. |
| **Admin\_Password** | Varchar(50) | Not Null | It Contains Password for the Admin table. |
| **Admin\_Gender** | Varchar(50) | Not Null | It Contains Gender for the Admin table. |
| **Admin\_Hobby** | Varchar(50) | Not Null | It Contains Hobby for the Admin table. |
| **Admin\_DOB** | Date | Not Null | It Contains DOB for the Admin table. |
| **Admin\_Address** | Varchar(50) | Not Null | It Contains Address for the Admin table. |
| **Admin\_Profile** | Varchar(50) | Not Null | It Contains Profile for the Admin table. |
| **Admin\_Role** | Varchar(50) | Not Null | It Contains Role for the Admin table. |
| **Status** | tinyint | Not Null | It Contains Status for the Admin table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the Admin table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the Admin table. |

**2) State**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **State\_Id** | Int | Primary Key | It Indicates the Id of the State table. |
| **State\_Name** | Varchar(50) | Not Null | It Contains StateName for the State table. |

**3) City**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **City\_Id** | Int | Primary Key | It Indicates the Id of the City table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Name** | Varchar(50) | Not Null | It Contains CityName for the City table. |

**4) Area**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Area\_Id** | Int | Primary Key | It Indicates the Id of the Area table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **Area\_Name** | Varchar(50) | Not Null | It Contains Name for the Area table. |

**5) RTO\_Office**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Office\_Id** | Int | Primary Key | It Indicates the Id of the Office table. |
| **Area\_Id** | Int | Foreign Key | It Contains the AreaId of the Area table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **Office\_Address** | Varchar(50) | Not Null | It Contains Address for the RTO\_Office table. |

**6) Category**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Category\_Id** | Int | Primary Key | It Indicates the Id of the Category table. |
| **Category\_Name** | Varchar(50) | Not Null | It Contains Name for the Category table. |

**7) Subcategory**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Subcategory\_Id** | Int | Primary Key | It Indicates the Id of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **Subcategory\_Name** | Varchar(50) | Not Null | It Contains SubcategoryName for the Subcategory table. |

**8) Tutorial**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Tutorial\_Id** | Int | Primary Key | It Indicates the Id of the Tutorial table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **Tutorial\_Type** | Varchar(50) | Not Null | It Contains TutorialType for the Tutorial table. |
| **Tutorial\_Title** | Varchar(50) | Not Null | It Contains TutorialTitle for the Tutorial table. |
| **Tutorial\_Description** | Varchar(50) | Not Null | It Contains TutorialDescription for the Tutorial table. |
| **Tutorial\_Image** | Varchar(50) | Not Null | It Contains TutorialImage for the Tutorial table. |

**9) Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Rules\_Id** | Int | Primary Key | It Indicates the Id of the Rules table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **Rules\_Title** | Varchar(50) | Not Null | It Contains RulesTitle for the Rules table. |
| **Rules\_Description** | Varchar(50) | Not Null | It Contains RulesDescription for the Rules table. |

**10) Exam**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Exam\_Id** | Int | Primary Key | It Indicates the Id of the Exam table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Area\_Id** | Int | Foreign Key | It Contains the AreaId of the Area table. |
| **Exam\_Name** | Varchar(50) | Not Null | It Contains ExamName for the Exam table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Slot\_Id** | Int | Foreign Key | It Contains the SlotId of the Slot table. |
| **Exam\_Mark** | Int | Not Null | It Contains ExamMark for the Exam table. |
| **Exam\_Passing\_Mark** | Int | Not Null | It Contains ExamPassingMark for the Exam table. |
| **Status** | tinyint | Not Null | It Contains Status for the Exam table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the Exam table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the Exam table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |

**11) Exam\_Question**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Exam\_Question\_Id** | Int | Primary Key | It Indicates the Id of the Exam\_Question table. |
| **Exam\_Id** | Int | Foreign Key | It Contains the ExamId of the Exam table. |
| **Question\_Id** | Int | Foreign Key | It Contains the QuestionId of the Exam\_Question table. |
| **Question** | Varchar(50) | Not Null | It Contains the Question of the Exam\_Question table. |
| **Ans1** | Varchar(50) | Not Null | It Contains the Ans1 of the Exam\_Question table. |
| **Ans2** | Varchar(50) | Not Null | It Contains the Ans2 of the Exam\_Question table. |
| **Ans3** | Varchar(50) | Not Null | It Contains the Ans3 of the Exam\_Question table. |
| **Ans4** | Varchar(50) | Not Null | It Contains the Ans4 of the Exam\_Question table. |
| **Final\_Ans** | Varchar(50) | Not Null | It Contains FinalAns for the Exam\_Question table. |
| **Status** | tinyint | Not Null | It Contains Status for the Exam\_Question table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the Exam\_Question table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the Exam\_Question table. |

**12) Exam\_Type**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Exam\_Type\_Id** | Int | Primary Key | It Indicates the Id of the Exam\_Type table. |
| **Exam\_Id** | Int | Foreign Key | It Contains the ExamId of the Exam table. |
| **Exam\_Question\_Id** | Int | Foreign Key | It Contains the ExamQuestionId of the Exam\_Question table. |
| **Exam\_Question** | Varchar(50) | Not Null | It Contains the ExamQuestion of the Exam\_Type table. |
| **Exam\_Ans1** | Varchar(50) | Not Null | It Contains the ExamAns1 of the Exam\_Type table. |
| **Exam\_Ans2** | Varchar(50) | Not Null | It Contains the ExamAns2 of the Exam\_Type table. |
| **Exam\_Ans3** | Varchar(50) | Not Null | It Contains the ExamAns3 of the Exam\_Type table. |
| **Exam\_Ans4** | Varchar(50) | Not Null | It Contains the ExamAns4 of the Exam\_Type table. |
| **Exam\_Final\_Ans** | Varchar(50) | Not Null | It Contains ExamFinalAns for the Exam\_Type table. |
| **Status** | tinyint | Not Null | It Contains Status for the Exam\_Type table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the Exam\_Type table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the Exam\_Type table. |

**13) Vehical**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Vehical\_Id** | Int | Primary Key | It Indicates the Id of the Vehical table. |
| **Vehical\_Name** | Varchar(50) | Not Null | It Contains VehicalName for the Vehical table. |

**14) Proof**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Proof\_Id** | Int | Primary Key | It Indicates the Id of the Proof table. |
| **Proof\_Name** | Varchar(50) | Not Null | It Contains ProofName for the Proof table. |

**15) Licence\_Register\_Image**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Licence\_Register\_Image\_Id** | Int | Primary Key | It Indicates the Id of the Licence\_Register\_Image table. |
| **Licence\_Register\_Image\_Name** | Varchar(50) | Not Null | It Contains LicenceRegisterImageName for the Licence\_Register\_Image table. |
| **Licence\_Register\_Id** | Int | Foreign Key | It Contains LicenceRegisterId for the Licence\_Register table. |
| **Licence\_Register\_Image** | Varchar(50) | Not Null | It Contains LicenceRegisterImage for the Licence\_Register\_Image table. |

**16) Post**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Post\_Id** | Int | Primary Key | It Indicates the Id of the Post table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **Post\_Type** | Varchar(50) | Not Null | It Contains PostType for the Post table. |
| **Post\_Title** | Varchar(50) | Not Null | It Contains PostTitle for the Post table. |
| **Post\_Description** | Varchar(50) | Not Null | It Contains PostDescription for the Post table. |

**17) Post\_Image**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Post\_Image\_Id** | Int | Primary Key | It Indicates the Id of the Post\_Image table. |
| **Post\_Id** | Int | Foreign Key | It Contains PostId for the Post table. |
| **Post\_Image\_Path** | Varchar(50) | Not Null | It Contains PostImagePath for the Post\_Image table. |

**18) Slot**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Slot\_Id** | Int | Primary Key | It Indicates the Id of the Slot table. |
| **Slot\_Time** | Time | Not Null | It Contains SlotTime for the Slot table. |

**19) Slot\_Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Slot\_Schedule\_Id** | Int | Primary Key | It Indicates the Id of the Slot\_Schedule table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Slot\_Start\_Date** | Date | Not Null | It Contains SlotStartDate for the Slot\_Schedule table. |
| **Slot\_End\_Date** | Date | Not Null | It Contains SlotEndDate for the Slot\_Schedule table. |
| **Slot1** | Varchar(50) | Not Null | It Contains the Slot1 of the Slot\_Schedule table. |
| **Slot2** | Varchar(50) | Not Null | It Contains the Slot2 of the Slot\_Schedule table. |
| **Slot3** | Varchar(50) | Not Null | It Contains the Slot3 of the Slot\_Schedule table. |
| **Slot4** | Varchar(50) | Not Null | It Contains the Slot4 of the Slot\_Schedule table. |
| **Slot5** | Varchar(50) | Not Null | It Contains the Slot5 for the Slot\_Schedule table. |

