**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“Jnana Sangama”, Belagavi-590018**



**A**

**DBMS Laboratory Mini Project Report**

**On**

**“Project Title”**

**Submitted in partial fulfillment for 5th SEMESTER**

**BACHELOR OF ENGINEERING**

**IN**

**INFORMATION SCIENCE AND ENGINEERING**

**Submitted by**

**Student Name (USN)**

**Under the Guidance of:**

**Internal Guide Name**

**Designation,**

**Dept. of ISE, SJBIT,**



**Department of information science and engineering**

**SJB Institute of Technology**

**BGS Health and Education city,**

**Kengeri, Bengaluru-560060, Karnataka, India**.

**2021 - 2022**

**|| Jai Sri Gurudev ||**

**Sri AdichunchanagiriShikshana Trust ®**

**SJB INSTITUTE OF TECHNOLOGY**

## BGS Health & Education City, Kengeri, Bengaluru – 560 060

**Department of Information Science & Engineering**



**CERTIFICATE**

Certified that the Mini project work entitled “**AROGYA SEVA**” carried out by Mr./Ms. **NAMES**Prajwal Bharadwaj BM bearing USN **1JB19IS061** is a bonafide student of **SJB Institute of Technology** in partial fulfilment for 5th Semester DBMS Mini Project with Laboratory in **INFORMATION SCIENCE AND ENGINEERING** of the **Visvesvaraya Technological University**, **Belagavi** during the academic year **2021-22.** It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for the said degree.

**Name of the Guide Dr. MOHAN H.S.**

**Designation Professor & Head**

**Dept. of ISE, SJBIT Dept. of ISE, SJBIT**

1. **Internal Examiner**
2. **External Examiner**



**ACKNOWLEDGEMENT**

****

We would like to express our profound grateful to His Divine Soul **Padmabhushan Sri SriSri Dr. BalagangadharanathaMaha Swamiji** and His Holiness **Jagadguru Sri SriSri Dr. NirmalanandanathaMahaSwamiji** for providing us an opportunity to complete our academics in this esteemed institution.

We would also like to express our profound thanks to **Revered Sri Sri Dr. PrakashnathSwamiji**, Managing Director, SJB Institute of Technology, for his continuous support in providing amenities to carry out this project in this admired institution.

We express our gratitude to **Dr.Ajai Chandran**, **Principal**, SJB Institute of Technology, for providing us an excellent facilities and academic ambience; which have helped us in satisfactory completion of project work.

We extend our sincere thanks to **Dr. Mohan H S**, Professor &Head, Department of Information Science and Engineering; for providing us an invaluable support throughout the period of our project work.

We wish to express our heartfelt gratitude to our **guide, Mr. Santhrupth**, **Assistant Professor,** Department of Information Science and Engineering for his / her valuable guidance, suggestions and cheerful encouragement during the entire period of our project work.

We express our truthful thanks to Mini Project Co-ordinator**, Mr. Chetan R**, **Assistant Professor,**Department of Information Science and Engineering for his valuable support.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents, Teaching &Non-teaching staffs of the department, the library staff and all our friends, who have directly or indirectly supported us during the period of our project work.

Regards,

**Prajwal Bharadwaj B M**

**(1JB19IS061)**

**ABSTRACT**

The purpose of the project entitled as “AROGYA SEVA” is to computerize the Front Office Management of Hospital to develop software, which is user friendly simple, fast, and cost – effective. It deals with the collection of patient’s information, diagnosis details, etc. Traditionally, it was done manually. The main function of the system is register and store patient details and doctor details and retrieve these details as and when required, and to manipulate these details meaningfully System input contains patient details, diagnosis details, while system output is to get these details on to the screen. The Hospital Management System can be entered using a username and password.

This system stores data and enables functionality that organizes and maintains the patient details and appointment with doctors. It is accessibleeither by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast. This project is insight into the design and implementation of Hospital management system. The primary aim of thus is to improve accuracy and enhance safety and efficiency in the hospital. Today management is one of the key features of all form.

Management provides sophistication to perform any kind of task in a particular form.

