CHAPTER 1

INTRODUCTION

1. 1 PROJECT ABSTRACT

Dental Clinic is a Graphical User Interface based website. The main objective of this website is to make it easy for the clinic to perform their daily tasks. The product will take care of Registrations, Appointments, and Dental services etc. It also gives an abstract idea about the clinic and its atmosphere.

Dental clinic is a website that manages registrations, appointment details, etc. It makes the job of the dentists and patients easy and efficient. A patient can make an appointment with the doctors of their choice for different services. Normally it was done through documentations, it requires a lot of time and man power by dentists and patients. This project implementing a real time system, through which patients can register his/her details, view appointments and communicate with corresponding coordinators.

The main users in this system are:

- Admin
- Patient

1.2 OBJECTIVE AND SCOPE

To manage clinic information and details for efficient working. This project provides efficient utilization of human resource. To reduce unnecessary paper work in the clinic and corresponding events. To reduce redundancy of data. The system provides a good management tool. It allows better viewing of information. The project aims to develop a system in order to provide solution to current problems and increase the efficiency.

The project Dental Clinic is divided into 2 main modules to make the development and maintenance easy.

- Admin Module
- Patient Module

ADMIN MODULE

The administrator can manage dentist, staff, clinic information, confirming an appointment, appointment dates, dental services. The main modules are:

1. Dentists
2. Patients
3. Staff
4. Dental service codes
5. Appointment Confirmations
6. Appointment dates
7. Clinic information

PATIENT MODULE

The user register for appointments, view clinic information and dentists, dental services. The main modules are:

1. Patient Registration
2. Appointments
3. Clinic information
4. Dental services
5. Dentists

1.3 PROBLEM STATEMENT

The admin and dentists have to handle the records of all the patients, appointments and the information is very difficult to retrieve and find particular information. These results are inconvenience and wastage of time. Hence there is a need to upgrade the system with a computer-based information system.

1.4 ORGANIZATION PROFILE

SAINTGITS COLLEGE OF APPLIED SCIENCES

Saintgits College of Applied Sciences was founded by a group of well-known academicians. They are pioneering educators, having unmatched experience in the field of education with a belief that the continuous pursuit of knowledge is the sole path to success. The primary focus of the institution is to expose the young minds to the world of technology instilling in them confidence and fortitude to face new challenges for enabling them to shine in their chosen fields. The college inculcates the development of all facts of the mind culminating in an intellectual and balanced personality. Our team of dedicated and caring faculty strives to widen the student′s horizon of learning thereby achieving excellent results for every student.

VISION

Holistic development of the youth to attain ″Learning of Life″.

MISSION

Impart quality higher education to equip and empower the youth with lifelong learning skills by including a spirit of enquiry, integrity and comparison.

QUALITY POLICY

We at Saintgits aim to provide quality education in career-oriented courses in Applied Sciences emphasizing on development of self-confident and skills, thus preparing the individual for a life-time of learning and professional growth. This is achieved through continual improvement in all the management processes of institution.

CHAPTER 2

SYSTEM CONFIGURATION

2.1 HARDWARE SPECIFICATION

Processor	:	Intel Core i3
RAM	:	4 GB DDR2
Hard Disk Drive	:	500 GB SATA
Monitor	:	Intel Original DG41RQ
Keyboard	:	Logitech USB Keyboard

2.2 SOFTWARE SPECIFICATION

Tool Used	:	PHP
Database Used	:	MYSQL Server
Operating system	:	Microsoft Windows 10

2.3 ABOUT THE DEVELOPMENT TOOLS

PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Ramus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page, it now stands for PHP: Hypertext Pre-processor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page. PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

HTML

HTML stands for Hypertext Mark-up Language, was invented by **Tim Burners Lee**. It is a simple text formatting language used to create hypertext documents. It is a platform independent language unlike most other programming languages. HTML is neutral and can be used on any platform or desktop. It is this feature of HTML that makes it popular as standard on the WWW. This versatile language allows the creation of hypertext links, also known as hyperlinks. The language used to develop web pages is called **H**yper **T**ext **M**ark-up **L**anguage (**HTML**). HTML is the language interpreted by a browser. HTML is specified as TAGS in an HTML document (i.e. the web page).

HTML TAGS

Tags are instructions that are embedded directly into the text of the document. An HTML tag is a signal to a browser that it should do something other than just throw text up on the screen. By convention all HTML tags begin with an open angle bracket (<) and end with a close angle bracket (>).

THE STRUCTURE OF AN HTML PROGRAM

Every HTML program has a rigid structure. The entire web page is enclosed within <html></html> tags. Within these tags two distinct sections are created using the <head></head> tags and the <body></body> tags.

