**FULL STACK DEVELOPMENT**

(RFID PARKING SYSTEM)

*Summer Internship Report Submitted in partial fulfillment*

*of the requirement for under graduate degree of*

**Bachelor of Technology**

In

**Computer Science and Engineering**

By

**V.AKHILA**

**221710316028**

**https://github.com/akhila999/Full-Stack**

*Under the Guidance of*

Mr.Anil Kumar

Assistant Professor



Department Of Computer Science and Engineering

GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

                                                        July 2020

**DECLARATION**

I submit this industrial training work entitled **“RFID Parking System**” to GITAM (Deemed to Be University), Hyderabad in partial fulfillment of the requirements for the award of the degree of “**Bachelor of Technology**” in “**CSE**”. I declare that it was carried out independently by me under the guidance of (**Mr.Anil kumar)**, Asst. Professor, GITAM (Deemed to Be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place: HYDERABAD                 V.AKHILA

Date:    27-07-2020                                                                                                    221710316028

 GITAM (DEEMED TO BE UNIVERSITY)

     Hyderabad-502329, India

                                                                                Dated: 27-07-2020

**CERTIFICATE**

 This is to certify that the Industrial Training Report entitled **“RFID Parking System”** is being submitted by V.AKHILA(221710316028) in partial fulfillment of the requirement for the award of **Bachelor of Technology** **in CSE** at GITAM (Deemed To Be University), Hyderabad during the academic year 2019-20

               It is faithful record work carried out by her at the **CSE**, GITAM University Hyderabad Campus under my guidance and supervision.

**Mr. M. Venkateswarlu                                        Dr. K. Manjunath Chari**

Assistant Professor                                                                 Professor and HOD

Department of ECE                                                               Department of ECE

**ACKNOWLEDGEMENT**

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank respected **Dr. N. Siva Prasad,** Pro Vice Chancellor, GITAM Hyderabad and **Dr. CH. Sanjay,** Principal, GITAM Hyderabad

I would like to thank respected **Dr. K. Manjunath Chari,** Head of the Department of Electronics and Communication Engineering for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present a internship report. It helped me a lot to realize of what we study for.

I would like to thank the respected faculties **Mr. M. Venkateswarlu** who helped me to make this internship a successful accomplishment.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

                                                                                                     V.AKHILA

(221710316028)

**ABSTRACT**

Radio Frequency Identification (RFID) technology is widely used in various applications such as attendance system, tracking system, monitoring system or parking system. Currently, the existing parking system used manual entrance through security guard to access the premise. Therefore, the company need to hire security guard to monitor the premise. In addition, the security guards need to monitor all movement of vehicle or person that enter or leave the premise. As a result, unauthorized vehicle or person can easily access the building. To address this problem, we proposed a parking system using RFID technology that can monitor vehicle’s movement that enter or leave the specific area or place by scanning the RFID tag. The potential benefit is it can improve security for both security guards and users. Besides that, this parking system can facilitate access control for users and improve traffic flow during peaks period. There are five modules in the proposed parking systems which are user registration, vehicle registration, RFID tag, staff and report generation.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE NO.** |
| **1.** | **Introduction** | **10-12** |
|  | 1.1 Importance of Full Stack Development | 10 |
|  | 1.2 Front End | 11-12 |
|  | 1.3 Back End | 12 |
| **2.** | **Web Development** | **13-15** |
|  | 2.1 Web Site | 13-14 |
|  | 2.2 Web Page | 14 |
|  | 2.3 Web Browser | 14-15 |
|  | 2.4 Mobile Web Browser | 15 |
| **3.** | **The Steps to Create a Website** | **16-28** |
|  | 3.1 UI development | 17 |
|  | 3.1.1 HTML | 17-18 |
|  | 3.1.2 CSS | 19-21 |
|  | 3.1.3 Bootstrap | 21-23 |
|  | 3.2 Scripting | 23 |
|  | 3.2.1 Server-side scripting | 23-24 |
|  | 3.2.1.1 Server-side scripting languages | 24-25 |
|  | 3.2.2 Client-side scripting | 24-25 |
|  | 3.3 Database | 26-27 |
|  | 3.4 SQL | 27 |
|  | 3.5 Queries | 28 |
| **4.** | **Scripting Languages** | **29-31** |
|  | 4.1 Java Script | 29 |
|  | 4.2 jQuery | 29-30 |

|  |  |  |
| --- | --- | --- |
|  | 4.3 Ajax | 30 |
|  | 4.4 Json | 30 |
|  | 4.5 Node Js | 31 |
| **5.** | **Software Requirement Specification (SRS)** | **32** |
|  | 5.1 Hardware Requirement | 32 |
|  | 5.2 Software Requirement | 32 |
| **6.** | **Work Flow** | **33-35** |
|  | 6.1 Context Level Diagram | 33 |
|  | 6.2 Level 1-DFD | 33 |
|  | 6.3 Level 2-DFD | 34 |
|  | 6.4 Use Case Diagram | 34 |
|  | 6.5 E-R Diagram | 35 |
| **7.** | **Project** | **36-41** |
|  | 7.1 Project Aim and Objectives | 36 |
|  | 7.1.1 Background of the Project | 36-37 |
|  | 7.1.2 RFID System Fundamentals | 37 |
|  | 7.1.3 Components | 38 |
|  | 7.1.4 Component Description | 38-39 |
|  | 7.1.5 Circuit Diagram | 39 |
|  | 7.1.6 Working | 40 |
|  | 7.1.7 Advantages and Applications | 40 |
|  | 7.1.8 Disadvantages | 40 |
|  | 7.1.9 Limitations | 40 |

|  |  |  |
| --- | --- | --- |
|  | 7.2 Technologies Used | 41 |
|  | 7.3 Technical Details | 41 |
| **8.** | **Backend screenshots** | **42-51** |
|  | 8.1 Sign in | 43 |
|  | 8.2 Sign up | 45 |
|  | 8.3 Feedback | 49 |
| **9.** | **Maintenance** | **52** |
| **10.** | **Future Scope and Future Enhancement** | **53** |
| **11.** | **Conclusion** | **54** |
| **12.** | **References** | **55** |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Figure 1.1 | Basic structure of full stack development | 11 |
| Figure 1.2 | System parts | 17 |
| Figure 1.3 | Login page without CSS | 21 |
| Figure 1.4 | Login page with CSS | 21 |
| Figure 1.5 | Programming language popularity by GitHub projects | 24 |
| Figure 1.6 | Client-side and server-side scripting | 26 |

**CHAPTER 1**

**FULL STACK DEVELOPMENT**

**INTRODUCTION:**

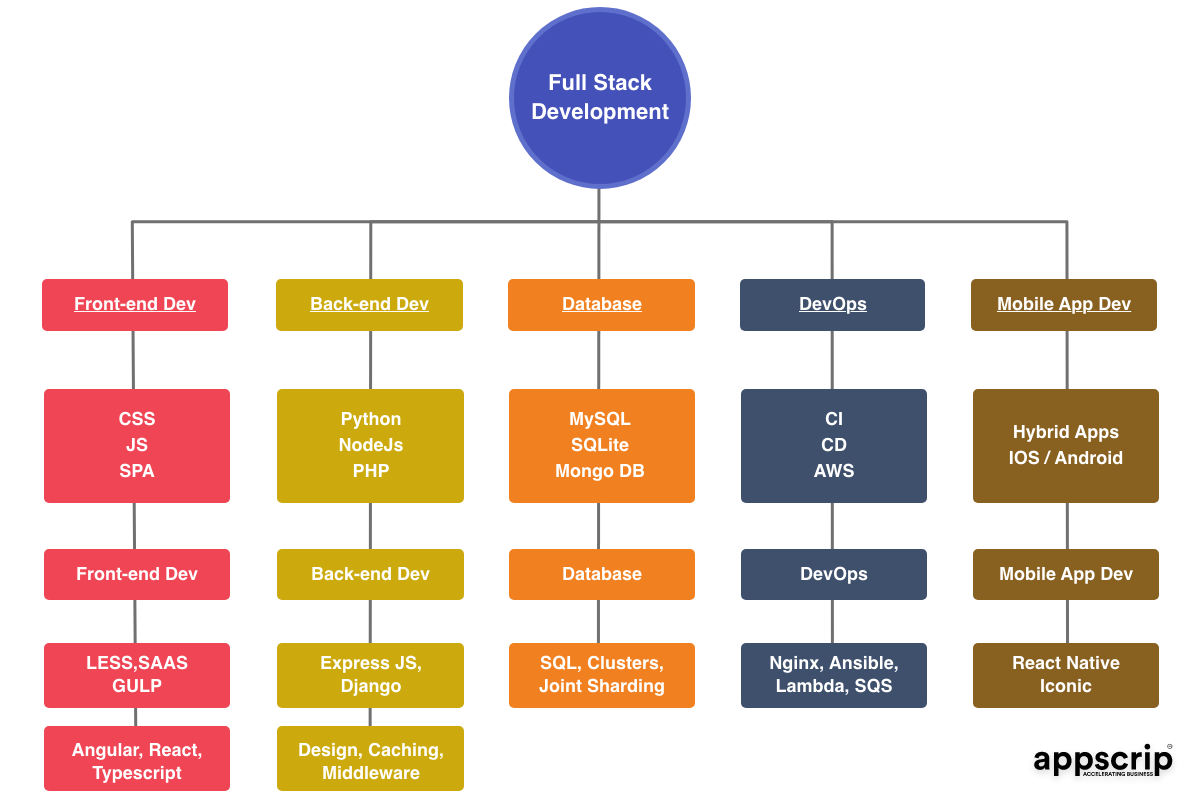
The industry definition of a Full Stack Developer is an engineer who can work on different levels of an application stack. The term stack refers to the combination of components and tools that make up the application. The components could be in the front-end (client side) and the back-end (server side) of the web application.

**1.1 IMPORTANCE OF FULL STACK DEVELOPMENT:**

The main objective of full stack engineer is to keep every part of the system running smoothly. A Full Stack Developer can perform tasks ranging from resizing an image or text in a webpage to patching the kernel.

Change is the only constant and in our ever-changing world of technology, this has been proven time and again. With growing demand and constant development, technology is always reinventing itself to provide consumers with better and sharper products and software tools. With more and more people moving towards the virtual world, anyone with any sort of business or enterprise needs to have an online presence as well. While there are options for business owners to build and to develop their own sites, there also exists the option of hiring a professional website developer for the purpose. A website would need a professional hand. This is because consumers have numerous choices online. Hence, their attention span is very limited so, one’s webpage has not only to be a platform to sell or promote for the owner but also a platform that is informative, technically sound and is easy to use with all the required security codes in place.