This program can be used in any Hospitals having a database to maintain. It is a computer-based system which helps hospital management to improve appointments, more slots, and efficient allotment etc. The software used can generate reports, as per the riser’s requirements. Using this hospital management system user is also able to book vaccine slot. The system allows the user to entertheir comorbidities, which in turn books an appointment for the specific doctor. It can also maintain the record of bookingby the patient. The system will also give report showing the track of bookingrecord for that userID. The system services and goals are established by consultation with system user. It also involves manual entry upon new patient with same userID. Arogya seva is being built on robust, integrated technology. Thismanagement system is user friendly

**INDEX**

1. INTRODUCTION

1.1 Introduction to DBMS

1.2 Introduction to SQL

1.3 Introduction to Hospital management System

2. REQUIREMENTS SPECIFICATION

2.1 Introduction

2.2 Hardware requirements

2.3 Software requirements

3. SOFTWARE SPECIFICATION

3.1 HTML

3.2 CASCADING STYLE SHEETS (CSS):

3.3 JAVASCRIPT

3.4 Why use Python?

3.5 Tools

3.6 Connectivity

4. DESIGN

4.1 Entity Relation Diagram

4.2 Relational Schema Diagram

5. SYSTEM IMPLEMENTATION

5.1 Implementation

5.2 Table creation

5.3 Insertion of values into tables

6. SNAPSHOTS

6.1

6.2

7. CONCLUSION

8. BIBLOGRAPHY

CHAPTER 1

**INTRODUCTION**

**1.1 Introduction to DBMS**

DBMS = database + management system. Database is a collection of data and management system is a set of programsto store and retrieve those data. DBMS is a collection of inter-related data and set of programs to store & access thosedata in an easy and effective manner.Database management system (DBMS) is a software for storing and retrieving users' data while consideringappropriate security measures. It consists of a group of programs which manipulate the database. The DBMS acceptsthe request for data from an application and instructs the operating system to provide the· Specific data. In large

systems, a DBMS helps users and other third-party software to store and retrieve data. DBMS allows users to createtheir own databases as per their requirement. The term "DBMS" includes the user of the database and other applicationprograms. It provides an interface between the data and the software application.DBMS provides security and removes redundancy DBMS has many advantages over tradition flat file management

system. End-users, application programmers and database administrators are the type of users who access a DBMS.DBMS is widely used in banking, airlines, telecommunication, finance and other industries.

Four types of DBMS systems are

I) hierarchical

2) network

3) relational

4) object-oriented DBMS.

DBMS serves as an efficient handler to balance the needs of multiple applications using the same data. Cost ofhardware and software of a DBMS is quite high which increases the budget of your organization.

**1.2 Introduction to SQL**

SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is thestandard language for relational database management systems . SQL statements are used to perform tasks such asupdate data on a database, or retrieve data from a database. Some common relational database management systemsthat use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc.Although most database systems use SQL , most of them also have their own additional proprietary extensions that areusually only used on their system. However, the standard SQL commands such as "Select", "Insert ", "Update","Delete" , "Create", an d "Drop" can be used to accomplish almost everything that one needs to do with a database .MySQL is the most popular Open-Source Relational SQL Database Management System. MySQL is one of the bestRDBMS being used for developing various web-based software applications. MySQL is developed, marketed andsupported by MySQL AB , which is a Swedish company .A database is a separate application that stores a collection of data . Each database has one or more distinct API’s forcreating, accessing, managing, searching and replicating the data it holds . Other kinds of data stores can also be used,such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast andeasy with those types of systems . Nowadays, we use relational database management systems (RDBMS) to store andmanage huge volume of data. MySQL is released under an open-source license .So , you have nothing to pay to use it.MySQL is a very powerful program in its own right. MySQL works very quickly and works well even with large datasets.

**Indicative areas for the use of a DBMS**

• Airlines : reservations , schedules etc .

• Telecom : calls made, customer details, network usage etc.

• Universities: registration, results, grades etc.

• Sales: products, purchases, customers etc.

• Banking: all transactions etc.

**1.3 Introduction:**

The project Hospital Management system includes registration of patients, storing their details into the system. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. It includes a search facility to know the current status of each room. User can search availability of a doctor and the details of a patient using the id.

The Hospital Management System can be entered using a username and password. It is accessible either by an administrator or receptionist or a patient. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast.

Hospital Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals.

Hospital Management System is designed for multispecialty hospitals, to cover a wide range of hospital administration and management processes. It is an integrated end-to-end Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow.