**20) Exam\_Create**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Exam\_Create\_Id** | Int | Primary Key | It Indicates the Id of the Exam\_Create table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **Exam\_Id** | Int | Foreign Key | It Contains the ExamId of the Exam table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Slot\_No** | Int | Not Null | It Contains the SlotNo of the Exam\_Create table. |
| **Exam\_Date** | Date | Not Null | It Contains the ExamDate of the Exam\_Create table. |

**21) User**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **User\_Id** | Int | Primary Key | It Indicates the Id of the User table. |
| **User\_Name** | Varchar(50) | Not Null | It Contains UserName for the User table. |
| **User\_Email** | Varchar(50) | Not Null | It Contains Email Id for the User table. |
| **User\_Contact** | Varchar(13) | Not Null | It Contains Contact for the User table. |
| **User\_Password** | Varchar(50) | Not Null | It Contains Password for the User table. |
| **User\_Gender** | | Varchar(50) | Not Null | It Contains Gender for the User table. |
| **User\_DOB** | Date | Not Null | It Contains DOB for the User table. |
| **User\_Address** | Varchar(50) | Not Null | It Contains Address for the User table. |
| **User\_Profile** | Varchar(50) | Not Null | It Contains Profile for the User table. |
| **Status** | tinyint | Not Null | It Contains Status for the User table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the User table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the User table. |

**22) Licence\_Register**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Licence\_Register\_Id** | Int | Primary Key | It Indicates the Id of the Licence\_Register table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Area\_Id** | Int | Foreign Key | It Contains the AreaId of the Area table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Name** | Varchar(50) | Not Null | It Contains Name for the Licence\_Register table. |
| **Email** | Varchar(50) | Not Null | It Contains Email Id for the Licence\_Register table. |
| **Address** | Varchar(50) | Not Null | It Contains Address for the Licence\_Register table. |
| **Contact\_No** | Bigint(20) | Not Null | It Contains ContactNo for the Licence\_Register table. |
| **Vehical\_Id** | Int | Foreign Key | It Contains the VehicalId of the Vehical table. |
| **DOB** | Date | Not Null | It Contains DOB for the Licence\_Register table. |
| **Blood Group** | Varchar(50) | Not Null | It Contains BloodGroup for the Licence\_Register table. |
| **Qualification** | Varchar(50) | Not Null | It Contains Qualification for the Licence\_Register table. |
| **Photo** | Varchar(50) | Null | It Contains Photo for the Licence\_Register table. |
| **Id\_Proof\_Type** | Varchar(50) | Not Null | It Contains IdProofType for the Licence\_Register table. |
| **Id\_Proof\_Image** | Varchar(50) | Not Null | It Contains IdProofImage for the Licence\_Register table. |
| **Register\_Date** | Date | Not Null | It Contains RegisterDate for the Licence\_Register table. |
| **Created\_at** | timestamp | Not Null | It Contains Created At for the Licence\_Register table. |
| **Updated\_at** | timestamp | Not Null | It Contains Update At for the Licence\_Register table. |
| **Nominee\_Name** | Varchar(50) | Not Null | It Contains NomineeName for the Licence\_Register table. |
| **Adhar\_No** | Int | Null | It Contains AdharNo for the Licence\_Register table. |
| **User\_Id** | Int | Foreign Key | It Contains UserId for the Licence\_Register table. |

**23) Licence\_Conforme**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Conform\_Id** | Int | Primary Key | It Indicates the Id of the Licence\_Conform table. |
| **User\_Id** | Int | Foreign Key | It Contains UserId for the Licence\_Conform table. |
| **Vehical\_Id** | Int | Foreign Key | It Contains VehicalId for the Vehical table. |
| **Licence\_Type** | Varchar(50) | Not Null | It Contains LicenceType for the Licence\_Conform table. |
| **Issue\_Date** | Date | Not Null | It Contains IssueDate for the Licence\_Conform table. |
| **Validity\_Date** | Date | Not Null | It Contains ValidityDate for the Licence\_Conform table. |
| **Licence\_Register\_Id** | Int | Foreign Key | It Contains LicenceRegisterId for the Licence\_Conform table. |
| **Licence\_Num** | Int | Not Null | It Contains LicenceNum for the Licence\_Conform table. |

**24) Vehical\_Registration**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Vehical\_Registration**  **\_Id** | Int | Primary Key | It Indicates the Id of the Vehical\_Registration table. |
| **Vehical\_Id** | Int | Foreign Key | It Contains VehicalId for the Vehical table. |
| **Vehical\_No** | Int | Not Null | It Contains VehicalNo for the Vehical\_Registration table. |
| **Perminate\_Add** | Varchar(50) | Not Null | It Contains PerminateAdd for the Vehical\_Registration table. |
| **Temp\_Add** | Varchar(50) | Not Null | It Contains TempAdd for the Vehical\_Registration table. |
| **Body** | Varchar(50) | Not Null | It Contains Body for the Vehical\_Registration table. |
| **Year\_Of\_Manufacture** | Date | Not Null | It Contains YearOfManufacture for the Vehical\_Registration table. |
| **No\_Of\_Cilyinder** | Varchar(50) | Not Null | It Contains NoOfCilyindeer for the Vehical\_Registration table. |
| **Chaiss\_No** | Varchar(50) | Not Null | It Contains ChaissNo for the Vehical\_Registration table. |
| **Engine\_No** | Varchar(50) | Not Null | It Contains EngineNo for the Vehical\_Registration table. |
| **Registration\_Date** | Date | Not Null | It Contains RegistrationDate for the Vehical\_Registration table. |
| **Licence\_Register\_Id** | Int | Foreign Key | It Contains the LicenceRegisterId of the Licence\_Register table. |
| **User\_Id** | Int | Foreign Key | It Contains UserId for the User table. |

**25) Inquiry**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Inquiry\_Id** | Int | Primary Key | It Indicates the Id of the Inquiry table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Area\_Id** | Int | Foreign Key | It Contains the AreaId of the Area table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Inquiry\_Title** | Varchar(50) | Not Null | It Contains the InquiryTitle of the Inquiry table. |
| **Inquiry\_Question** | Varchar(50) | Not Null | It Contains the InquiryQuestion of the Inquiry table. |

**26) Slot\_Book**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Slot\_Book\_Id** | Int | Primary Key | It Indicates the Id of the Slot\_Book table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **Slot\_Id** | Int | Foreign Key | It Contains the SlotId of the Slot table. |
| **State\_Id** | Int | Foreign Key | It Contains the StateId of the State table. |
| **City\_Id** | Int | Foreign Key | It Contains the CityId of the City table. |
| **Office\_Id** | Int | Foreign Key | It Contains the OfficeId of the Office table. |
| **Licence\_Register\_Id** | Int | Foreign Key | It Contains the LicenceRegisterId of the Licence\_Register table. |
| **Subcategory\_Id** | Int | Foreign Key | It Contains the SubcategoryId of the Subcategory table. |
| **Category\_Id** | Int | Foreign Key | It Contains the CategoryId of the Category table. |
| **Vehical\_Type\_Id** | Int | Foreign Key | It Contains the VehicalTypeId of the Vehical\_Registration table. |
| **Slot\_No** | Int | Not Null | It Contains the SlotNo of the Slot\_Book table. |
| **Slot\_Book\_Date** | Date | Not Null | It Contains the SlotBookDate of the Slot\_Book table. |

**27) Inquiry\_Ans**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Inquiry\_Ans\_Id** | Int | Primary Key | It Indicates the Id of the Inquiry\_Ans table. |
| **Inquiry\_Id** | Int | Foreign Key | It Contains the InquiryId of the Inquiry table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **Inquiry\_Ans** | Varchar(50) | Not Null | It Contains the InquiryAns of the Inquiry\_Ans table. |

**28) Result**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Result\_Id** | Int | Primary Key | It Indicates the Id of the Result table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **Question\_Id** | Int | Foreign Key | It Contains the QuestionId of the Exam\_Type table. |
| **Ans** | Varchar(50) | Not Null | It Contains the Ans of the Result table. |
| **Date** | Date | Not Null | It Contains the Date of the Result table. |
| **Exam\_Id** | Int | Foreign Key | It Contains the ExamId of the Exam table. |