Full stack web developers have the ability to design complete web application and websites. They work on the frontend, backend, database and debugging of web application or websites.



**Figure 1.1: Basic Structure of Full Stack Development**

**1.2 FRONT END:**

Front-end web development, also known as client-side development is the practice of producing HTML, CSS and JavaScript for a website or Web Application so that a user can see and interact with them directly. The challenge associated with front end development is that the tools and techniques used to create the front end of a website change constantly and so the developer needs to constantly be aware of how the field is developing.

The objective of designing a site is to ensure that when the users open up the site, they see the information in a format that is easy to read and relevant. This is further complicated by the fact that users now use a large variety of devices with varying screen sizes and resolutions thus forcing the designer to take into consideration these aspects when designing the site. They need to ensure that their site comes up correctly in different browsers (cross-browser), different operating systems (cross-platform) and different devices (cross-device), which requires careful planning on the side of the developer.

Front end development manages everything that users visually see first in their browser or application. Front end developers are responsible for the look and feel of a site. It is the visible part of a website or web application which is responsible for user experience. The user directly interacts with the front-end portion of the web application or website.

**1.3 BACK END:**

Back end development refers to the server side of an application and everything that communicates between the database and the browser. It is responsible for managing the database through queries and APIs by client-side commands. Back end development refers to the server side of development where you are primarily focused on how the site works. Making updates and changes in addition to monitoring functionality of the site will be your primary responsibility. This type of web development usually consists of three parts: a server, an application, and a database. Code written by back end developers is what communicates the database information to the browser. Anything you can’t see easily with the eye such as databases and servers is the work of a back-end developer. Back end developer positions are often called programmers or web developers.

**CHAPTER 2**

**WEB-DEVELOPMENT**

**Web development** is a broad term for the work involved in developing a web site for the Internet (World Wide Web) or an intranet (a private network). Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications, electronic businesses, and social network services. A more comprehensive list of tasks to which web development commonly refers, may include web engineering, web design, web content development, client liaison, client-side/side scripting, web server and network security configuration, and e-commerce development. Among web professionals, "web development" usually refers to the main non-design aspects of building web sites: writing markup and coding. Most recently Web development has come to mean the creation of content management systems or CMS. These CMS can be made from scratch, proprietary or open source. In broad terms the CMS acts as middleware between the database and the user through the browser. A principle benefit of a CMS is that it allows non-technical people to make changes to their web site without having technical knowledge.

For larger organizations and businesses, web development teams can consist of hundreds of people (web developers) and follow standard methods like Agile methodologies while developing websites. Smaller organizations may only require a single permanent or contracting developer, or secondary assignment to related job positions such as a graphic designer or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department. There are three kind of web developer specialization: front-end developer, back-end developer, and full-stack developer.

**2.1 WEB-SITE:**

A **website** is a collection of related web pages, including multimedia content, typically identified with a common domain name, and published on at least one web server. A website may be accessible via a public Internet Protocol (IP) network, such as the Internet, or a private local area network (LAN), by referencing a uniform resource locator (URL) that identifies the site.

Web pages, which are the building blocks of websites, are documents, typically composed in plain text interspersed with formatting instructions of Hypertext Markup Language (HTML, XHTML). They may incorporate elements from other websites with suitable markup anchors. Web pages are accessed and transported with the Hypertext Transfer Protocol (HTTP), which may optionally employ encryption (HTTP Secure, HTTPS) to provide security and privacy for the user. The user's application, often a web browser, renders the page content according to its HTML markup instructions onto a display terminal.

Hyperlinking between web pages conveys to the reader the site structure and guides the navigation of the site, which often starts with a home page containing a directory of the site web content. Some websites require user registration or subscription to access content. Examples of subscription websites include many business sites, news websites, academic journal websites, gaming websites, file-sharing websites, message boards, web-based email, social networking websites, websites providing real-time stock market data, as well as sites providing various other services. As of 2016 end users can access websites on a range of devices, including desktop and laptop computers, tablet computers, smartphones and smart TVs. A web site consists of web pages which are interconnected to each other and contain various data and functionalities.

**2.2 WEB-PAGE:**

A **web page**, or **webpage**, is a document that is suitable for the World Wide Web and web browsers. A web browser displays a web page on a monitor or mobile device. The web page is what displays, but the term also refers to a computer file, usually written in HTML or comparable markup language. Web browsers coordinate the various web resource elements for the written web page, such as style sheets, scripts, and images, to present the web page. Typical web pages provide hypertext that includes a navigation bar or a sidebar menu to other web pages via hyperlinks, often referred to as links.

On a network, a web browser can retrieve a web page from a remote web server. On a higher level, the web server may restrict access to only a private network such as a corporate intranet or it provides access to the World Wide Web. On a lower level, the web browser uses the Hypertext Transfer Protocol (HTTP) to make such requests.

A static web page is delivered exactly as stored, as web content in the web server's file system, while a dynamic web page is generated by a web application that is driven by server- side software or client-side scripting. Dynamic website pages help the browser (the client) to enhance the web page through user input to the server.

**2.3​ WEB BROWSER**

A **web browser**​ (commonly referred to as a ​**browser**​) ​ is a software application for accessing information on the World Wide Web. When a user requests a web page from a particular website, the web browser retrieves the necessary content from a web server and then displays the page on the user’s device.

A web browser is not the same thing as a search engine, though the two are often confused. For a user, a search engine is just a website that stores searchable data about other websites. However, to connect to a website’s server and display its web pages, a user must have a web browser installed.

Web browsers are used on a range of devices, including desktops, laptops, tablets, and smartphones. In 2019, an estimated 4.3 billion people used a browser. The most used browser is Google Chrome, with a 64% global market share on all devices, followed by Safari with 18%.

##### **2.4 MOBILE WEB BROWSER**

A **mobile browser** is a web browser designed for use on a mobile device such as a mobile phone or PDA. Mobile browsers are optimized so as to display Web content most effectively for small screens on portable devices. Mobile browser software must be small and efficient to accommodate the low memory capacity and low-bandwidth of wireless handheld devices. Typically, they were stripped-down web browsers, however, some recent mobile browsers can handle latest technologies also such as CSS 3, JavaScript, and Ajax.

Websites designed so that they may be accessed from these browsers are referred to as wireless portals or collectively as the Mobile Web. They may automatically create “mobile” version of each page, for example the Wikipedia website. ​

**CHAPTER-3**

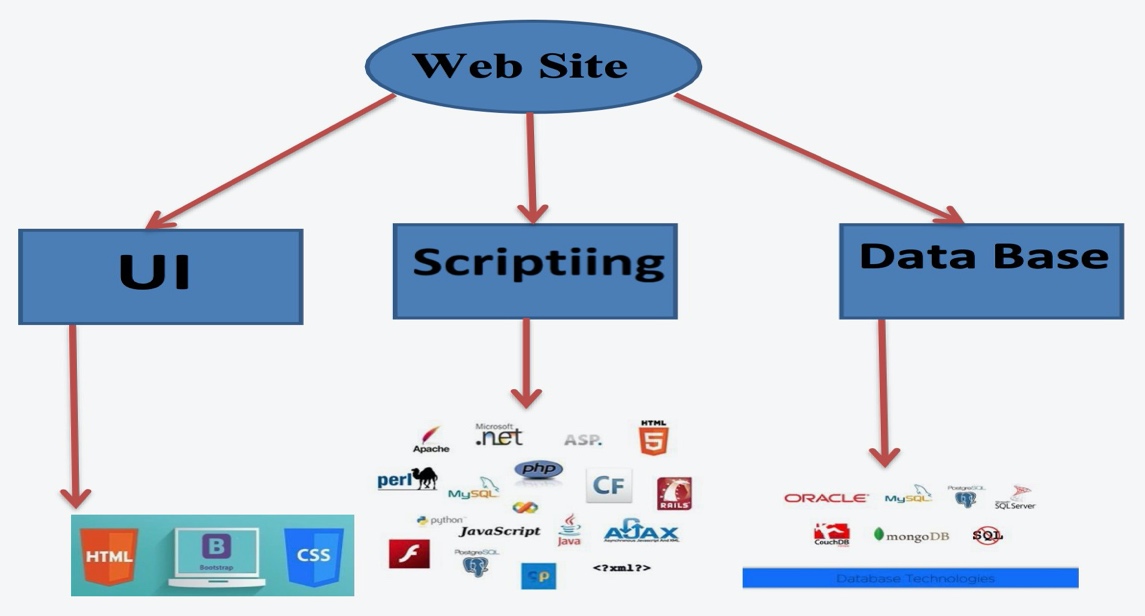
**STEPS TO CREATE A WEBSITE**

Creating a web site requires multiple steps which includes the following:

* Creating a UI (User interface)
* Scripting (Both at server end and client end)
* Creating a backend or the database

**System parts**

* **Back-End**
  + Operating System (OS)
  + Firewall
  + Web server
  + Database (SQL or NoSQL)
  + Caches
  + Message queuing software
  + Application
* **Front-End**
  + HTML
  + CSS
  + JavaScript
  + jQuery



**Figure 1.2: System Parts**

**3.1 UI DEVELOPMENT:**

Technologies that are mostly used to develop a User Interface are:

* HTML
* CSS
* Bootstrap

**3.1.1 HTML:**

**Hypertext Mark-up Language** (**HTML**) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a webserver or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input />introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript which affect the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997. HTML markup consists of several key components, including those called tags (and their attributes), character-based data types, character references and entity references. HTML tags most commonly come in pairs like <h1> and </h1>, although some represent empty elements and so are unpaired, for example <img>. The first tag in such a pair is the start tag, and the second is the end tag (they are also called opening tags and closing tags).

Another important component is the HTML document type declaration, which triggers standards mode rendering.

The following is an example of the classic Hello world program, a common test employed for comparing programming languages, scripting languages and mark-up languages. This example is made using 9 lines of code:

**General Syntax of HTML**

<!DOCTYPE html> <html>

<head>  
<title>This is a title</title>

</head> <body>

<p>Hello world! </p> </body>

</html>

* The text between <html> and </html> describes the web page.
* The text between <body> and </body> is the visible page content.
* The mark-up text "<title>This is a title</title>" defines the browser page title.
* The Document Type Declaration <!DOCTYPE html> is for HTML5. If a declaration is not included, various browsers will revert to "quirks mode" for rendering.