Hospital Management System is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the hospital helps you manage your processes

CHAPTER 2

**REQUIREMENTSPECIFICATION**

**2.1 INTRODUCTION:**

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as(computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

**2.2 HARDWARE REQUIREMENTS:**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

HARDWARE REQUIREMENTS FOR PRESENT PROJECT:

PROCESSOR : Intel dual Core ,i3

RAM : 1 GB

HARD DISK : 80 GB

**2.3 SOFTWARE REQUIREMENTS:**

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:

OPERATING SYSTEM : Windows 7/ XP/8/10/11

FRONT END : Html,css,javascript

SERVER SIDE SCRIPT :Flask

DATABASE :mysql

CONNECTIVTY :Xampp ,sqlaclchemy

**CHAPTER 3**

**SOFTWARE SPECIFICATION**

**SOFTWARE SPECIFICATION**

**3.1 HTML:**

**HTML** or **Hypertext Markup Language** is the standard [markup language](http://en.wikipedia.org/wiki/Markup_language) used to create [web pages](http://en.wikipedia.org/wiki/Web_page).

HTML is written in the form of [HTML elements](http://en.wikipedia.org/wiki/HTML_element) consisting of *tags* enclosed in [angle brackets](http://en.wikipedia.org/wiki/Angle_brackets) (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent *empty elements* and so are unpaired, for example <img>. The first tag in a pair is the *start tag*, and the second tag is the *end tag* (they are also called *opening tags* and *closing tags*). Though not always necessary, it is best practice to append a slash to tags which are not paired with a closing tag.

The purpose of a [web browser](http://en.wikipedia.org/wiki/Web_browser) is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML describes the structure of a website [semantically](http://en.wikipedia.org/wiki/Semantic) along with cues for presentation, making it a [markup language](http://en.wikipedia.org/wiki/Markup_language) rather than a [programming language](http://en.wikipedia.org/wiki/Programming_language).

HTML elements form the building blocks of all [websites](http://en.wikipedia.org/wiki/Website). HTML allows [images and objects](http://en.wikipedia.org/wiki/Img_(HTML_element)) to be embedded and can be used to create [interactive forms](http://en.wikipedia.org/wiki/Fieldset). It provides a means to create [structured documents](http://en.wikipedia.org/wiki/Structured_document) by denoting structural [semantics](http://en.wikipedia.org/wiki/Semantic) for text such as headings, paragraphs, lists, [links](http://en.wikipedia.org/wiki/Hyperlink), quotes and other items. It can embed [scripts](http://en.wikipedia.org/wiki/Scripting_language) written in languages such as [JavaScript](http://en.wikipedia.org/wiki/JavaScript) which affect the behavior of HTML web pages.

**3.2 CASCADING STYLE SHEETS** (**CSS**):

It is a [style sheet language](http://en.wikipedia.org/wiki/Style_sheet_language) used for describing the [look and formatting](http://en.wikipedia.org/wiki/Presentation_semantics) of a document written in a [markup language](http://en.wikipedia.org/wiki/Markup_language). While most often used to style [web pages](http://en.wikipedia.org/wiki/Web_page) and [interfaces](http://en.wikipedia.org/wiki/Interface_(computing)) written in [HTML](http://en.wikipedia.org/wiki/HTML) and [XHTML](http://en.wikipedia.org/wiki/XHTML), the language can be applied to any kind of [XML](http://en.wikipedia.org/wiki/XML) document, including [plain XML](http://en.wikipedia.org/wiki/Plain_Old_XML), [SVG](http://en.wikipedia.org/wiki/Scalable_Vector_Graphics) and [XUL](http://en.wikipedia.org/wiki/XUL). CSS is a cornerstone specification of [the web](http://en.wikipedia.org/wiki/The_web) and almost all web pages use CSS style sheets to describe their presentation.

CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the[layout](http://en.wikipedia.org/wiki/Page_layout), [colors](http://en.wikipedia.org/wiki/Color), and [fonts](http://en.wikipedia.org/wiki/Typeface).[[1]](http://en.wikipedia.org/wiki/Cascading_Style_Sheets#cite_note-1) This separation can improve content [accessibility](http://en.wikipedia.org/wiki/Accessibility), provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content .

CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or [screen reader](http://en.wikipedia.org/wiki/Screen_reader)) and on [Braille-based](http://en.wikipedia.org/wiki/Braille_display), tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified. However if the author or the reader did not link the document to a specific style sheet the default style of the browser will be applied.

**3.3 JAVASCRIPT:**

JavaScript is the scripting language of the Web. All modern HTML pages are using JavaScript. A scripting language is a lightweight programming language.JavaScript code can be inserted into any HTML page, and it can be executed by all types of web browsers. JavaScript is easy to learn.

**WHY TO USE JAVASCRIPT**:

JavaScript is one of the 3 languages all web developers  must learn:

1. HTML to define the content of web pages
2. CSS to specify the layout of web pages
3. JavaScript to specify the behavior of web pages

Example

x = document.getElementById("demo");  //Find the HTML element with id="demo"  
x.innerHTML = "Hello JavaScript";     //Change the content of the HTML element

**document.getElementById()** is one of the most commonly used HTML DOM methods.

**OTHER USES OF JAVASCRIPT**:

* Delete HTML elements
* Create new HTML elements
* Copy HTML elements
* In HTML, JavaScript is a sequence of statements that can be executed by the web browser.

**JAVASCRIPT STATEMENTS**:

* JavaScript statements are "commands" to the browser.
* The purpose of the statements is to tell the browser what to do.
* This JavaScript statement tells the browser to write "Hello Dolly" inside an HTML element with id="demo":

Semicolon;

* Semicolon separates JavaScript statements.
* Normally you add a semicolon at the end of each executable statement.
* Using semicolons also makes it possible to write many statements on one line.

**JAVASCRIPT CODE:**

* JavaScript code (or just JavaScript) is a sequence of JavaScript statements.
* Each statement is executed by the browser in the sequence they are written.
* This example will manipulate two HTML elements:
* Example
* document.getElementById("demo").innerHTML="Hello Dolly";  
  document.getElementById("myDIV").innerHTML="How are you?";

## JAVASCRIPT PROPERTIES:

* Properties are the values associated with a JavaScript object.
* A JavaScript object is a collection of unordered properties.
* Properties can usually be changed, added, and deleted, but some are read only.

**3.4 Why use Python?**

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library

**3.5 Tools**

**Python Flask**

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.[2] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

**SQLAlchemy**

The Python SQL Toolkit and Object Relational Mapper

SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL.

It provides a full suite of well known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.

**Flask-SQLALchemy**

Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy to your application. It aims to simplify using SQLAlchemy with Flask by providing useful defaults and extra helpers that make it easier to accomplish common tasks.

Installing

Install and update using pip:

$ pip install -U Flask-SQLAlchemy

**Werkzeug**

Werkzeug (German for "tool") is a utility library for the Python programming language, in other words a toolkit for Web Server Gateway Interface (WSGI) applications, and is licensed under a BSD License. Werkzeug can realize software objects for request, response, and utility functions. It can be used to build a custom software framework on top of it and supports Python 2.7 and 3.5 and later

**Jinja**

Jinja, also by Ronacher, is a template engine for the Python programming language and is licensed under a BSD License. Similar to the Django web framework, it handles templates in a sandbox

**3.6 Connectivity**

**Xampp**

Xampp is a free and open-source cross-platform web server solution stack package developed by Apache Friends,consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.The term XAMPP is an apparent acronym. However, there is no official acronym expansion specified on the Apache Friends website. Their homepage header reads "XAMPP Apache + MariaDB + PHP + Perl", indicating that this abbreviation is a recursive acronym.

**CHAPTER 4**

**DESIGN**

**Entity Relation-Diagram**

Diagram

Description automatically generated

**Schema Diagram**

Chart, box and whisker chart

Description automatically generated

**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1IMPLEMENTATION:**

Inroduction:

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it’s constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods

**5.2 Commands to create tables**

-- --------------------------------------------------------

**-- Table structure for table `doctor`**

CREATE TABLE `doctor` (

`doctorid` int(11) NOT NULL,

`email` varchar(40) NOT NULL,

`dname` varchar(40) NOT NULL,

`dlname` varchar(40) NOT NULL,

`doctor\_gender` varchar(40) NOT NULL,

`ddept` varchar(40) NOT NULL,

`dphno` varchar(40) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**-- Table structure for table `patients`**