**29) F\_Result**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **F\_Result\_Id** | Int | Primary Key | It Indicates the Id of the F\_Result table. |
| **Result\_Id** | Int | Foreign Key | It Contains the ResultId of the Result table. |
| **Total\_Ans** | Varchar(50) | Not Null | It Contains the TotalAns of the F\_Result table. |
| **Result\_Type** | Varchar(50) | Not Null | It Contains the ResultType of the F\_Result table. |
| **User\_Exam\_Id** | Int | Foreign Key | It Contains the UserExamId of the Exam\_Create table. |
| **User\_Id** | Int | Foreign Key | It Contains the UserId of the User table. |
| **Admin\_Id** | Int | Foreign Key | It Contains the AdminId of the Admin table. |
| **Exam\_Id** | Int | Foreign Key | It Contains the ExamId of the Exam table. |
| **Result\_Description** | Varchar(50) | Not Null | It Contains the ResultDescription of the F\_Result table. |
| **Total\_Mark** | Varchar(50) | Not Null | It Contains the Total\_Mark of the F\_Result table. |

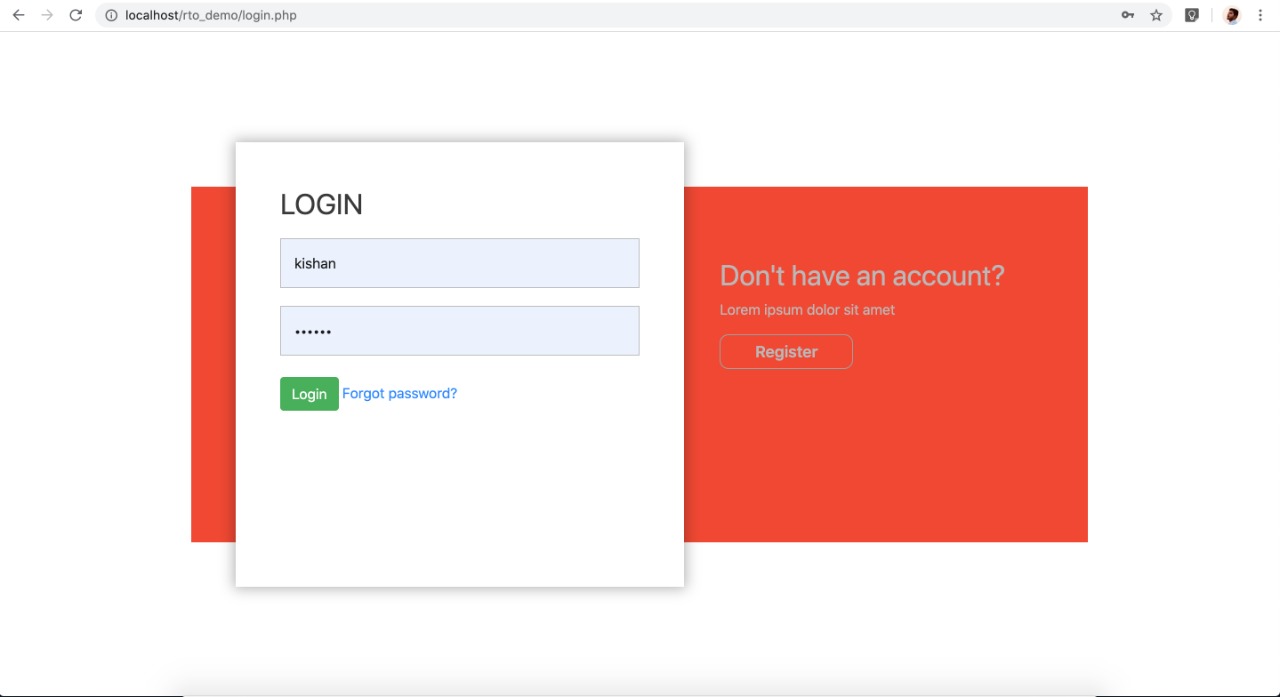
**30) Proof\_Image**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **Proof\_Image\_Id** | Int | Primary Key | It Indicates the Id of the Proof\_Image table. |
| **Licence\_Register\_Id** | Int | Foreign Key | It Contains the Licence\_Register\_Id of the Licence\_Register table. |
| **Proof\_Image\_Path** | Varchar(50) | Not Null | It Contains the Proof\_Image\_Path of the Proof\_Image table. |

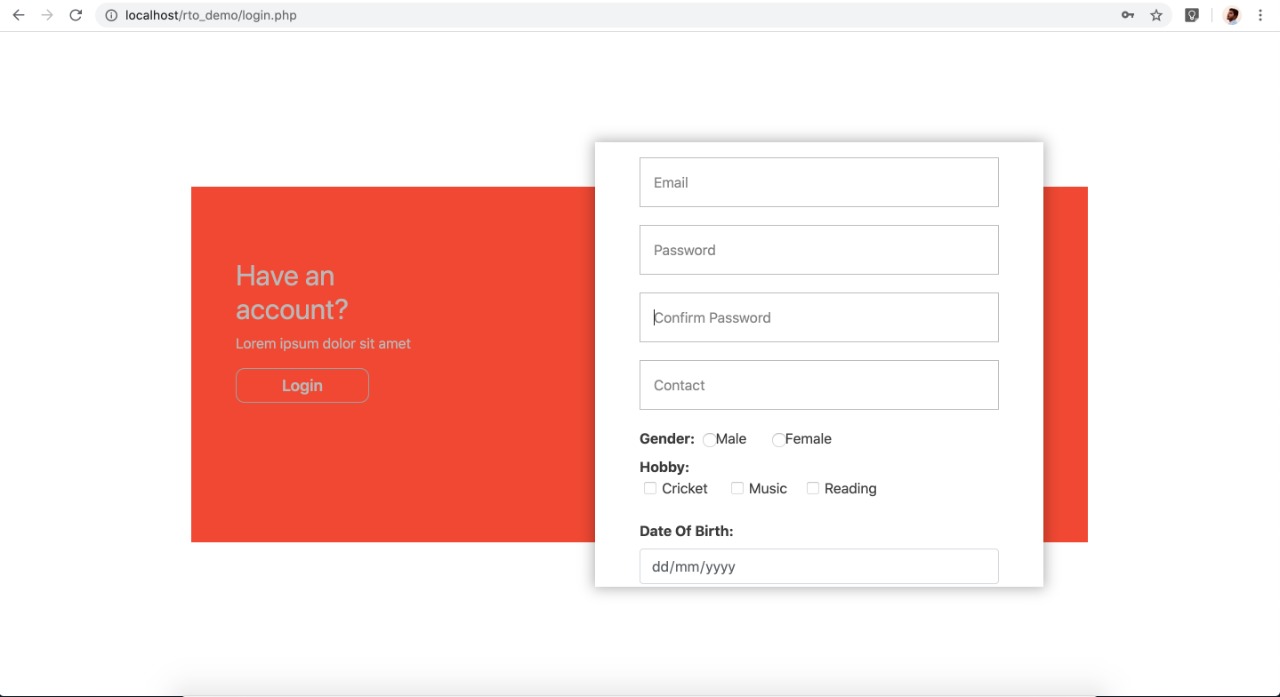
# 8. Designing

## 8.1 User Interface

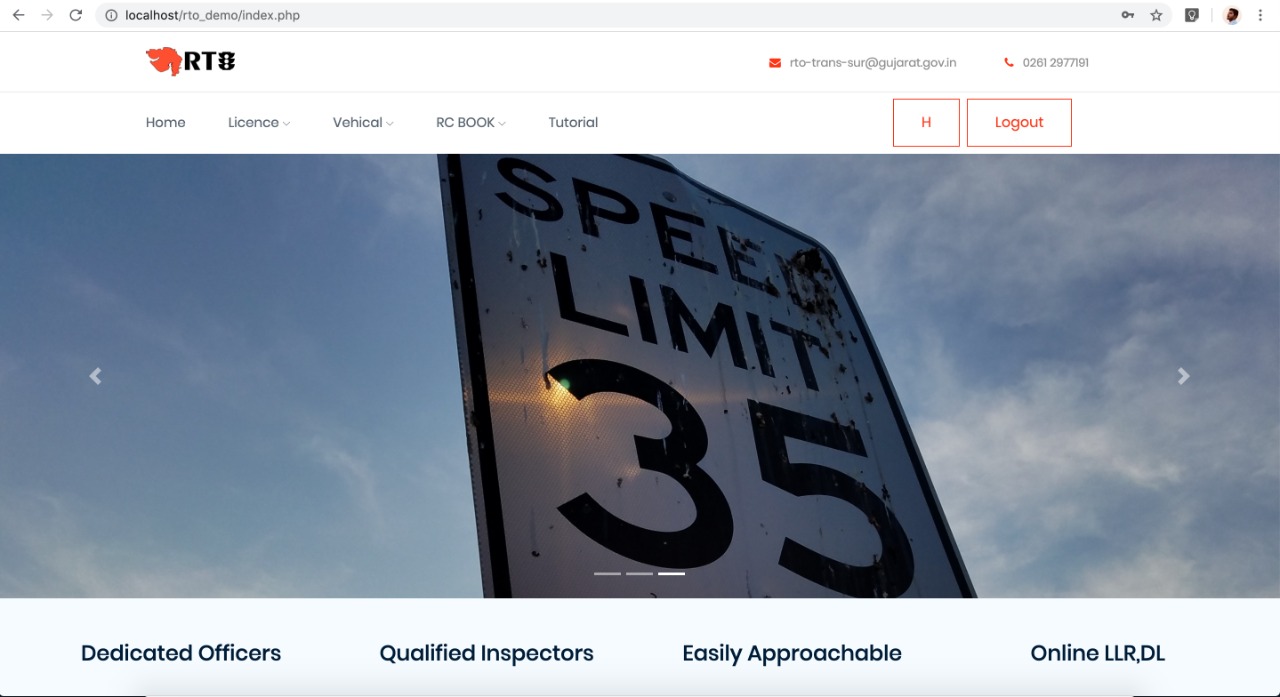
Login(User):