**3.1.2 CSS:**

**Cascading Style Sheets** (**CSS**) is a style sheet language used for describing the presentation of a document written in a mark-up language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .CSS file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same mark-up page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing mark-up in the documents. The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/CSS is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

**Types of CSS:**

* **Inline CSS:**

In this CSS is applied in between the tags

E.g.: <tag style= styling>Hello World</tag>

* **Internal CSS:**

In this the CSS code is defined inside the style tag in the head section of the HTML page.

**General Syntax:**

<html>  
<head>

<style>  
<! -- CSS STYLING -- > </style>

</head> </html>

* **External CSS:**

In this the CSS code is written on another page and is linked to the HTML page. It is advantageous to use this type of styling as we can use the same file to style various HTML pages.

External CSS uses the extension .CSS and is applied using the following syntax

<html>

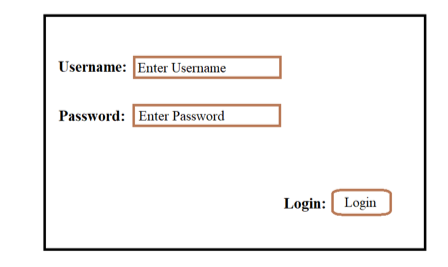
<head>  
<link relation=” stylesheet” type=”CSS” href=”URL to the page”>

</head> </html>

All the CSS style types are important but can be used in different situations.

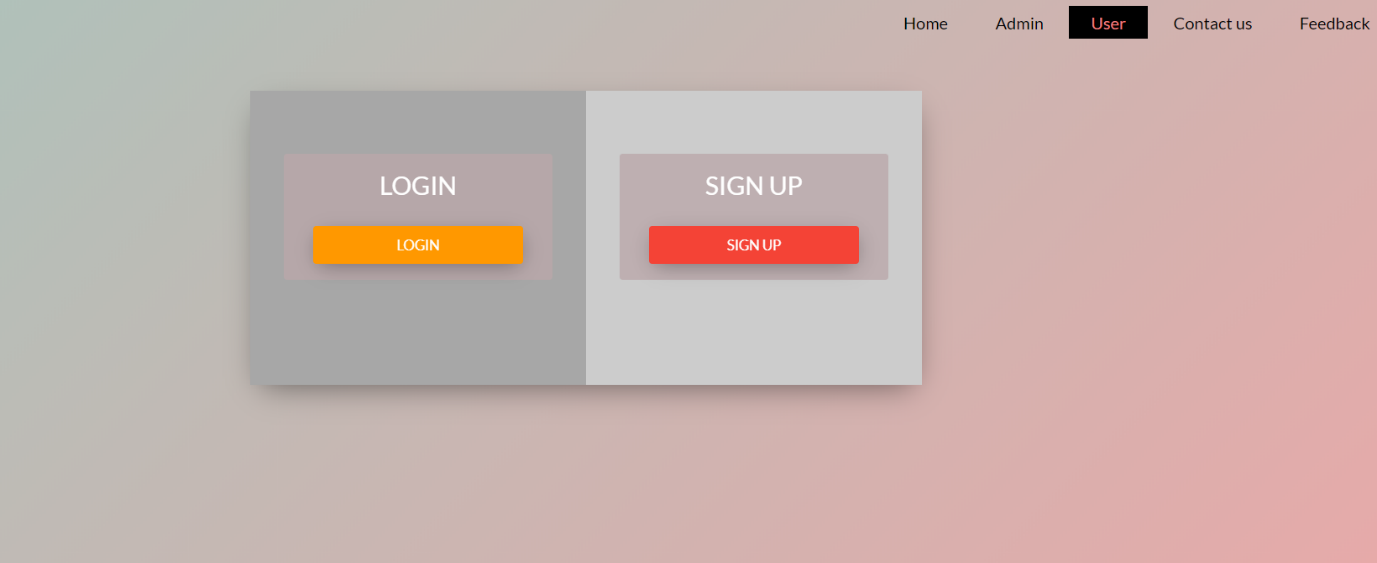
* Inline CSS is used when only small changes are to be done to the HTML tag and the changes are to be reflected only to that specific tag.
* Internal CSS is used when the individual HTML pages have to be designed differently. This also slows the page load system if the internal styling is long.
* External CSS files are maintained to design multiple pages and use common styles over various pages. It is useful as it helps in managing the resources in an easy manner.

**Before using CSS in HTML page**



**Figure 1.3: Login Page without CSS**

**After using CSS in HTML Page**



**Figure 1.4: Login Page with CSS**

**3.1.3 BOOTSTRAP:**

**Bootstrap** is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

Bootstrap is the second most-starred project on GitHub, with more than 107,000 stars and 48,000 forks. Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden. According to twitter developer Mark Otto: “A super small group of developers and I got together to design and build a new internal tool and saw an opportunity to do something more. Through that process, we saw ourselves build something much more substantial than another internal tool. Months later, we ended up with an early version of Bootstrap as a way to document and share common design patterns and assets within the company.”

After a few months of development by a small group, many developers at Twitter began to contribute to the project as a part of Hack Week, a hackathon-style week for the Twitter development team. It was renamed from Twitter Blueprint to Bootstrap, and released as an open source project on August 19, 2011. It has continued to be maintained by Mark Otto, Jacob Thornton, and a small group of core developers, as well as a large community of contributors.

On January 31, 2012, Bootstrap 2 was released, which added a twelve-column responsive grid layout system, inbuilt support for Glypicans, several new components, as well as changes to many of the existing components.

On August 19, 2013, Bootstrap 3 was released, which redesigned components to use flat design, and a mobile first approach.

On October 29, 2014, Mark Otto announced that Bootstrap 4 was in development. The first alpha version of Bootstrap 4 was released on August 19, 2015.

Bootstrap 3 supports the latest versions of the Google Chrome, Firefox, Internet Explorer, Opera, and Safari (except on Windows). It additionally supports back to IE8 and the latest Firefox Extended Support Release (ESR). Since 2.0, Bootstrap supports responsive web design. This means the layout of web pages adjusts dynamically, taking into account the characteristics of the device used (desktop, tablet, mobile phone). Starting with version 3.0, Bootstrap adopted a mobile-first design philosophy, emphasizing responsive design by default.

**Installing and linking bootstrap to the HTML page:**

* Install bootstrap from https://getbootstrap.com/
* Copy the bootstrap.min.css file to your CSS folder and link it to the HTML page in the similar manner to how any other CSS file is linked.
* Link the bootstrap.min.js file which is present in the JS folder of the bootstrap. It can be linked using script tag.

E.g.: <script src=”URL to bootstrap.min.js”></script>

* Now use bootstrap classes to reduce the work of designing which was earlier done

through CSS.

* 1. **SCRIPTING:**

There are two scripting methodologies.

**1) Server-side scripting:** This scripting is done at the server end

**2) Client-side scripting:** This scripting is done at the client end or the browser.

* + 1. **SERVER-SIDE SCRIPTING:**

**Server-side scripting** is a technique used in web development which involves employing scripts on a web server which produce a response customized for each user’s (client’s) request to the website. The alternative is for the web server itself to deliver a static web page. Scripts can be written in any of a number of server-side scripting languages that are available (see below). Server-side scripting is distinguished from client-side scripting where embedded scripts, such as JavaScript, are run client-side in a web browser, but both techniques are often used together.

Server-side scripting is often used to provide a customized interface for the user. These scripts may assemble client characteristics for use in customizing the response based on those characteristics, the user’s requirements, access rights, etc. Server-side scripting also enables the website owner to hide the source code that generates the interface, whereas with client-side scripting, the user has access to all the code received by the client. A down-side to the use of server-side scripting is that the client needs to make further requests over the network to the server in order to show new information to the user via the web browser. These requests can slow down the experience for the user, place more load on the server, and prevent use of the application when the user is disconnected from the server.

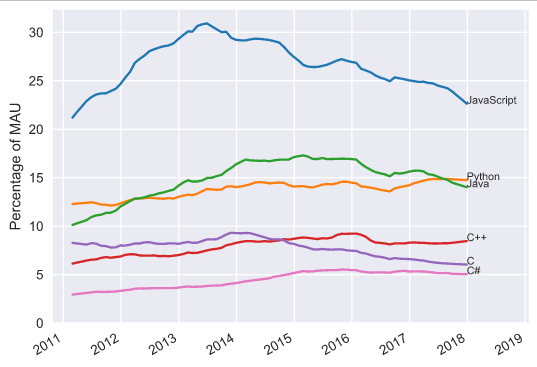
When the server serves data in a commonly used manner, for example according to the HTTP or FTP protocols, users may have their choice of a number of client programs (most modern web browsers can request and receive data using both of those protocols). In the case of more specialized applications, programmers may write their own server, client, and communications protocol that can only be used with one another.

Programs that run on a user’s local computer without ever sending or receiving data over a network are not considered clients, and so the operations of such programs would not be considered client-side operations.

**3.2.1.1 SERVER-SIDE SCRIPTING LANGUAGES:**

There are several languages that can be used for server-side programming:

* + Angular js
  + ASP.NET (C# OR Visual Basic)
  + C++
  + Java and JSP
  + Node JS
  + Python Ruby on Rails and so on.



**Figure 1.5: Programming Language Popularity by GitHub Projects**

**One of the server-side scripting language used in our project is:**

**Node.JS:**

Node.js is an open source server-side technology it works on different platforms like windows, Linux, Unix, Macules, JavaScript on the server and can generate the dynamic page content. It creates, delete, read, write and can collect form data.Node.js modification of database it supports

**Here is how Node.js handles a file request:**

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request. It runs single-threaded, non-blocking, asynchronously programming, which is very memory efficient.

**Installing Node.JS:**

Download Node.JS -The official Node.js website has installation instructions for Node.js: <https://nodejs.org>

* + 1. **CLIENT-SIDE SCRIPTING:**

Client-side scripting is changing interface behaviors within a specific web page in response to mouse or keyboard actions, or at specified timing events. In this case, the dynamic behavior occurs within the presentation. The client-side content is generated on the user's local computer system.

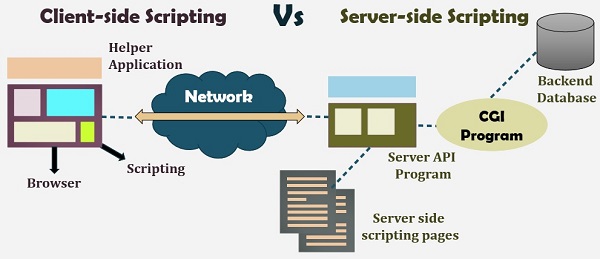
Such web pages use presentation technology called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. Client-side scripting also allows the use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden frame, XML Http Requests, or a Web service.