JAVASCRIPT

JavaScript is an object based, cross-platform, loosely typed multiuse programming language that is used to add interactivity to the web pages. A JavaScript is a program that is included on an HTML page. Because it is enclosed in the <script> tag, the text of the script doesn't appear on the user's screen, and the Web browser knows to run the JavaScript program. The <script> tag is most often found within the <head> section of the HTML page. Scripts that write text to the screen or that write HTML is best put in the body section. JavaScript allows you create an active interface, giving the users feedback as they navigate your pages. JavaScript can be used to make sure that your users enter valid information in forms, which can save time and money. If the forms require calculations, you can do them in JavaScript the user's machine without needing to use a complex server CGI.

With JavaScript, you have the ability to create custom HTML pages depending on actions that the user takes. JavaScript controls the browser, because JavaScript has a set of date and time features. Java script deals with commands called event handles. An action by the user on the page triggers an event handler in your script. JavaScript is case sensitive. Scripts can be put in either of two places on an HTML pages: between the <head> and </head> tag or between the <body> and </body> tag.

One of the main uses of JavaScript is to provide feedback to people browsing your site. An alert window can be created that pops up and gives the user the vitally important information that they need to know about the page. Different languages versions can be have had on different scripts on one page. One script might be for any JavaScript version, another for JavaScript1.1 and higher, and a third for JavaScript1.2. In the case of JavaScript, the function is a set of JavaScript statements that performs a task. Function can be called as many times as needed.

DBMS DESCRIPTION

A database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. The general objective of database design is to make the database access easy, inexpensive and flexible to the user. Database design is used to define and then specify the structure of business used in the client/server system. A business object is nothing but information that is visible to the users of the system. The database must be normalized one. Database design is one of the important parts in developing software. It is a process of developing the conceptual model of data. It minimizes the artificiality embedded in using separate files. It is a definition of the entire information content of the organization and it specifies a relation between the data.

The primary objectives are fast response time to enquiries, more information at low cost, control of redundancy, clarity and ease-of-use and program independence, accuracy and integrity of the system, fast recovery, privacy and security of information and availability of powerful and user languages. For designing a table, the analyst must decide the fields of the tables, types of the fields, field length, default values etc. For this firstly the entity and relationship must be identified. Secondly, their attributes must be specified. This method of organizing the data table is known as normalization.

The data structure can be later redefined through a normalization process that groups data in the simplest way possible so that later changes can be made with ease. Normalization is designed to simplify relationship and establish logical links between files without losing information. An inherit problem is data redundancy and the inefficiency it generates. In other words, normalization implies splitting the tables into two or more tables with fewer columns. Most designing techniques try to reach and a few 4NF, but many reach 5NF.

The six normalization rules are:

- 1NF – each row or column must have a single value with no repeating values.
- 2NF – each non-key column must depend on the primary key column.
- 3NF – no non-key column can depend on another non-key column.
- BCNF – no attribute of a composite key depends on the attribute of another composite key.
- 4NF – an entity cannot have a 1:1 relation between key column and non-key column.
- 5NF –if and only if every non-trivial join dependency in it is implied by the candidate key. It is also known as project join normal form.

OPERATING SYSTEM

This project work is done in Windows 10, which is the operating system. An operating system is a set of software tools designed to make it easy for people or programmers to make optimum use of the computer. People can be separated into two groups, users and programmers. The user wants a convenient set of commands to manage files of data or programs, copy and run application packages while a programmer uses a set of tools that can be held together and debug programs. No matter where you are working, your computer will easier to use and manage, because Microsoft Windows 10 is more compatible and powerful than any workstation you have used.

The main features of Windows 10 are:

1. Easier to use

2. Easier to manage

3. More compatible

4. More powerful

1. EASIER TO USE

With Windows 10, you can have faster access to information, and you are able to accomplish tasks more quickly and easily.

Windows 10 makes it easier to:

- Work with files
- Find information.
- Personalize computing environment.
- Work remotely
- Work taking place the web

2. EASIER TO MANAGE

You and your network administrators can work more efficiently now, because many of the most common management tasks are streamlined with Windows 10.

With Windows 10 your workstation will be easier to:

- Setup
- Administrate
- Support

3. MORE COMPATIBLE

Windows 10 offers increased compatibility. With different types of network and with wide array of hardware and software.

Windows 10 also provides:

- Improved driver support
- Increased support for new generation hardware multimedia technologies.

4. **MORE POWERFUL**

For all your computing needs Windows 10 provides:

- Industrial-strength reliability.
- The highest level of security □ Powerful performance.