Registration(User):

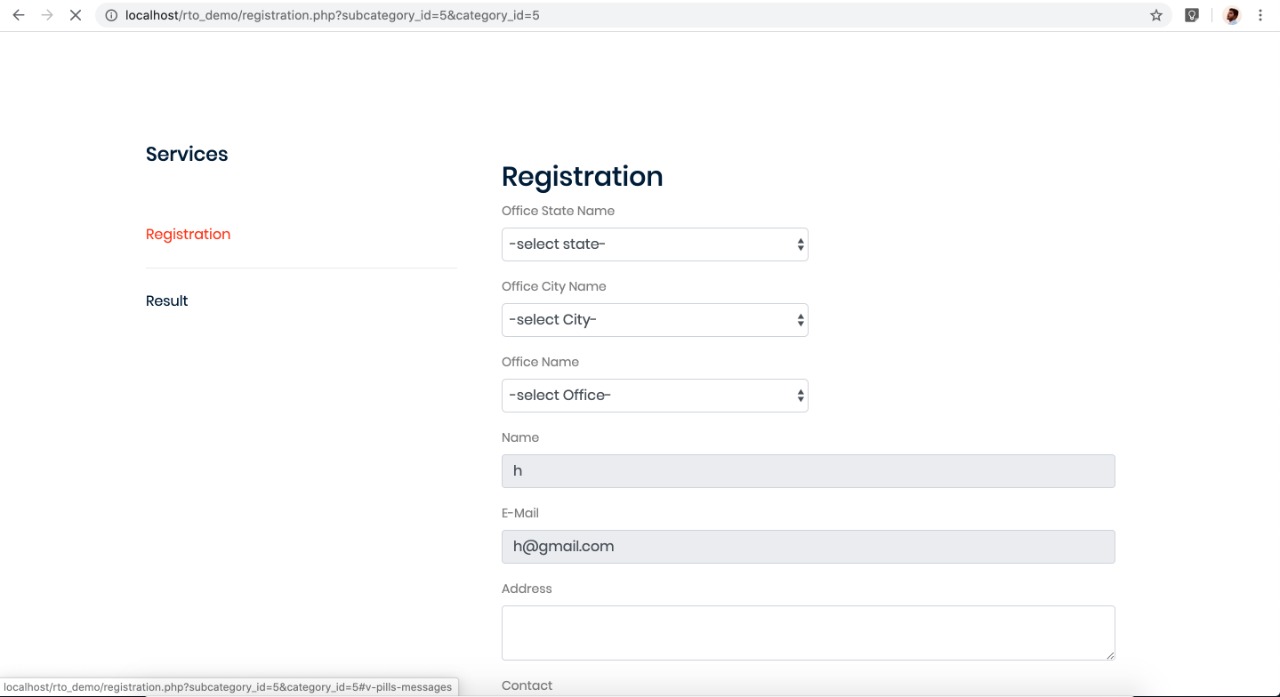


Homepage:

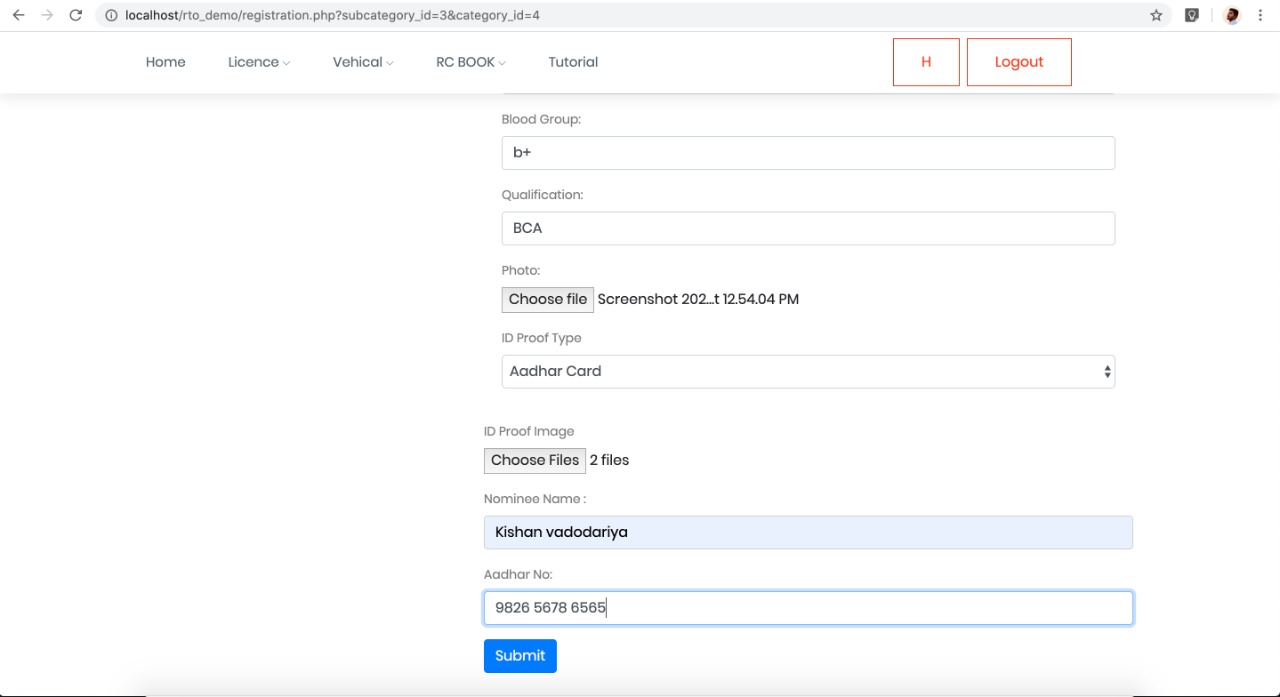


License /Learner License Registration:

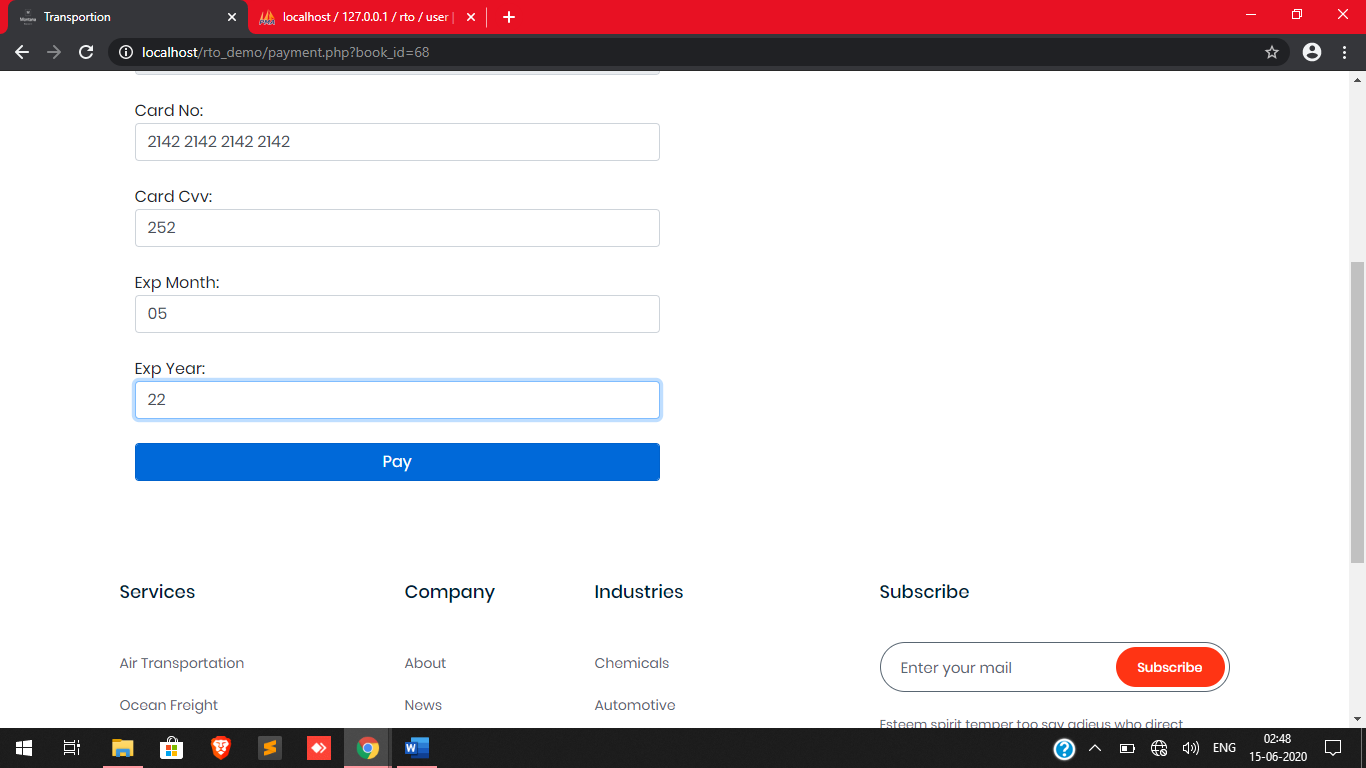
(i)