The first widespread use of JavaScript was in 1997, when the language was standardized as ECMAScript and implemented in Netscape 3.

**Example:**

The client-side content is generated on the client's computer. The web browser retrieves a page from the server, then processes the code embedded in the page (typically written in JavaScript) and displays the retrieved page's content to the user.

The most popularly used client-side scripting languages is **Java Script**. Flow of request from browser to server:



**Figure 1.6: Client-Side and Server-Side Scripting**

* 1. **DATABASE:**

A **database** is an organized collection of data. It is the collection of schemas, tables, queries, reports, views, and other objects. The data are typically organized to model aspects of reality in a way that supports processes requiring information, such as modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

A **database management system** (**DBMS**) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data. A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases. Well-known DBMSs include MySQL, PostgreSQL, MongoDB, MariaDB, Microsoft SQL Server, Oracle, Sybase, SAP HANA, MySQL and IBM DB2. A database is not generally portable across different DBMSs, but different DBMS can interoperate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS. Database management systems are often classified according to the database model that they support; the most popular database systems since the 1980s have all supported the relational model as represented by the SQL language. Sometimes a DBMS is loosely referred to as a "database".

* 1. **SQL:**

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language, data manipulation language, and data control language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages for Edgar F. Codd’s relational model, as described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks." Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, most SQL code is not completely portable among different database systems without adjustments.

* 1. **QUERIES:**

The most common operation in SQL, the query, makes use of the declarative SELECT statement. SELECT retrieves data from one or more tables, or expressions. Standard SELECT statements have no persistent effects on the database. Some non-standard implementations of SELECT can have persistent effects, such as the SELECT INTO syntax provided in some databases. Queries allow the user to describe desired data, leaving the database management system (DBMS) to carry out planning, optimizing, and performing the physical operations necessary to produce that result as it chooses.

A query includes a list of columns to include in the final result, normally immediately following the SELECT keyword. An asterisk ("\*") can be used to specify that the query should return all columns of the queried tables. SELECT is the most complex statement in SQL, with optional keywords and clauses that include:

* The FROM clause, which indicates the table(s) to retrieve data from. The FROM clause can include optional JOIN subclauses to specify the rules for joining tables.
* The WHERE clause includes a comparison predicate, which restricts the rows returned by the query. The WHERE clause eliminates all rows from the result set where the comparison predicate does not evaluate to True.
* The GROUP BY clause projects rows having common values into a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregation functions or to eliminate duplicate rows from a result set. The WHERE clause is applied before the GROUP BY clause.
* The HAVING clause includes a predicate used to filter rows resulting from the GROUP BY clause. Because it acts on the results of the GROUP BY clause, aggregation functions can be used in the HAVING clause predicate.
* The ORDER BY clause identifies which column[s] to use to sort the resulting data, and in which direction to sort them (ascending or descending). Without an ORDER BY clause, the order of rows returned by an SQL query is undefined.
* The DISTINCT keyword eliminates duplicate data.

**CHAPTER-4**

**SCRIPTING LANGUAGES**

**4.1 JAVA SCRIPT:**

**JavaScript**, often abbreviated as "JS", is a high-level, dynamic, untyped, and interpreted run-time language. It has been standardized in the ECMAScript language specification. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production; the majority of websites employ it, and all modern Web browsers support it without the need for plug-ins. JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions, but does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

Although there are strong outward similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two are distinct languages and differ greatly in their design. JavaScript was influenced by programming languages such as self and Scheme. JavaScript is also used in environments that are not Web-based, such as PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines (VMs) and platforms built upon them have also increased the popularity of JavaScript for server-side Web applications. On the client side, developers have traditionally implemented JavaScript as an interpreted language, but more recent browsers perform just-in-time compilation. Programmers also use JavaScript in video-game development, in crafting desktop and mobile applications, and in server-side network programming with run-time environments such as Node.js.

**4.2 JQUERY:**

**jQuery** is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It is free, open-source software using the permissive MIT license. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin. jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its selector engine (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard Selectors API.

Microsoft and Nokia bundle jQuery on their platforms. Microsoft includes it with Visual Studio for use within Microsoft's ASP.NET AJAX and ASP.NET MVC frameworks while Nokia has integrated it into the Web Run-Time widget development platform.

**4.3 AJAX:**

**Ajax** (also **AJAX** short for "asynchronous JavaScript and XML") is a set of Web development techniques using many Web technologies on the client side to create asynchronous Web applications. With Ajax, Web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page. By decoupling the data interchange layer from the presentation layer, Ajax allows for Web pages, and by extension Web applications, to change content dynamically without the need to reload the entire page. In practice, modern implementations commonly substitute JSON for XML due to the advantages of being native to JavaScript.

Ajax is not a single technology, but rather a group of technologies. HTML and CSS can be used in combination to mark up and style information. The DOM is accessed with JavaScript to dynamically display – and allow the user to interact with – the information presented. JavaScript and the XML Http Request object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads.

**4.4 JSON:**

In computing, **JavaScript Object Notation** is an open- standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). It is a very common data format used for asynchronous browser/server communication, including as a replacement for XML in some AJAX-style systems. JSON is a language-independent data format. It was derived from JavaScript, but as of 2017 many programming languages include code to generate and parse JSON-format data. The official Internet media type for JSON is application/json. JSON filenames use the extension. json.

**4.5 Node Js**

**Node.js** is an open-source, cross-platform, JavaScript runtime environment (Framework) that executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server- and client-side scripts.

Though .js is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).

**CHAPTER 5**

**SOFTWARE REQUIREMENTS**

* 1. **HARDWARE REQUIREMENTS:**

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and requirements are also important.

|  |  |
| --- | --- |
| Processor | Intel CORE i5 |
| RAM | 8.0 GB |
| Hard Disk Drive | 500 |

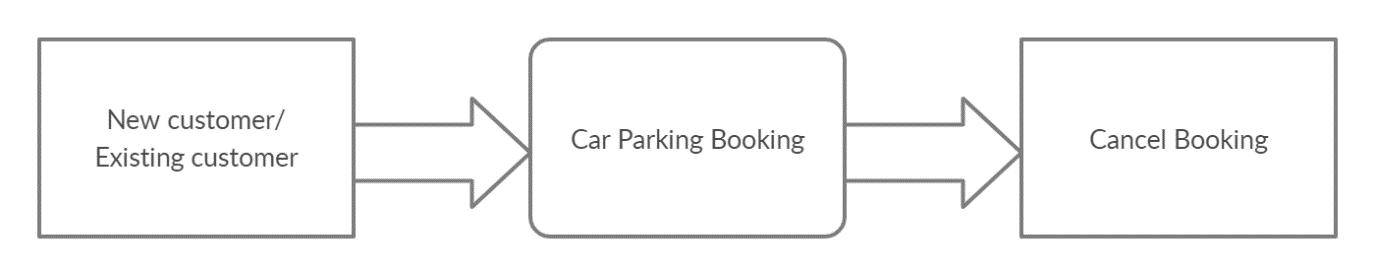
**5.2 SOFTWARE REQUIREMENTS:**

|  |  |
| --- | --- |
| Number | Description |
| 1 | Windows 10 |
| 2 | HTML/CSS/Ajax/JavaScript/Bootstrap. |
| 3 | Apache server/XAMPSERVER |
| 4 | Node js |
| 5 | MySQL |
| 6 | Compiler: MSVC11(Visual C++ 2012) |