CREATE TABLE `patients` (

`pid` int(11) NOT NULL,

`email` varchar(50) NOT NULL,

`fname` varchar(50) NOT NULL,

`gender` varchar(50) NOT NULL,

`slot` varchar(50) NOT NULL,

`disease` varchar(50) NOT NULL,

`time` time(6) NOT NULL,

`date` date NOT NULL,

`dept` varchar(50) NOT NULL,

`phno` varchar(50) NOT NULL,

`lname` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**-- Table structure for table `test`**

CREATE TABLE `test` (

`id` int(11) NOT NULL,

`fname` varchar(20) NOT NULL,

`lname` varchar(20) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**-- Table structure for table `triger`**

CREATE TABLE `triger` (

`trigerID` int(11) NOT NULL,

`pid` int(11) NOT NULL,

`email` varchar(40) NOT NULL,

`fname` varchar(40) NOT NULL,

`dept` varchar(40) NOT NULL,

`action` varchar(40) NOT NULL,

`timestamp` datetime NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**-- Table structure for table `user`**

CREATE TABLE `user` (

`id` int(11) NOT NULL,

`username` varchar(30) NOT NULL,

`email` varchar(30) NOT NULL,

`password` varchar(1000) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**-- Table structure for table `vaccine`**

CREATE TABLE `vaccine` (

`vid` int(11) NOT NULL,

`email` varchar(40) NOT NULL,

`fname` varchar(40) NOT NULL,

`lname` varchar(40) NOT NULL,

`gender` varchar(40) NOT NULL,

`vaccine` varchar(40) NOT NULL,

`dose` varchar(40) NOT NULL,

`age` varchar(40) NOT NULL,

`slot` varchar(40) NOT NULL,

`time` time(6) NOT NULL,

`date` date NOT NULL,

`phno` varchar(40) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

**--Alter table Statements**

ALTER TABLE `vaccine`

ADD CONSTRAINT `vaccine\_ibfk\_1` FOREIGN KEY (`email`) REFERENCES `user` (`email`) ON DELETE NO ACTION ON UPDATE NO ACTION;

COMMIT;

ALTER TABLE `triger`

ADD CONSTRAINT `triger\_ibfk\_1` FOREIGN KEY (`email`) REFERENCES `user` (`email`);

ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER TABLE `patients`

ADD CONSTRAINT `patients\_ibfk\_1` FOREIGN KEY (`email`) REFERENCES `user` (`email`) ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER TABLE `doctor`

ADD CONSTRAINT `doctor\_ibfk\_1` FOREIGN KEY (`email`) REFERENCES `user` (`email`) ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER TABLE `vaccine`

MODIFY `vid` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=8;

ALTER TABLE `user`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=10;

ALTER TABLE `triger`

MODIFY `trigerID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=13;

ALTER TABLE `test`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `patients`

MODIFY `pid` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=15; ALTER TABLE `doctor`

MODIFY `doctorid` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;

ALTER TABLE `vaccine`

ADD PRIMARY KEY (`vid`),

ADD KEY `email` (`email`);

ALTER TABLE `user`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `email` (`email`);

ALTER TABLE `triger`

ADD PRIMARY KEY (`trigerID`),

ADD KEY `triger\_ibfk\_1` (`email`);

ALTER TABLE `test`

ADD PRIMARY KEY (`id`);

ALTER TABLE `patients`

ADD PRIMARY KEY (`pid`),

ADD KEY `email` (`email`);

ALTER TABLE `doctor`

ADD PRIMARY KEY (`doctorid`),

ADD UNIQUE KEY `email` (`email`);

**5.3 Inserting Values into table:**

**Patients**

INSERT INTO `patients` (`pid`, `email`, `fname`, `gender`, `slot`, `disease`, `time`, `date`, `dept`, `phno`, `lname`) VALUES (NULL, 'karan@172', 'Karan', 'Male', 'morning', 'Regular Heart Checkup', '11:00:00.000000', '2022-01-19', 'cardiologist', '1234569582', 'kanur')

INSERT INTO `patients` (`pid`, `email`, `fname`, `gender`, `slot`, `disease`, `time`, `date`, `dept`, `phno`, `lname`) VALUES (NULL, 'nehagh@1443', 'Neha', 'Female', 'evening', 'Skin Allergy', '06:00:00.000000', '2022-01-21', 'Dermatologists', '8795621435', 'Tappur')