(ii)

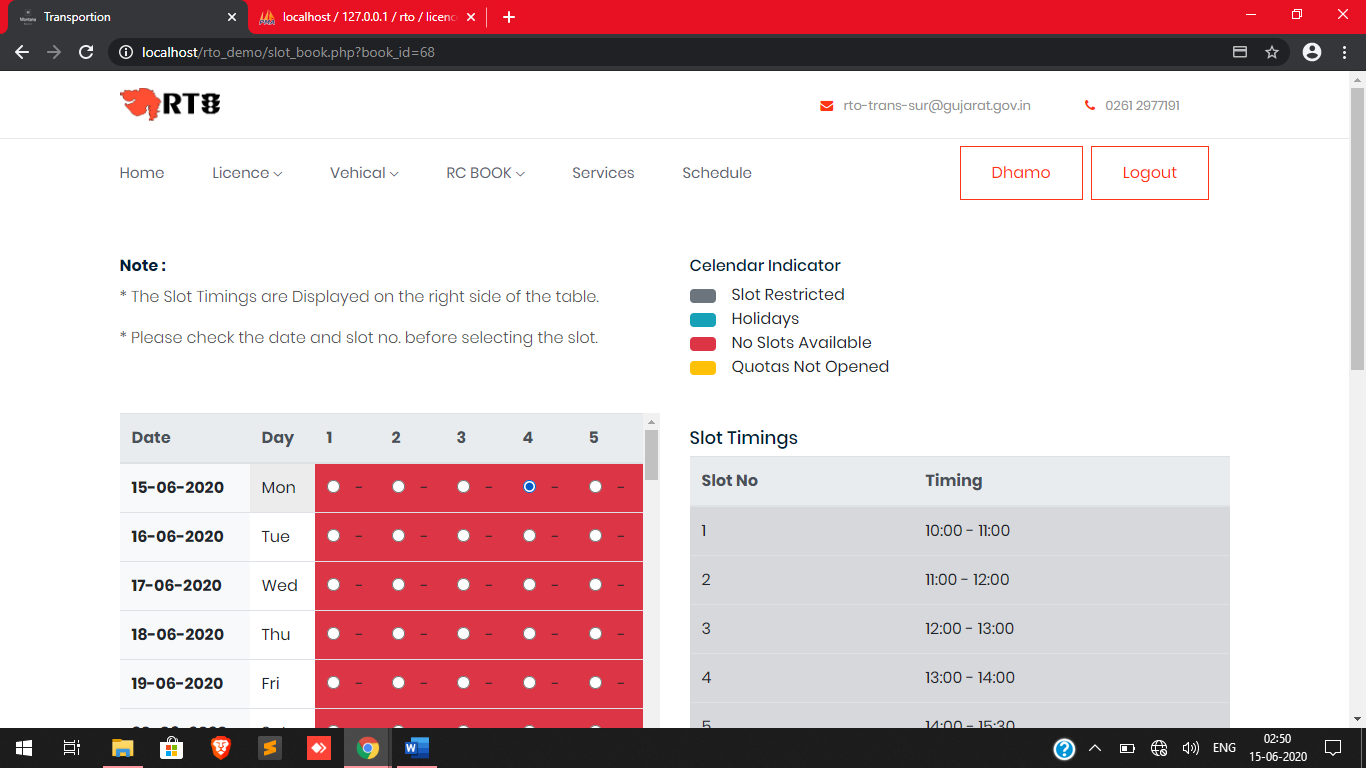


Payment:

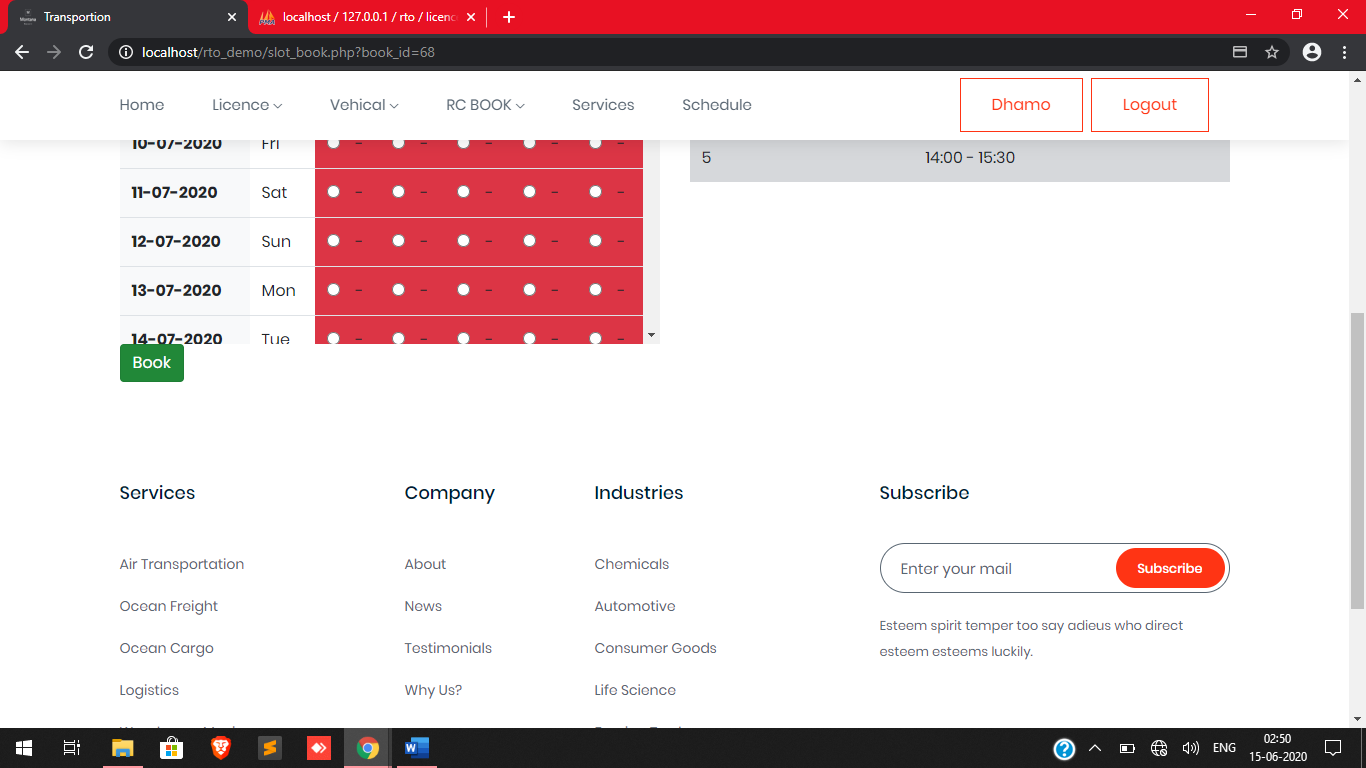


Slor Booking:

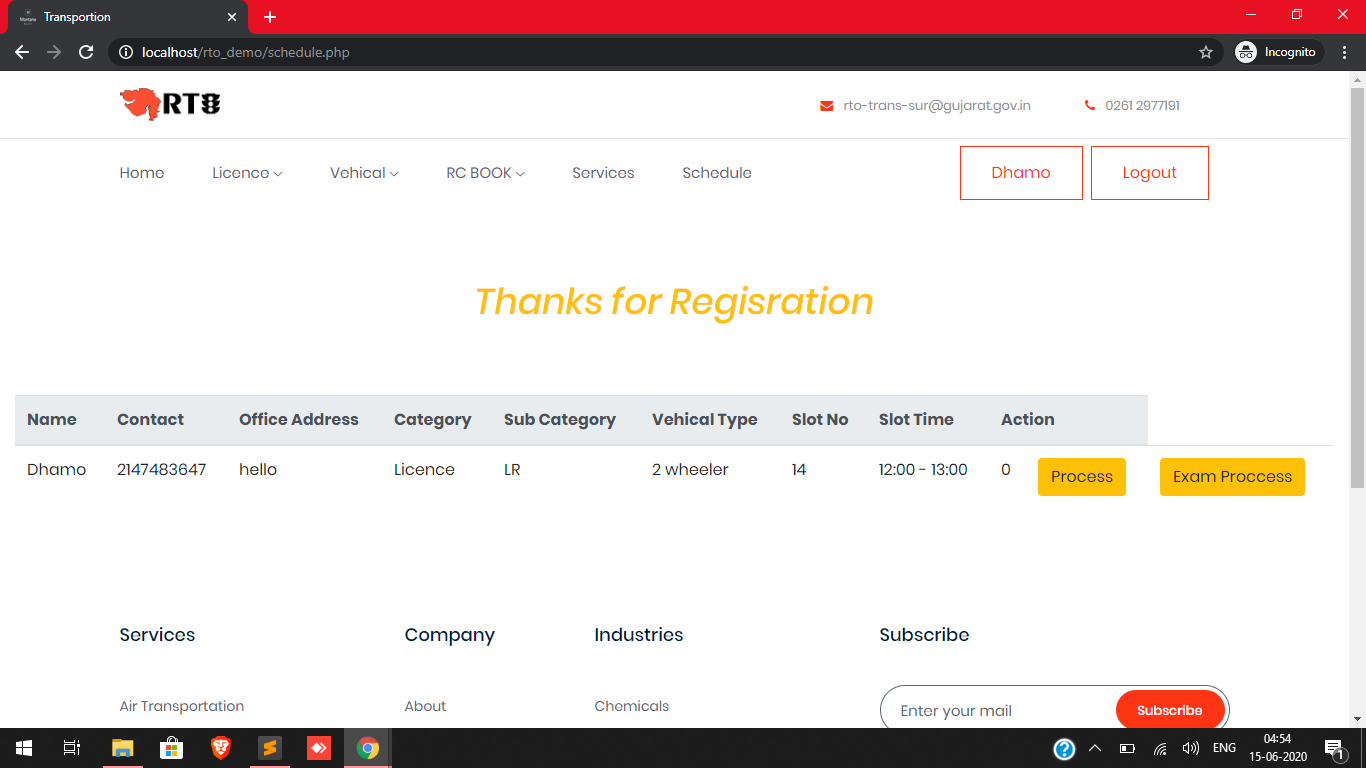
(i)



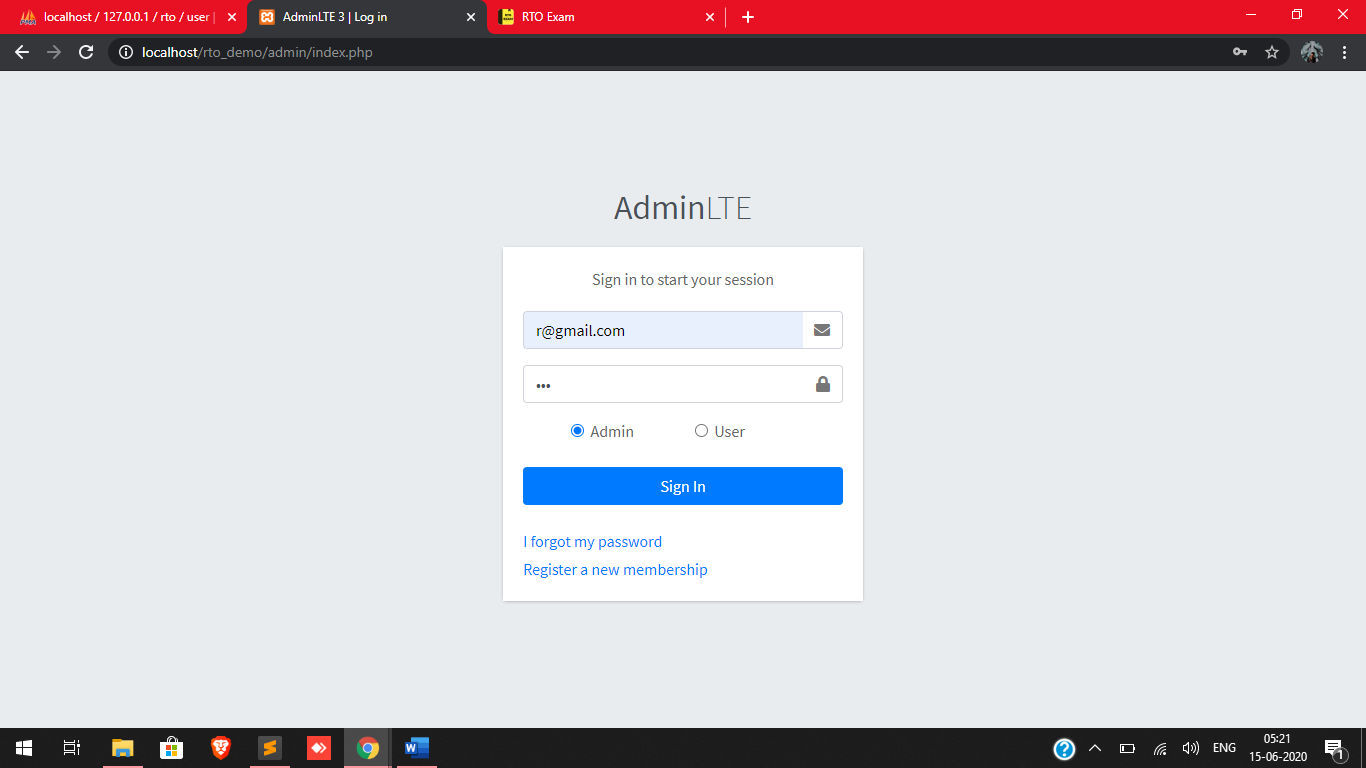
(ii)



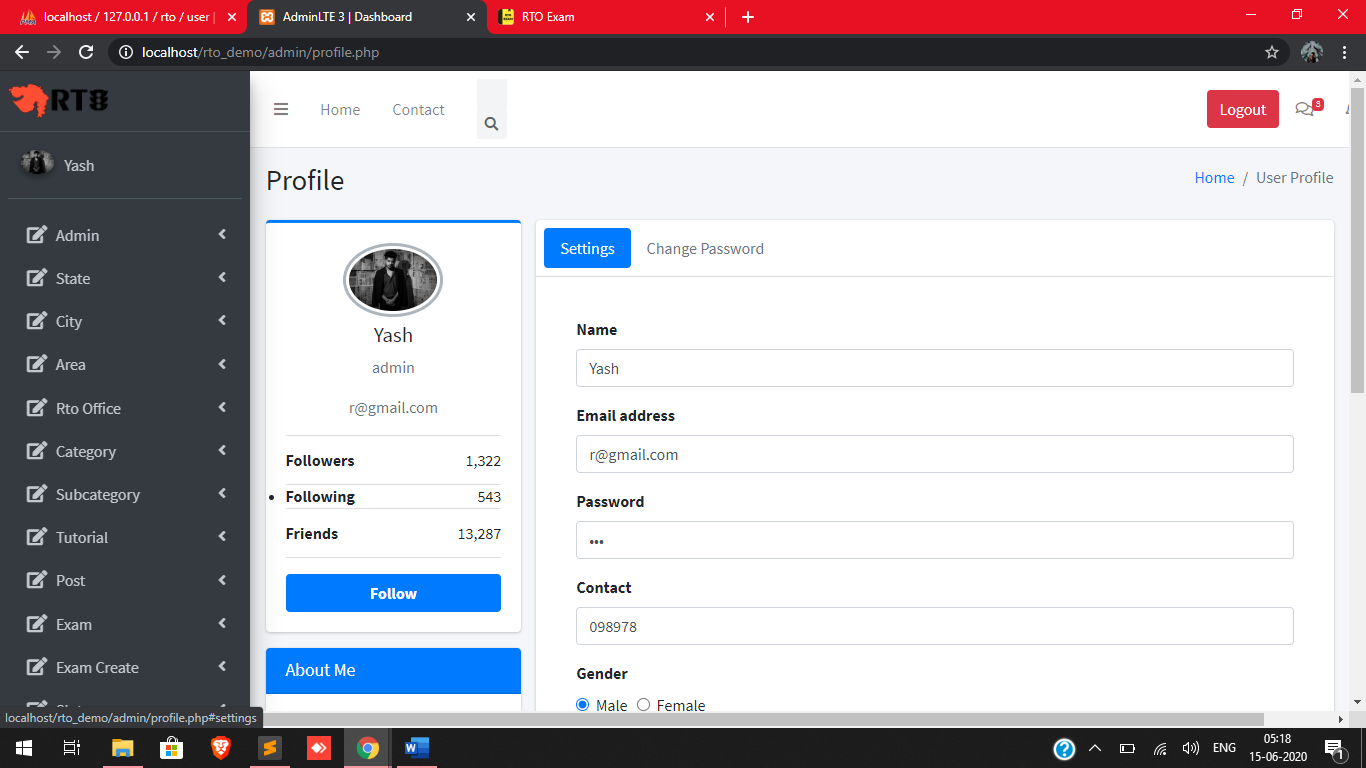
Confirmation of Registration:



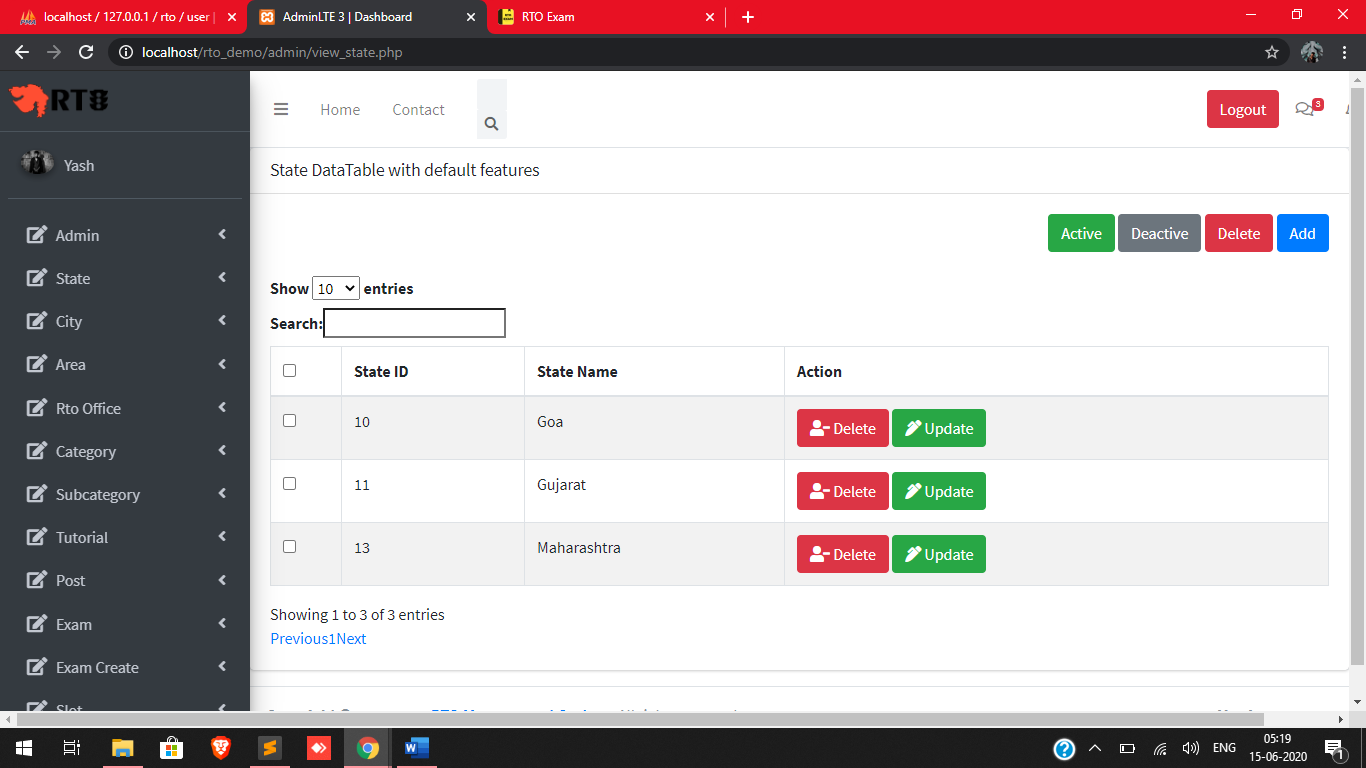
Admin Side (Login):



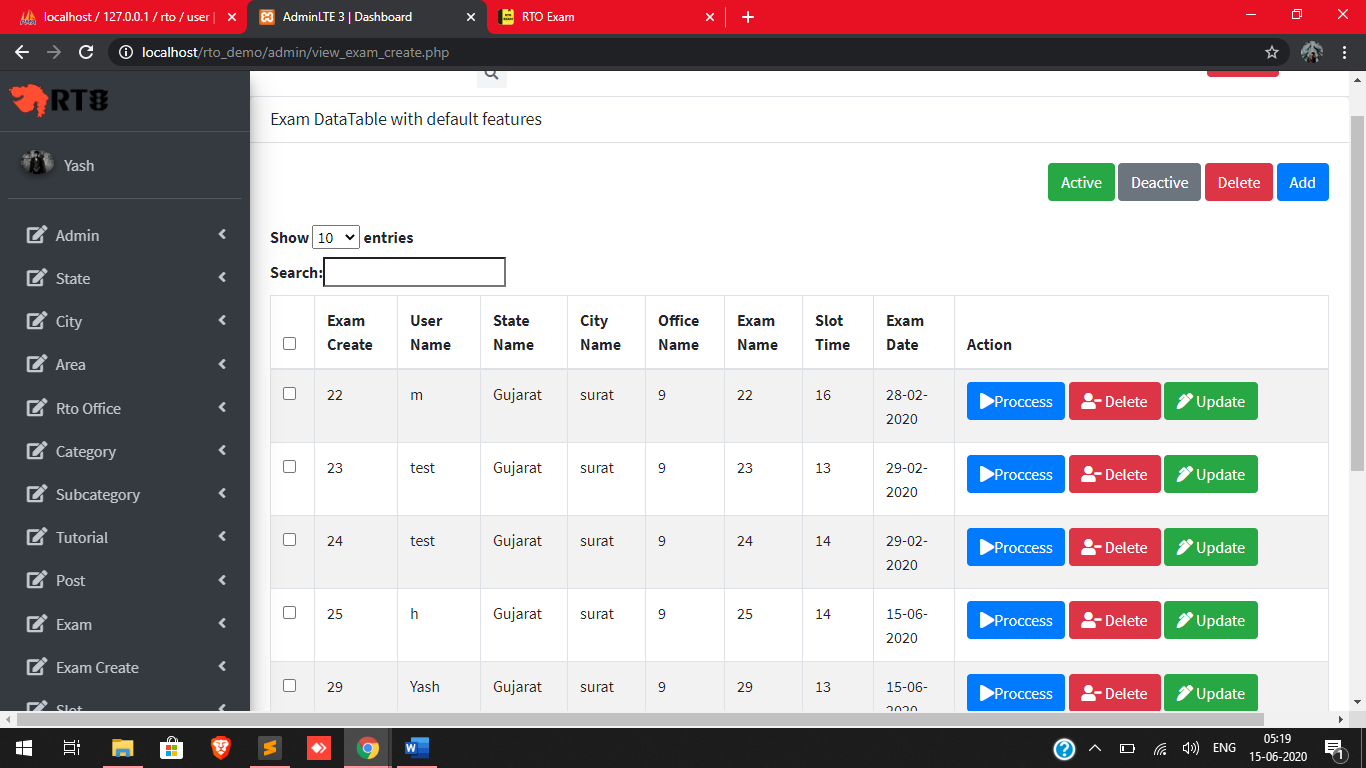
Admin Side (Add Details):