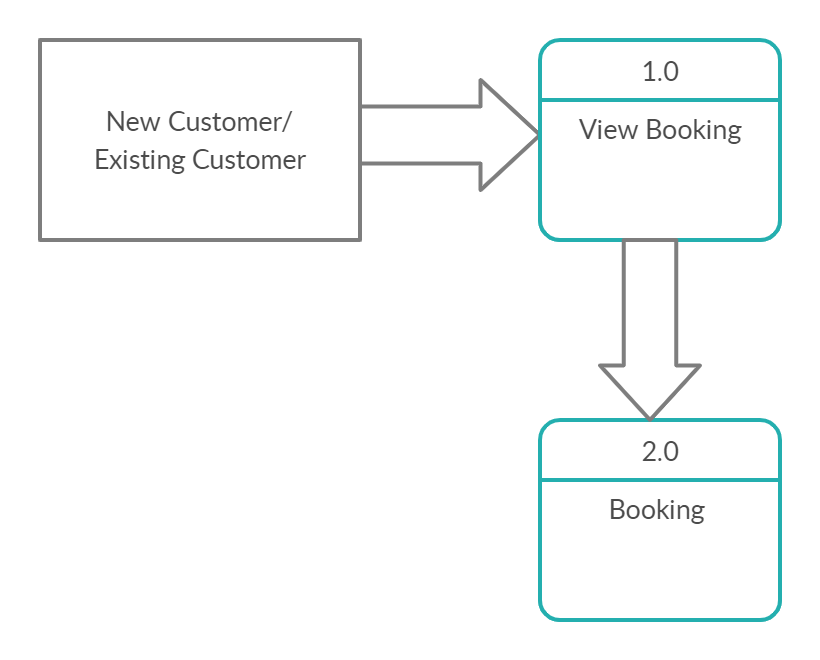
**CHAPTER 6**

**WORK FLOW**

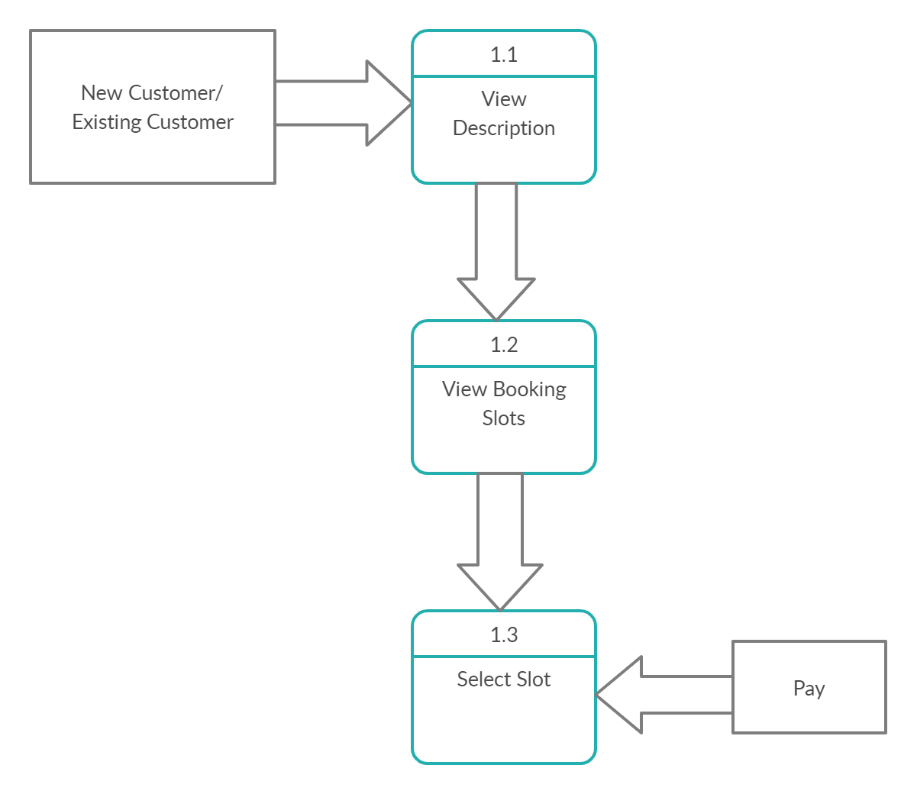
**6.1 Context Level Diagram**



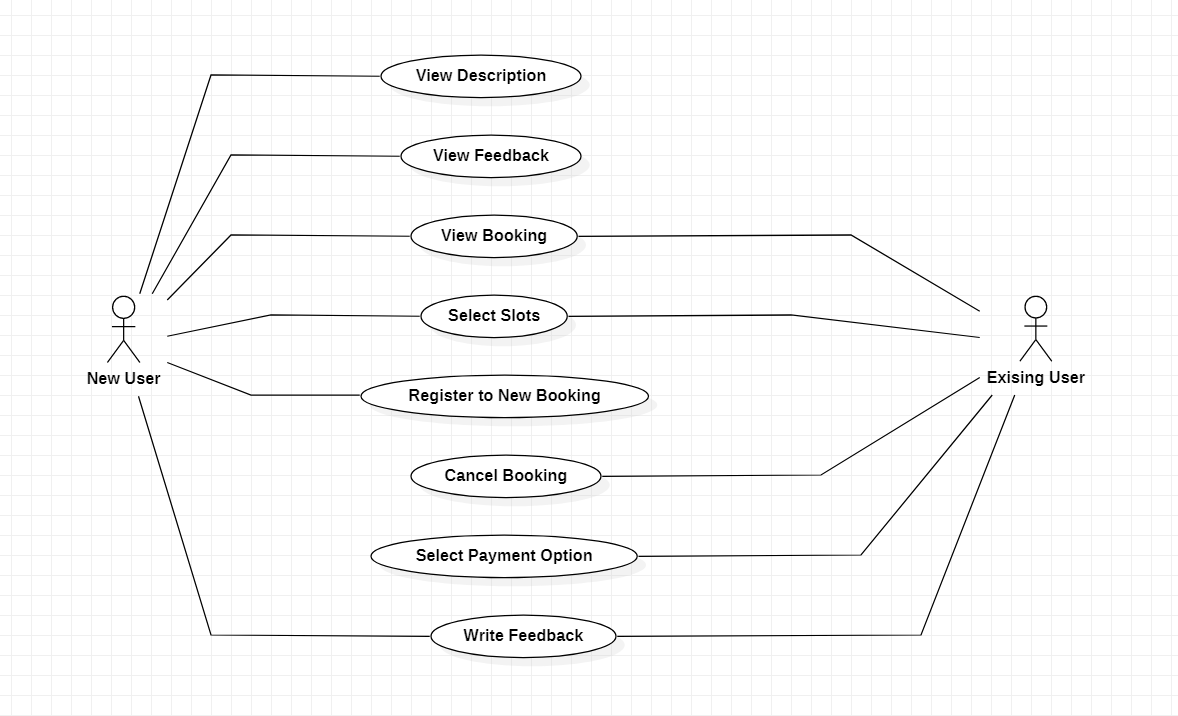
**6.2 Level 1-DFD**



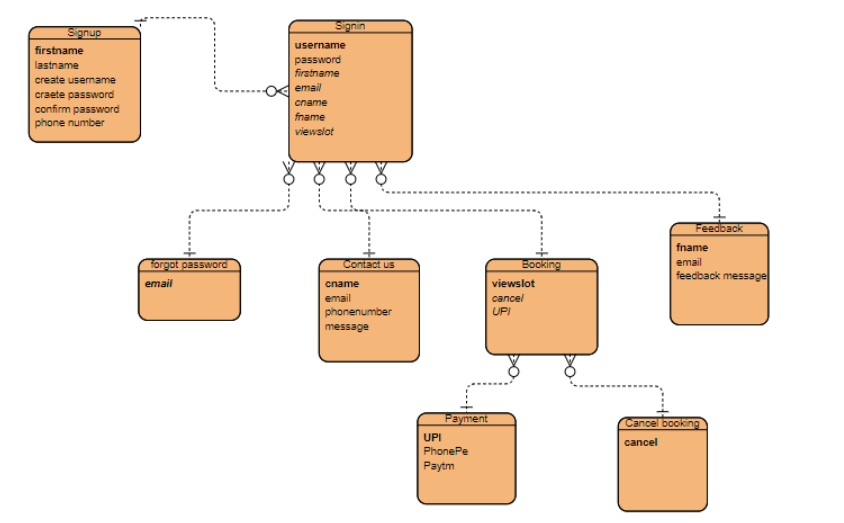
**6.3 Level 2-DFD**



**6.4 Use Case Diagram**



**6.5 E-R Diagram**

****

**CHAPTER 7**

**PROJECT**

**RADIO-FREQUENCY IDENTIFICATION (RFID)**

**PARKING SYSTEM**

**7.1 PROJECT AIMS AND OBJECTIVES:**

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

* + - Attendance System
    - Tracking System
    - Demonstrate hazel free parking.
    - Improve security for both security guards and users
    - Displays the vacant slots on the display at the entrance of the parking
    - Facilitate access control for users and improve traffic flow during peaks period
    - Monitor vehicle’s movement that enter or leave the specific area or place by scanning the RFID tag.

**7.1.1 BACKGROUND OF PROJECT:**

RFID Parking System avoid the cramming in the car parking area by implementing an efficient car parking system along with a user-friendly application for an ease of use. Normally at public places such as multiplex theatres, market areas, hospitals, function-halls, offices and shopping malls, one experiences the discomfort in looking out for a vacant parking slot, though it’s a paid facility with an attendant/ security guard. The parking management system is proposed to demonstrate hazel free parking. The proposed system uses infrared transmitter-receiver pairs that remotely communicate the status of parking occupancy to the raspberry pi and displays the vacant slots on the display at the entrance of the parking so that the user gets to know the availability /unavailability of parking space prior to his/her entry into the parking place. Implementation involves minimal human interaction and provides a seamless parking experience thereby reducing a lot of time wasted by the user in parking his/her vehicle.

Due to the large amount of time people spend on parking their car, there is a need for a better system which reduces the time taken to park vehicles, the current system which exists involves the client(user) to wait in a queue where each car has to wait for a token to be generated, this token is kept by the client till he is done with his work and on the time of exit has to return the token, which then calculates the time spent and the bill is generated. But the problem with this current system is the queue, each person is unique in their own way, so it depends from person to person, some people might quickly take the token and park their car, while some might take a longer time, leading to delay. The other problem with the current system is that there is no automatic way in which the user knows where there is parking space; he relies on another person (security guard) for the parking. To overcome these problems, we use RFID and IR sensors. RFID is used instead of generating a token again and again, each user goes through a onetime registration process where a RFID tag is attached to his vehicle, This RFID tag has information about him, which is unique to him (like Aadhar number) so there is no queue as such, users just pass through the gate seamlessly and park their vehicle. Once they pass through the entry gate, they can see a huge display which has the live parking slots available, once a car is parked the IR sensor updates the database. On exiting the tag is detected again and the amount is calculated and automatically deducted from the users account. A similar approach has been taken by Sumy RFID based parking solution, who provide their service to residential areas, societies, factories and industrial areas. Another solution is provided by FAST tag which uses RFID tags for automatic deduction of toll charges and lets you pass the toll gate without having to stop.

**7.1.2 RFID SYSTEM FUNDAMENTALS:**

Basically, an RFID system consists of an antenna or coil, a transceiver (with decoder) and a transponder (RF tag) electronically programmed with unique information. There are many different types of RFID systems in the market. These are categorized on the basis of their frequency ranges. Some of the most commonly used RFID kits are low-frequency (30-500kHz), mid-frequency (900kHz-1500MHz) and high-frequency (2.4-2.5GHz).

### 7.1.3 COMPONENTS:

* AT89C51 Microcontroller
* 11.0592 MHz Quartz Crystal
* 33pF Ceramic Capacitor – 2
* 10KΩ Resistors – 2
* Electrolytic Capacitor – 10µF
* Push Button
* 20 X 4 LCD Display
* 1KΩ Resistor (for pull-up)-5
* 10KΩ POT
* AT89C51 Programmer Board

### 7.1.4 COMPONENT DESCRIPTION:

#### **DS1307 RTC:**

A Real time clock or RTC, as the name indicates, is a clock module in the form of IC, which keeps track of the current time. RTCs are present in almost all electronic devices like personal computers, servers and embedded systems, which needs to keep accurate time.

The main advantage of RTC is that they run on battery and keeps the clock and calendar running even if there is power failure. The Real Time Clock used in this project is DS1307.

#### **EEPROM (24C04 – 4K):**

EEPROM stands for Electrically Erasable Programmable Read Only Memory. It is a type of non-volatile memory i.e. it can hold the data even when the power is removed and the data can be electrically erased and reprogrammed.

EEPROMs are used in electronic devices to store small amounts of data.

The EEPROM used in this project is 24C04 which has a memory of 4 KB.

**7.1.5 CIRCUIT DIAGRAM:**

The main connections for basic functioning of the microcontroller include a reset circuit, oscillator circuit and EA Pin. Reset circuit consists of a push button, 10KΩ resistor and a 10µF capacitor.

External oscillator circuit consists of an 11.0592 MHz quartz crystal and two 33pF capacitors. Finally, a 10KΩ resistor is used with EA pin to pull it high.

The next hardware we need to connect is the 20 x 4 LCD. The pin configuration of a 20 x 4 LCD is similar to that of a 16 x 2 LCD. The only difference is that a 20 x 4 LCD has few extra segments. In order to access those extra segments, we need to program the microcontroller accordingly.