INSERT INTO `patients` (`pid`, `email`, `fname`, `gender`, `slot`, `disease`, `time`, `date`, `dept`, `phno`, `lname`) VALUES (NULL, 'Siddharthg@1345', 'Sidharth', 'Male', 'night', 'Harmonal Imbalance', '08:30:00.000000', '2022-01-23', 'Endocrinologists', '5468217398', 'Gamun')

INSERT INTO `patients` (`pid`, `email`, `fname`, `gender`, `slot`, `disease`, `time`, `date`, `dept`, `phno`, `lname`) VALUES (NULL, 'Haridg@gmail.com', 'Hari', 'Male', 'morning', 'Surgery Checkup', '08:00:00.000000', '2022-01-25', 'Plastic Surgeon', '6258124598', 'Patel')

INSERT INTO `patients` (`pid`, `email`, `fname`, `gender`, `slot`, `disease`, `time`, `date`, `dept`, `phno`, `lname`) VALUES (NULL, 'Chandrudfg@mail.com', 'Chandru', 'Gender', 'evening', 'Skin Allergy', '05:06:00.000000', '2022-02-25', 'Dermatologists', '3214568524', 'Gowda')

**Doctor Table**

INSERT INTO `doctor` (`doctorid`, `email`, `dname`, `dlname`, `doctor\_gender`, `ddept`, `dphno`) VALUES (NULL, 'santhosh11@gmail.com', 'Santhosh ', 'Narayan', 'Male', 'Dermatologists', '2562147826')

INSERT INTO `doctor` (`doctorid`, `email`, `dname`, `dlname`, `doctor\_gender`, `ddept`, `dphno`) VALUES (NULL, 'Terrydubrow15@gmail.com', 'Terry', 'Dubrow', 'Male', 'Plastic Surgeon', '7842155981')

INSERT INTO `doctor` (`doctorid`, `email`, `dname`, `dlname`, `doctor\_gender`, `ddept`, `dphno`) VALUES (NULL, 'shraddhak25@gmail.com', 'Shraddha', 'Sharma', 'Female', 'Dermatologists', '2581473692')

INSERT INTO `doctor` (`doctorid`, `email`, `dname`, `dlname`, `doctor\_gender`, `ddept`, `dphno`) VALUES (NULL, 'sd@12', 'Kavya', 'Sharma', 'Female', 'Endocrinologists', '2136567895')

INSERT INTO `doctor` (`doctorid`, `email`, `dname`, `dlname`, `doctor\_gender`, `ddept`, `dphno`) VALUES (NULL, 'sd@13', 'Bhavya', 'Sharma', 'Female', 'General Physician', '2136567395')

**Triger Table**

INSERT INTO `triger` (`trigerID`, `pid`, `email`, `fname`, `dept`, `action`, `timestamp`) VALUES (NULL, '21', 'Chandrudfg@mail.com', 'Chandru', 'Dermatologists', 'PATIENT INSERTED', '2022-01-18 23:05:49')

INSERT INTO `triger` (`trigerID`, `pid`, `email`, `fname`, `dept`, `action`, `timestamp`) VALUES (NULL, '20', 'Haridg@gmail.com', 'Hari', 'Plastic Surgeon', 'PATIENT INSERTED', '2022-01-18 23:04:22')

INSERT INTO `triger` (`trigerID`, `pid`, `email`, `fname`, `dept`, `action`, `timestamp`) VALUES (NULL, '19', 'Siddharthg@1345', 'Sidharth', 'Endocrinologists', 'PATIENT INSERTED', '2022-01-18 22:55:40')

INSERT INTO `triger` (`trigerID`, `pid`, `email`, `fname`, `dept`, `action`, `timestamp`) VALUES (NULL, '18', 'nehagh@1443', 'Neha', 'Dermatologists', 'PATIENT INSERTED', '2022-01-18 22:53:39')

INSERT INTO `triger` (`trigerID`, `pid`, `email`, `fname`, `dept`, `action`, `timestamp`) VALUES (NULL, '17', 'karan@172', 'Karan', 'cardiologist', 'PATIENT INSERTED', '2022-01-18 22:51:53')