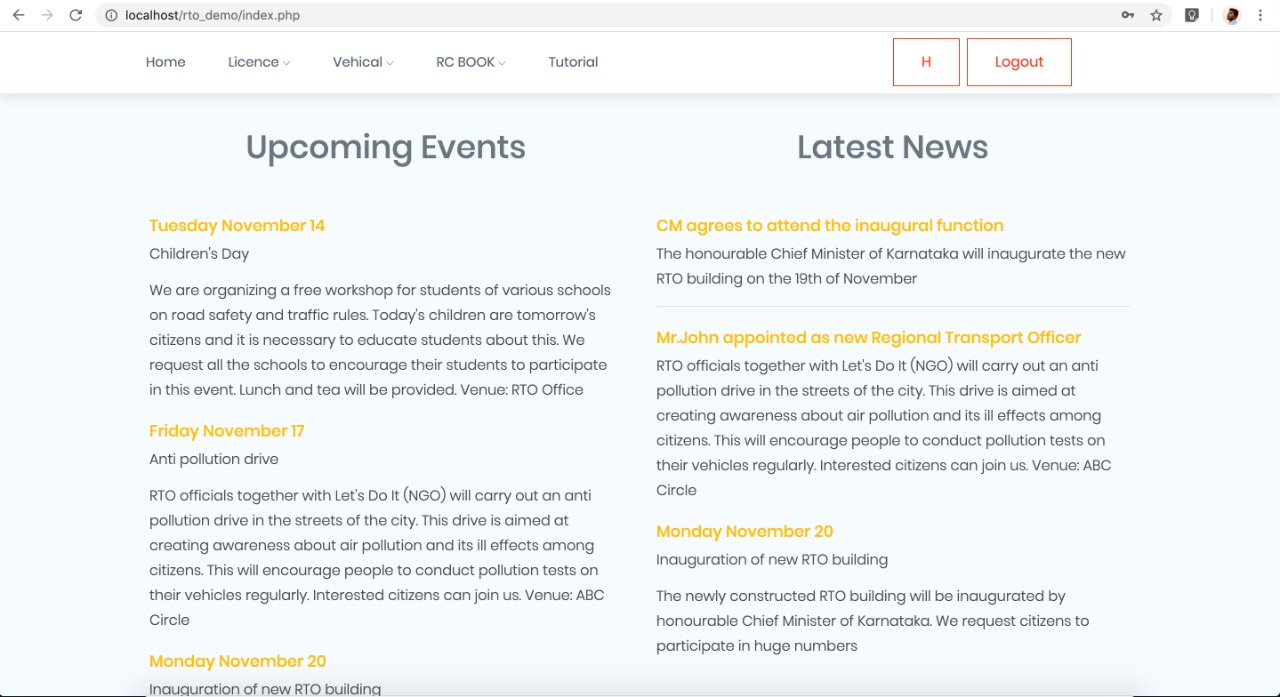
Admin Side (View Details):



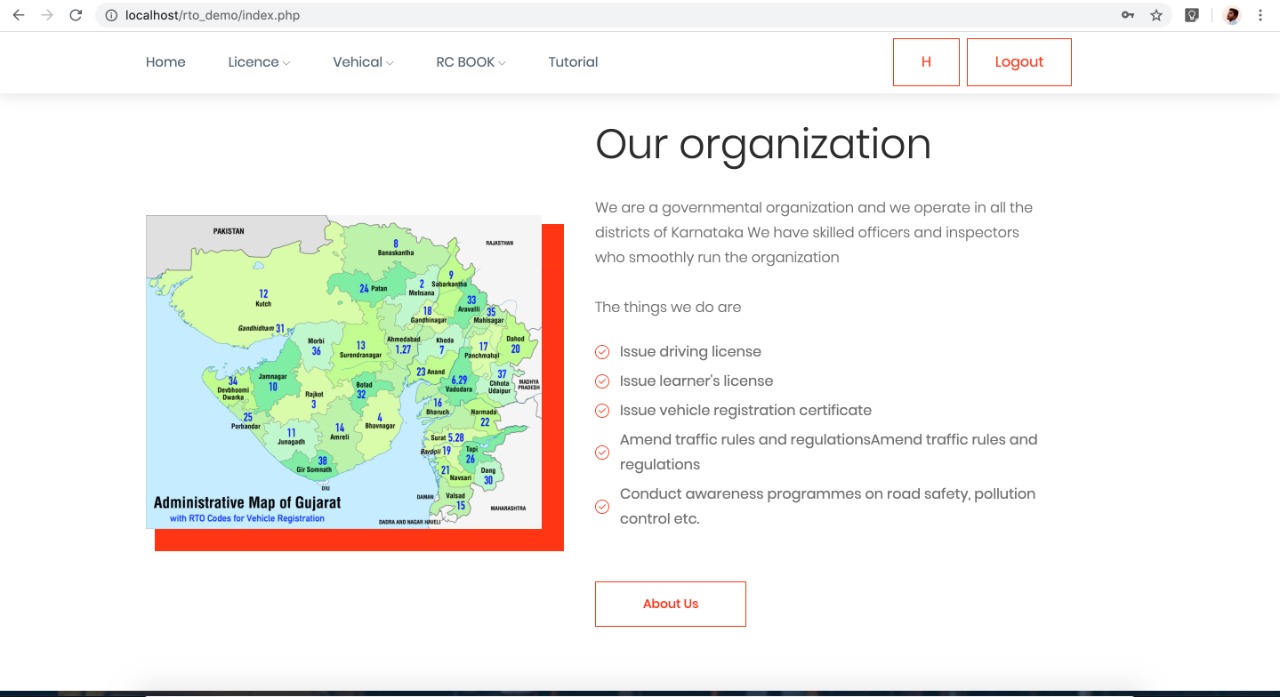
Exam Create:



Events & News:



About Us:



# 9. Testing

## 9.1 Software Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing is an exposure of a system to trial input to see whether software meets correct output. Testing cannot be determined whether software meets user’s needs, only whether it appears to confirm to requirements. Testing can show that a system is free of errors, only that it contains error. Testing finds errors, it does not correct errors. Software success is a quality product, on time and within cost. Through testing can reveal critical mistakes. Testing should therefore,

* Validate Performance
* Detects Errors
* Identify Inconsistencies

**Test Objective :**

* There is strong evidence that effective requirement management leads to overall project cost savings. The three primary reasons for this are,
* Requirement errors typically cost well over 10 times more to repair than other errors.
* Requirement errors typically comprise over 40% of all errors in a software project.
* Small reduction in the number of requirement errors pays big dividend in avoided rework costs and schedule delays.
* System are not designed as entire systems nor are they tested as single systems the analyst must perform both unit and system testing.

## 9.2 Unit Testing

* + In unit testing Module is tested separately and the programmer simultaneously along with the coding of the module performs it.
  + In unit testing the analyst tests the programs making up a system. For this reason, unit testing is sometime called program testing. Unit testing gives stress on modules independently of one another, to find errors. This helps the tester in detecting errors in coding and logic that are contained within that module alone. The errors resulting from the interaction between modules are initially avoided.
  + Unit testing can be performed from the bottom up, Starting with smallest and lowest-level modules and proceeding one at a time., for each module in Bottom-up testing a short program is used to execute the module and provides the needed data, so that the module is asked to perform the way it will when embedded within the larger system.

## 9.3 System Testing

In order to ensure the system is functioning as what is expected and to meet the organization’s requirement, it is essential to organize well-planned testing strategy. Thus we come to this proposed testing strategy, which cover brief information on the testing objectives, testing method, and testing schedule.

**Objectives of Testing:**

To identify and correct the errors in the proposed system. There are four types of testing.

Methods:

1. Unit Testing
2. Integration Testing
3. Validation Testing
4. System Testing

**1. Unit Testing**

The most fundamental part of unit testing is constructing individual test cases. A test case never a single questing about the code it is testing. A test case should able to run completely by itself , without any human input. It is about computerization. It determines by itself, whether the function it is testing has passed or failed without human interpreting the result.

**The steps are following:**

Manually code is tested like spelling check, logic and error etc.

Once the manually check is done the complication has been done. Syntactical errors if any have to be corrected.

After the clean completion of the programed, some dummy data as specification has been used for testing of that module to see if it works as specified.

**2. Integration Testing**

**Objectives:**

The main objectives of this testing is to make unit tested component and build a program structure that has been ordered by design.

Integration testing can proceed in a number of different ways, which can be broadly characterized as top-down or bottom-up integration testing.

**3. Validation Testing**

**Objectives:**

The objectives of the validation testing is to ensure the software is functioning as close as possible in a manner expect by the client.

At the end of the integration testing, validation-testing may begin when the software is completely assembled as a package and all errors have been corrected. In validation testing designer has to fulfil all the requirement of the company.

In our system for validation testing, we checked the organization information. We have checked information related to the organization.

**4. System Testing**

**Objectives:**

The objectives of the system testing are to promote software functionality, reliability and maintainability.

# 10. Reference

<https://www.google.co.in/>

<https://www.wikipedia.org/>

<https://parivahan.gov.in/parivahan/>

<https://sarathi.parivahan.gov.in/sarathiservice/>