P3.6, GND and P3.7 pins are connected to the three control pins of the 20 x 4 LCD i.e. RS, RW and E. The data pins of the 20 x 4 LCD are connected to Port 1 of the microcontroller.

The next component we are going to connect is the RTC. First, we have to connect a 32.786 MHz crystal oscillator between the oscillator pins of the RTC IC.

Then connect a 3V coin type Lithium battery to the VBAT terminal of the RTC IC. Finally, the I2C terminals i.e. SCL and SDA of the RTC IC are connected to PORT0 pins.

Hence, they must be pulled high with 1K resistors. Now you can connect SCL and SDA to P0.0 and P0.1 of the microcontroller.

After connecting the RTC, now we are going to connect the EEPROM. First connect the SCL and SDA pins of the EEPROM IC to the P0.0 and P0.1 pins of the microcontroller. Then, connect the address input pins of the EEPROM to Ground.

Finally, connect the RFID reader to the controller. Connect the TX pin of the RFID Reader to RXD pin (P3.0) of the Controller and RX of RFID is connected to TXD (P3.1) of microcontroller. RFID Security Access Control System using 8051.

**7.1.6 WORKING:**

* The project shows the design of an RFID based car parking system using 8051 micro controllers, in which only authorized personnel with valid RFID card are allowed access to park and also the IN/OUT time details along with the fare are automatically calculated.
* When the circuit is switched ON, current time is displayed on the LCD display.
* When the ID card is detected by the reader, a unique card number is sent to the microcontroller.
* If the card number is matched with saved number in microcontroller or database, the microcontroller will allow the car in order to park in the secured area.
* The entry time details of the particular card are stored in the EEPROM. A welcome message along with the in-time details are displayed on the LCD.
* When the same card is swiped again, the microcontroller will display the in and out time along with the calculated fare details on the LCD.

**7.1.7 ADVANTAGES AND APPLIACATIONS:**

* RFID based Car Parking System is implemented in this project and can be used to eliminate the hassle of manual operation of parking system.
* This system can help in reducing cost, increase in productivity and saves time.
* Accurate timing details are measured with the help of RTC Module.
* Prepaid and postpaid cards can be integrated with the system for easy payment options.

**7.1.8 DISADVANTAGES:**

* Materials like metal & liquid can impact signal.
* Sometimes not as accurate or reliable as barcode scanners.
* Cost – **RFID** readers can be 10x more expensive than barcode scanners.
* Implementation can be difficult & time consuming.

**7.1.9 LIMITATIONS:**

* Only fare details are calculated but can be extended to credit card or other monetary card system where the fare amount is automatically deducted.
* Further tracking systems like number plate tracking, face tracking etc. can be implemented for accuracy and security.

**7.2 TECNOLOGIES USED:**

* HTML
* CSS
* Bootstrap
* Java Script
* jQuery
* AJAX

**Server:**

**Database:** MySQL

**Operating System: Windows 10**

**Wire framing tool:**

**Team Size:** 3

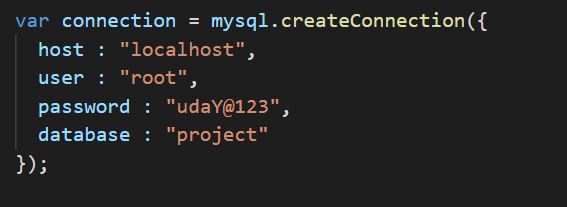
**7.3 TECHNICAL DETAILS:**

* Front end is designed using HTML, CSS and Bootstrap. Ajax used to perform behind the screen requests and JavaScript used to perform client-side scripting
* Backend is based on MySQL based RDB (Relational Data Base) model.
* The SQL queries are run using the CI SQL library functions
* Backend online host includes a centralized database resident on the server, the script which is built in Node.JS used to SQL query the database on user’s request for transaction of data
* The forms are made using the HTML, Bootstrap for designing and Node.JS, sql for back-end JavaScript, AJAX and jQuery used for client-side scripting and Node.JS for the server-side development

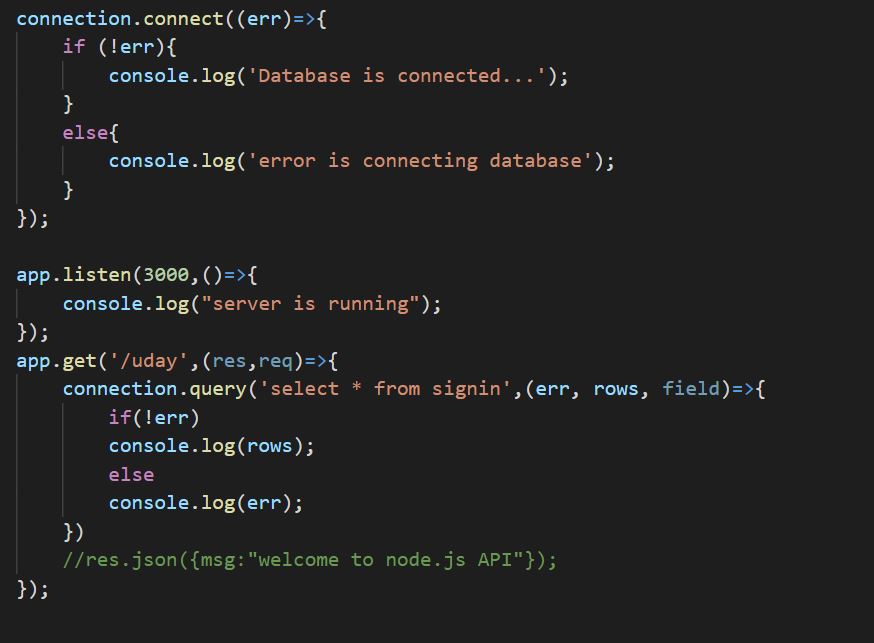
**CHAPTER 8**

**BACK-END SCREENSHOTS**

**Configuration database(config.db.js)**



**Connection of MY SQL DATABASE**

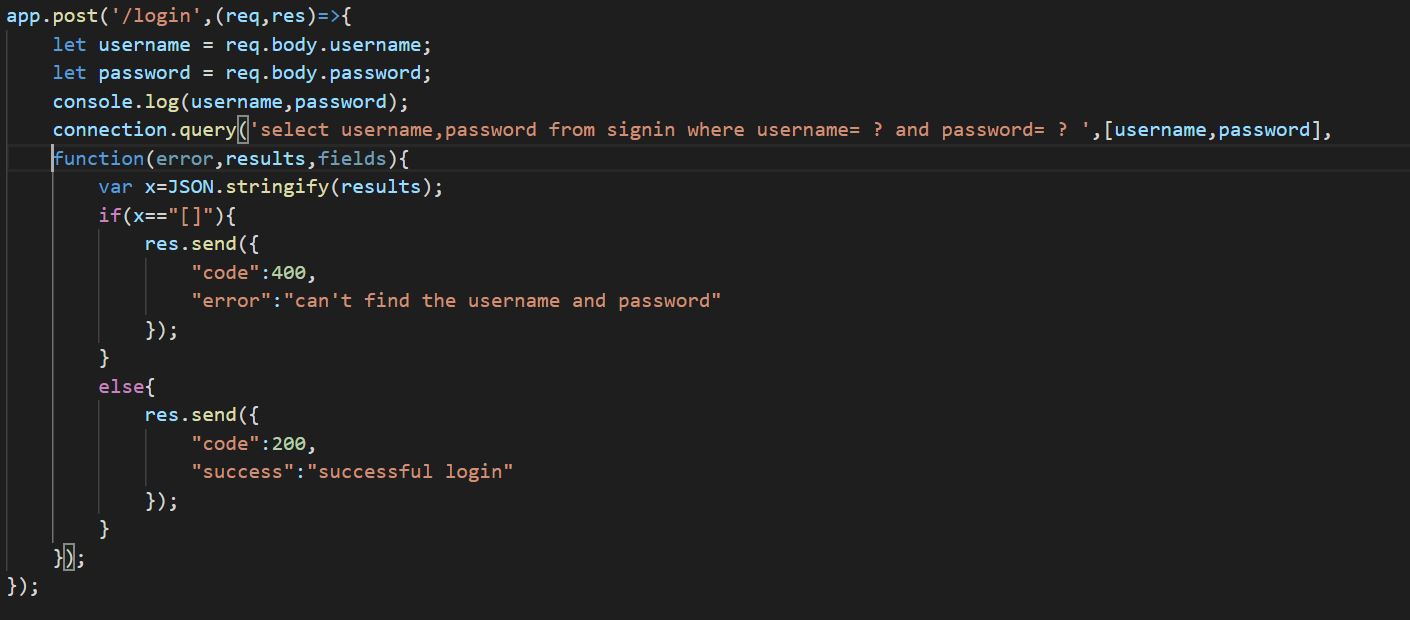


**Creating Database in the MYSQL Workbench**



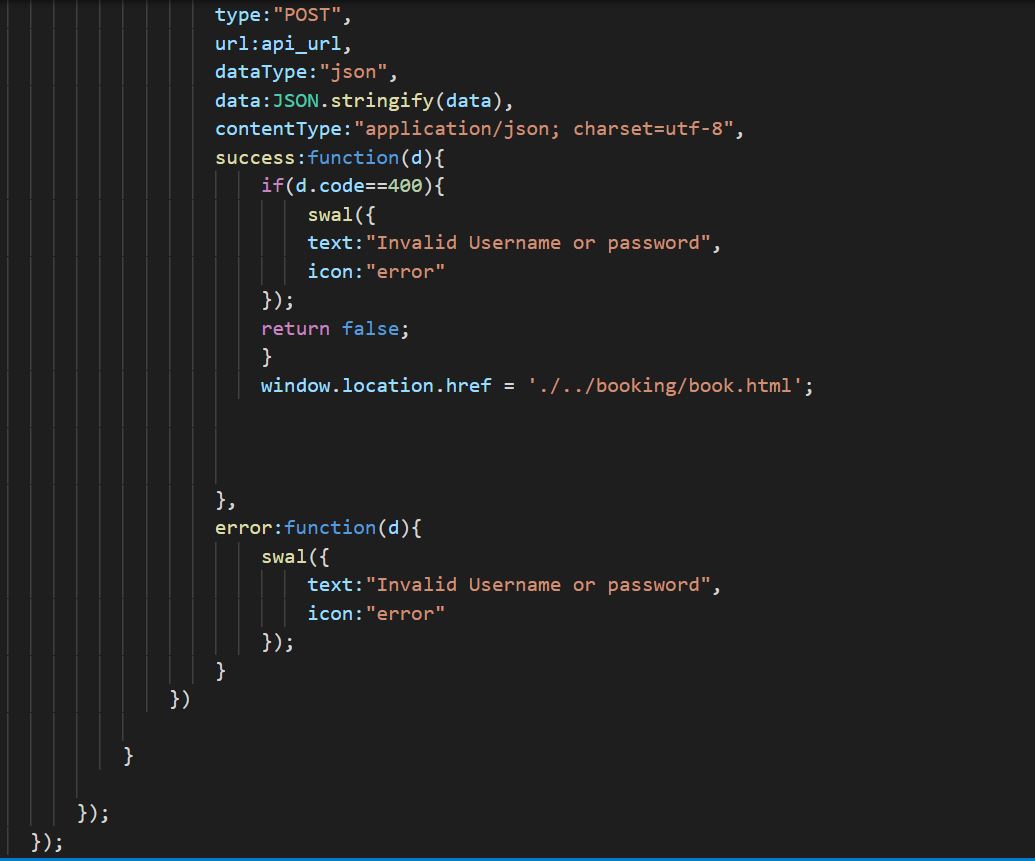
**API Code, Validations & Outputs of Data in the Database**