**Vaccine Table**

INSERT INTO `vaccine` (`vid`, `email`, `fname`, `lname`, `gender`, `vaccine`, `dose`, `age`, `slot`, `time`, `date`, `phno`) VALUES (NULL, 'KavyaSitar@182', 'Kavya', 'Sitar', 'Female', 'pfizer', 'dose2', '24', 'morning', '09:00:00.000000', '2022-02-28', '9685741325')

INSERT INTO `vaccine` (`vid`, `email`, `fname`, `lname`, `gender`, `vaccine`, `dose`, `age`, `slot`, `time`, `date`, `phno`) VALUES (NULL, 'Gowthami@gmail.com', 'Gowthami', 'jaitley', 'Female', 'covaxin', 'dose2', '19', 'morning', '11:00:00.000000', '2022-03-06', '9852361475')

INSERT INTO `vaccine` (`vid`, `email`, `fname`, `lname`, `gender`, `vaccine`, `dose`, `age`, `slot`, `time`, `date`, `phno`) VALUES (NULL, 'Vinayad@gmail.com', 'Vinay', 'K H', 'Male', 'pfizer', 'dose1', '29', 'morning', '08:00:00.000000', '2022-01-20', '9872563145')

INSERT INTO `vaccine` (`vid`, `email`, `fname`, `lname`, `gender`, `vaccine`, `dose`, `age`, `slot`, `time`, `date`, `phno`) VALUES (NULL, 'Arungf@gmail.com', 'Arun', 'Siya', 'Male', 'covishield', 'booster', '24', 'evening', '06:00:00.000000', '2022-02-07', '2583467925')

INSERT INTO `vaccine` (`vid`, `email`, `fname`, `lname`, `gender`, `vaccine`, `dose`, `age`, `slot`, `time`, `date`, `phno`) VALUES (NULL, 'aradya@gmail.com', 'Aradya', 'Aroha', 'Male', 'covaxin', 'dose2', '25', 'morning', '08:00:00.000000', '2022-01-20', '1234567892')

**Stored Procedure**

DELIMITER $$

CREATE DEFINER=`root`@`localhost` PROCEDURE `doctorroutine`()

SELECT \* from doctor

GROUP By email

order by ddept ASC$$

DELIMITER ;

**Triger Update**

CREATE TRIGGER `patientupdated` AFTER UPDATE ON `patients`

FOR EACH ROW INSERT INTO trigerVALUES(null,NEW.pid,NEW.email,NEW.fname,NEW.dept,'PATIENT UPDATED',NOW())

**Trigger Insert**

CREATE TRIGGER `patientinsertion` AFTER INSERT ON `patients`

FOR EACH ROW INSERT INTO trigerVALUES(null,NEW.pid,NEW.email,NEW.fname,NEW.dept,'PATIENT INSERTED',NOW())

**Trigger delete**

CREATE TRIGGER `patientdeleted` BEFORE DELETE ON `patients`

FOR EACH ROW INSERT INTO trigerVALUES(null,OLD.pid,OLD.email,OLD.fname,OLD.dept,'PATIENT DELETED',NOW())

**CHAPTER 6**

**RESULTS/SNAPSHOTS**

1.1 Home Page

A picture containing text, person, spectacles

Description automatically generated

1.2 Dashboard

A person wearing a mask

Description automatically generated with low confidence

1.3 Signup Page

A screenshot of a computer

Description automatically generated

1.4 Login Page

A screenshot of a computer

Description automatically generated

1.5 Doctor Detail

A screenshot of a computer

Description automatically generated with medium confidence

1.6 Patient Booking Details

A screenshot of a computer

Description automatically generated

1.7 Trigger

A screenshot of a computer

Description automatically generated with medium confidence

**Conclusion**

By using "Arogya Seva", patient can easily track and book his appointment. The orderwill be taken systematically and will be saved safely in the database rather than manually through paperwork. Itrequires less time and space to save the order details. Database can be cleared yearly or monthly according to the

company.

The system provides user to add, delete and update changes that has been made to the database. The system digitalizesthe aspect of keeping records.

**REFERENCES**

I. Database Systems Models, Languages, Design and Application programming, RamezElrnasri and

Shamkant B. Navathe, 7th Edition, Pearson.

2. Php Official Documentation - https://www.plus2net.com

3. W3Schools(HTML,CSS and PHP references) - https://www.3schools.com/tags/default.asp

4. Stack Overflow - https://www.stackoverflow.com/questions/tagged/javascript