**9.1 Sign in API Code**

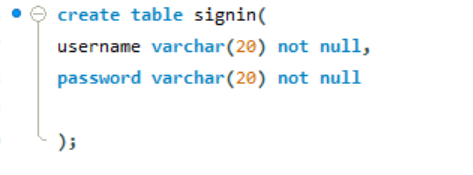


**Validations and Ajax for sign in**

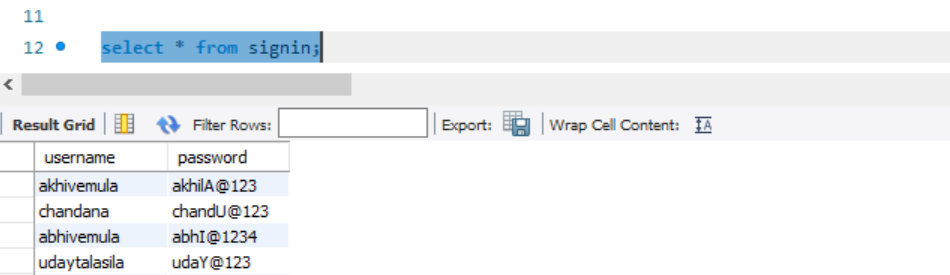




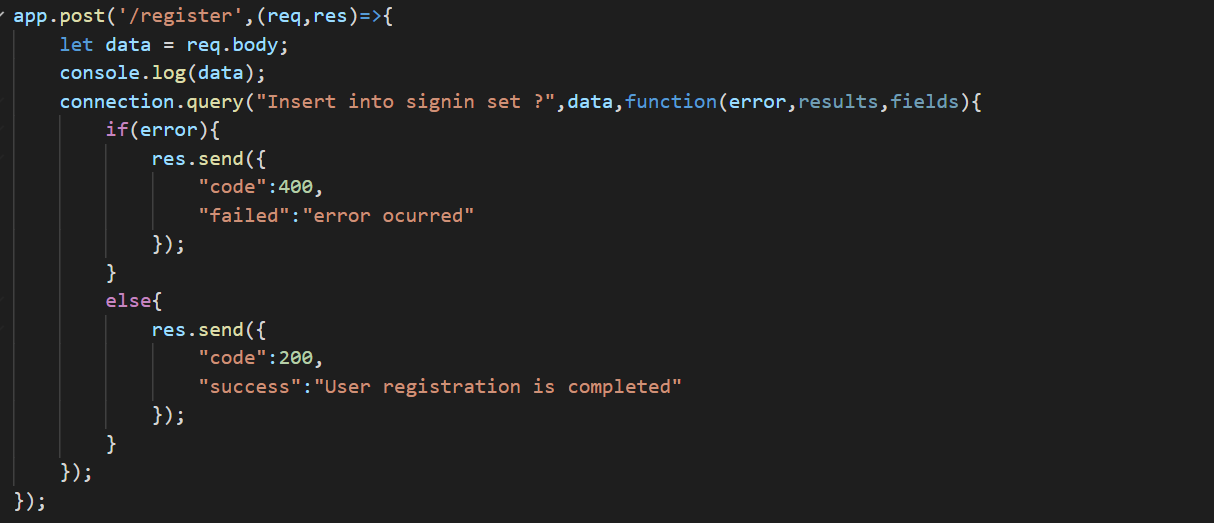
**Creating the table of sign in**



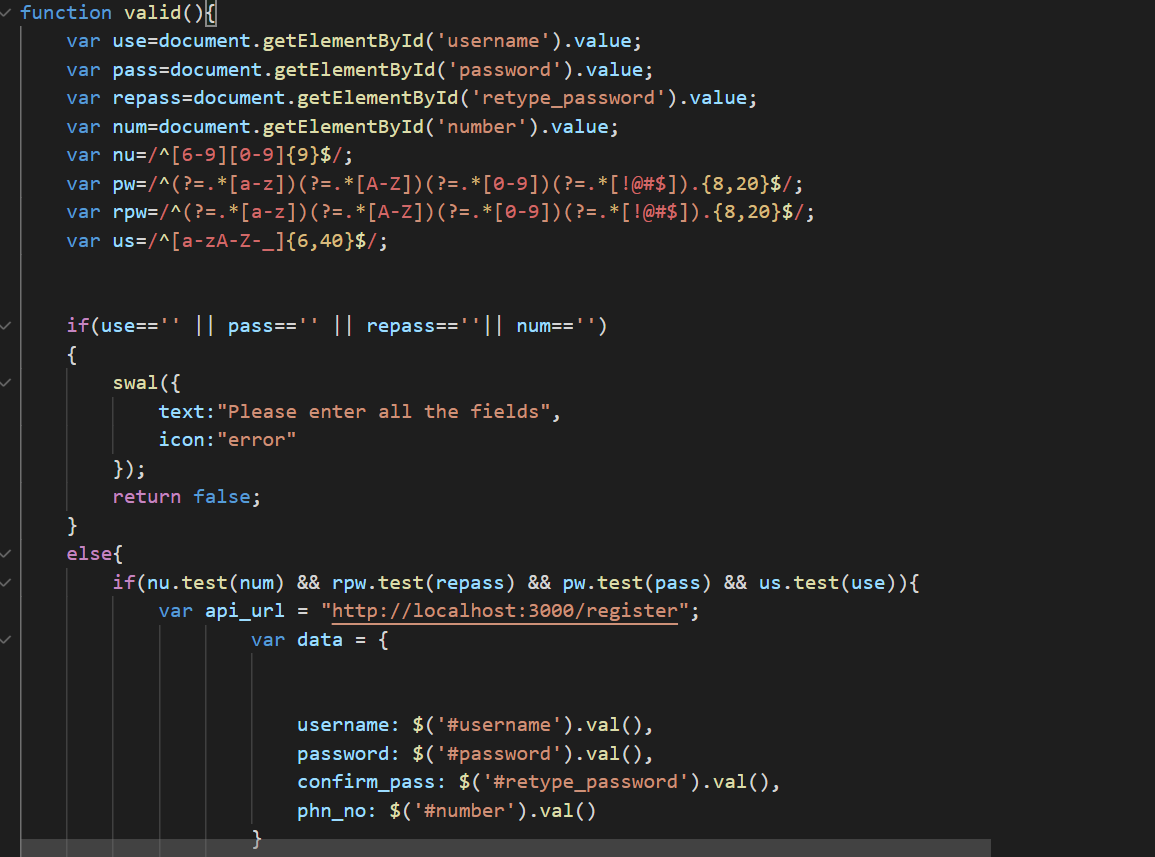
**Output of data in Database**



**9.2 Sign up API Code**



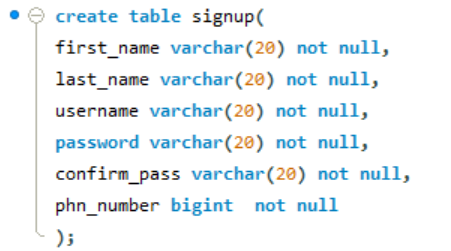
**Ajax and Validations**



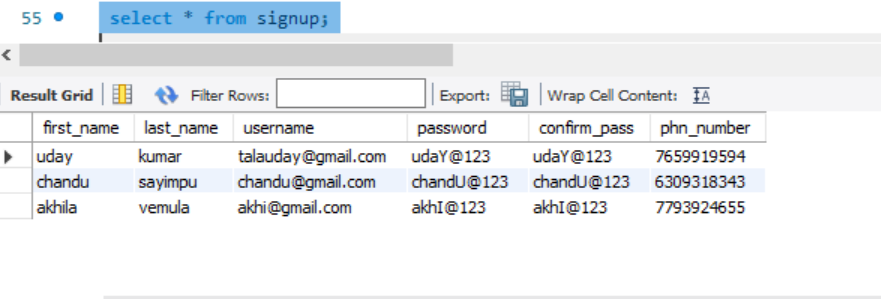




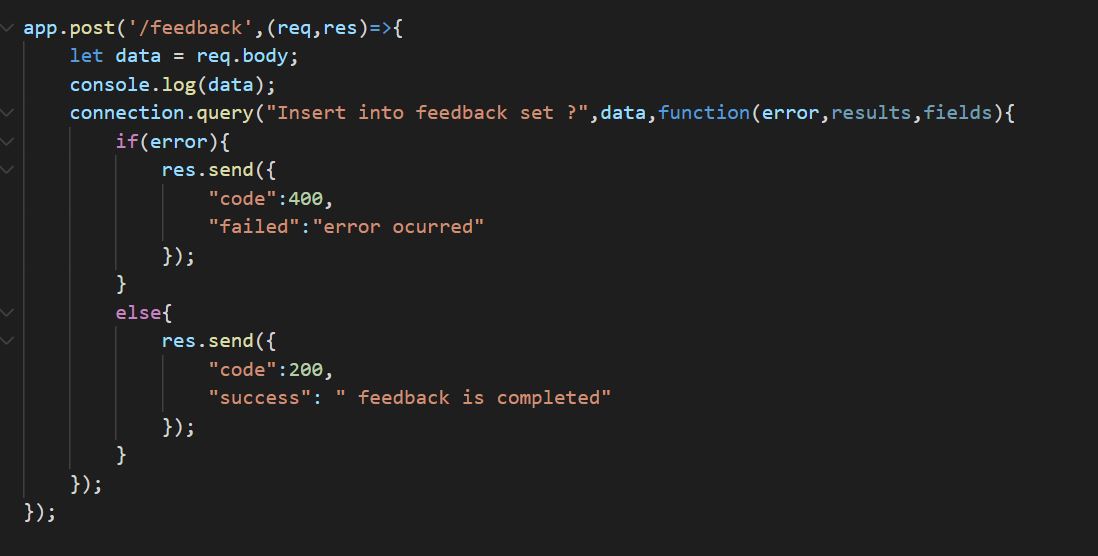
**Creating table of signup**



**Output of data in Database**

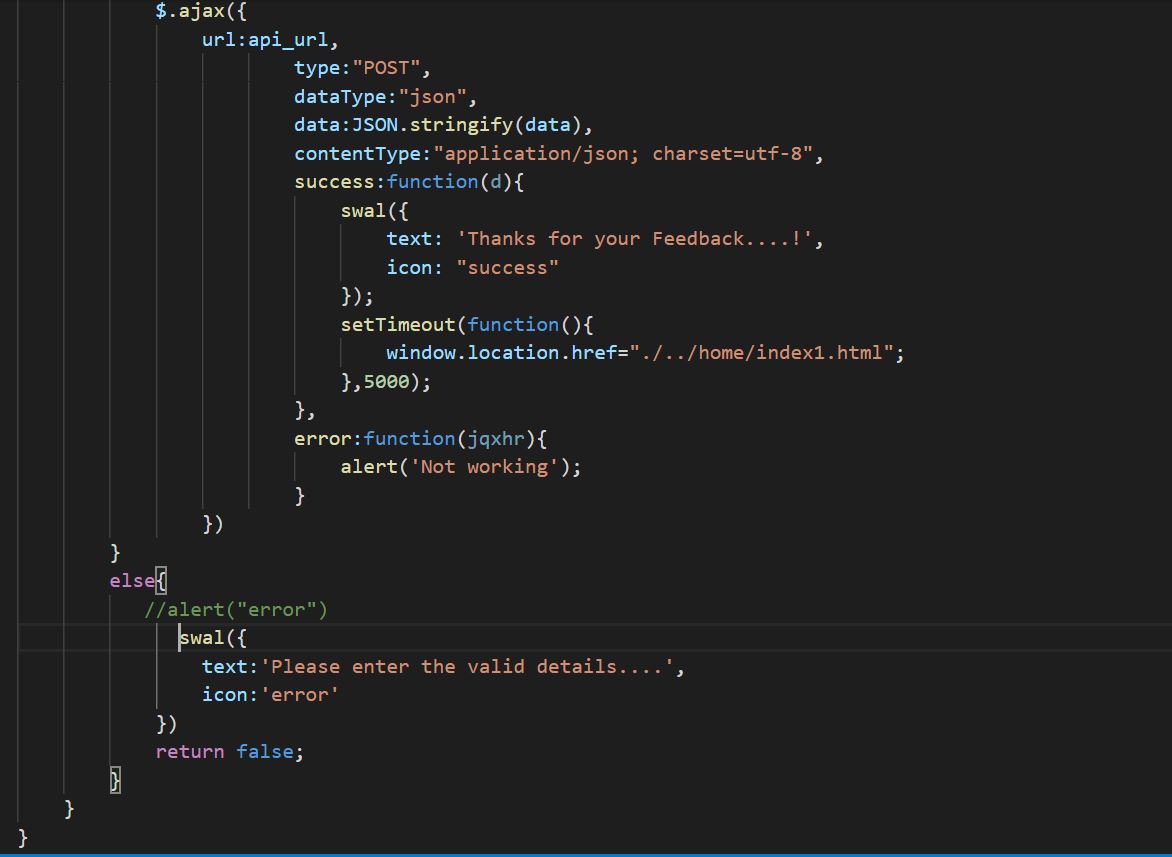


**9.3 API for Feedback**

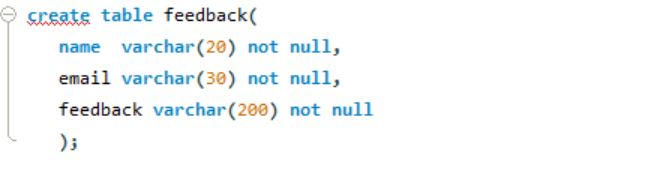


**Validation and Ajax**

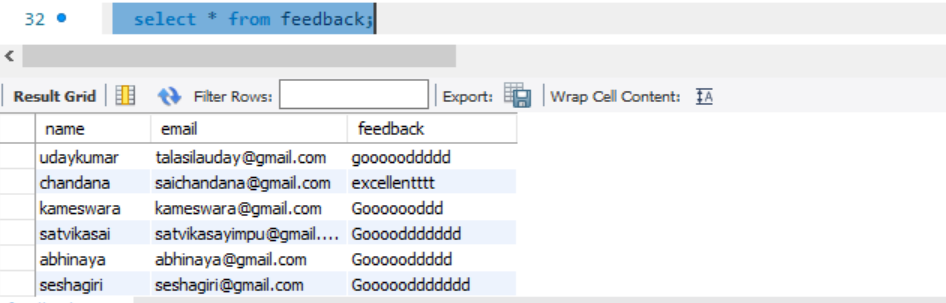




**Creating table for feedback**



**Output of data Database**



**CHAPTER 10**

**MAINTENANCE**

The maintenance phase involves making changes to hardware, software, and documentation to support its operational effectiveness. It includes making changes to improve a system’s performance, correct problems, enhance security, or address user requirements. To ensure modifications do not disrupt operations or degrade a system’s performance or security, organizations should establish appropriate change management standards and procedures.

Routine changes are not as complex as major modifications and can usually be implemented in the normal course of business. Routine change controls should include procedures for requesting evaluating, approving, testing, installing, and documenting website modifications.

Maintaining accurate, up-to-date hardware and software inventories is a critical part of all change management processes. Management should carefully document all modifications to ensure accurate system inventories. Management should coordinate all technology related changes through an oversight committee and assign an appropriate party responsibility for administering software patch management programs. Quality assurance, security, audit, regulatory compliance, network, and end-user personnel should be appropriately included in change management processes. Risk and security review should be done whenever a system modification is implemented to ensure controls remain in place. For maintenance of the website:

1. The database has to be updated regularly according to new available information.

2. Redundant and false information must be removed from the database.

3. Newer version of PHP and MYSQL can be used for up gradation of website and to improve the overall performance of the system.

**CHAPTER 11**

**FUTURE SCOPE AND FUTURE ENHANCEMENT**

The future of the RFID parking system is expected to be significantly influenced by the arrival of automated vehicles (AVs). Several cities around the world are already beginning to trial self-parking vehicles, specialized AV parking lots, and robotic parking valets. It helps to resolve the growing problem of traffic congestion. As for the future work the users can book a parking space from a remote location. GPS, reservation facilities and license plate scanner can be included in the future.

This concept can be applied to the techno-commercial application of the car park. In the future, we will consider the case with 200 car parks. In the experimentation, the 200 cars parks will be divided into 4 sections each of 50 lots respectively. For each five lots, we will require one Arduino, ten sensors, and one XBee ZigBee (Series 2) module, reaching to 40 Arduino, 400 sensors and 40 XBee ZigBee modules for the whole experiment. These will be coordinated with server Zigbee module to allow the administrator to see the availability of lots on display. This work can be extended to autonomous car parks where the display will be used to detect the availability of parking lots as well as accepting different payment methods. The system may be linked to smart phones through mobile apps to enable clients to reserve parking lots using their mobile phones. The system could also be connected to GPS systems to allow clients to search for empty parking lots in different parking areas remotely while driving thus saving them time.

For example, in Boulder, Colorado, Park Plus is working on deploying a fully automated parking garage in the Western United States through Boulder’s Pearl West mixed-use development. The company’s automated parking system uses lasers to scan cars and a robotic valet to park the vehicles. Vehicles are transported by a robotic dolly that lifts and transfers them to storage racks. Using this system, up to 4 times as many cars can be parked in the same amount of space as a traditional garage (since there is no need for extra space in between cars). The automated system is expected to deliver vehicles within 3-5 minutes of a retrieval request.

**CHAPTER 12**

**CONCLUSION**

The concepts of smart cities have always been a dream. There have been advancements made from the past couple of years to make smart city dream to reality. The advancement of internet of things and cloud technologies has given rise to the new possibilities in terms of smart cities. RFID parking facilities have always been the core of constructing smart cities. The system provides a real time process and information of the parking slots. This paper enhances the performance of saving users time to locate an appropriate parking space. It helps to resolve the growing problem of traffic congestion.

It is apparent that the demand for the smart car parking system will continue to increase in the upcoming years. Though the smart parking system already exists, our project is aimed at making the system more cost effective and user friendly thus increasing its adoption in the market. The project was successful & cost effective, user-friendly and had 90% accuracy. Future works will extend the system to administer 200 parking lots and incorporate other different technologies such as interlink with smart phones and GPS system to increase its dependability.

In this study, the various types of RFID parking system and has been presented. From the various examples of the implementation of the RFID parking system being presented, its efficiency in alleviating the traffic problem that arises especially in the city area where traffic congestion and the insufficient parking spaces are undeniable. It does so by directing patrons and optimizing the use of parking spaces.

With the study on all the sensor technologies used in detecting vehicles, which are one of the most crucial parts of the smart parking system, the pros and cons of each sensor technologies can be analyzed. Although, there are certain disadvantages in the implementation of visual based system in vehicle detection as described earlier, the advantages far outweigh its disadvantages.

Developments in RFID technology continue to yield larger memory capacities, wider reading ranges, and faster processing. The interest in RFID as a solution to optimize further the automation and tracking of documents are gathering momentum at an increasing pace, with more parking joining the trails. RFID is increasing in popularity among parking, as the early adopters of this technology have shown that, it makes good economic sense, both for large and small parking.

**REFERENCES**

-

* <https://www.electronicshub.org/rfid-based-car-parking-system/>
* <https://www.researchgate.net/publication/327575252_Radio_Frequency_Identification_RFID_Based_Car_Parking_System>
* <https://www.ripublication.com/irph/ijict_spl/ijictv4n4spl_05.pdf>
* <https://www.edgefxkits.com/blog/rfid-based-paid-car-parking/>
* <https://www.electronicsforu.com/electronics-projects/hardware-diy/rfid-based-automatic-vehicle-parking-system>