KERNEL FEATURES

The kernel is considered to be the heart of the operating system that provides services to the programs running on the computer. It takes care of the hardware, software, network resources, file systems and the remaining services such as

- Security
- System fault tolerance
- Multitasking
- Multiprocessing
- Platform independence
- File system reliability
- File system security
- Flexible protocol support
- Support multi-client operating system
- Enhanced scalability
- Multi-user environment
- Communication.

CHAPTER 3

SYSTEM ANALYSIS

3.1 PRELIMINARY INVESTIGATION

The first stage of any project, sometimes called the preliminary assessment is brief investigation of the system under consideration, system study and analysis deals with the study of the current system, this is the critical process of information development. It can be defined as problem solving which consist of four phases that can be successfully completed by applying appropriate skill and carefully addressing each dimension of the information system.

The purpose of preliminary study phase is to determine the initial feasibility of a project work. The product of the phase is a feasibility survey that is presented to a steering committee for a decision on whether the project should be developed.

After feasibility analysis, the next phase is the study of the current system. The purpose of this phase is to learn how the current system operates. The analyst identifies the problems, limitations and constraints forms preliminary solutions finally. The analyst updates the feasibility estimates and presents the findings as a problem statement for final study of phase reports.

The third phase of the system analysis is to define end-user requirements for a new system. The purpose of this phase is to identify what the new and improved information system must be able to do. The product of this phase is the requirement statement.

The fourth phase to select a feasible solution from alternatives that are evaluated in terms of operational, technical, and economic feasibility the analyst will recommend the best solution to the management for approval. A cost benefit analysis determines the expected system development lifetime, cost for a new system and the benefits of the new system.

3.2 EXISTING SYSTEM

The website is used to manage registrations, appointment details, dentists etc. It was done through documentations, it requires a lot of time and man power by the staff and patients.

3.2.1 LIMITATIONS OF EXISTING SYSTEM

- **Lack of immediate retrievals**: The information is very difficult to retrieve and find particular information like to find the expense details, the user has to go through various registers and find out manually. This results in inconvenience and wastage of time.

- **Lack of immediate information storage**: The information generated by
    - various transaction takes time and efforts to be stored at right place.

- **Error prone manual calculation**: Manual calculations are error one and take a
    - lot of time. This may result in incorrect information.

- **Preparation of accurate and prompt reports**: This becomes a difficult task
    - as information is prepared manually.

3.3 PROPOSED SYSTEM

The proposed system will handle all the difficulties of the existing systems. The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides proper security and reduces the manual work. The existing system has several disadvantages and many more difficulties to work well. The proposed system tries to eliminate or reduce these difficulties up to some extent. The proposed system will help the user to reduce the workload and provides user friendly environment so that he can easily do his jobs without time lagging.

All the activities are carried out through the software. The candidates can register for events and can communicate with the administrator. They can view all events and corresponding items through the website. The administrator can manage appointments, dentists, patients etc.

3.3.1 ADVANTAGES OF PROPOSED SYSTEM

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

1. BETTER SECURITY

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality.

Username and password requirement to sign in, ensures security. For employees they have unique user name and password. It will also provide data security as we are using the secured databases for maintaining the documents.

2. GREATER EFFICIENCY

The basic need of this website is efficiency and simplicity. The website should be efficient so that it can handle different entries according to their choice and user can easily view these entries and at the same time he can also request for them.

3. BETTER SERVICE

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

4. PERFORMANCE

During past several decades, there is need for effective and user-friendly system that give facility to reduce complexity. So, our computerized system is undertaken which is very user friendly and anyone can access it from one′s home also.

5. MINIMUM TIME REQUIRED

The proposed system will help the user to reduce the workload and provides user friendly environment so that he can easily do his jobs without time lagging.

3.4 FEASIBILITY ANALYSIS

In any project, feasibility analysis is a very important stage: here the project is checked for its feasibility. Any project may face scarcity in resources, time or workforce. Hence all these are to be studied in detail and a conclusion should be drawn whether the project under consideration is feasible or not. The main objective of the feasibility is to test the technical, social and economic feasibility of a project. System feasibility is attest or evaluation of the complete system plan. Such an evaluation is necessary to define the application area along with its extended and complexity, to provide the scope of computerization together with suggested output and input format and potential benefits.

During feasibility analysis for this project the following three primary areas of interest were considered.

- Technical feasibility
- Economic feasibility
- Operational feasibility

1. TECHNICAL FEASIBILITY

Technical feasibility is the most important of all types of feasibility analysis. An idea from the outline design to system requirements in terms of inputs, outputs, files and procedures is drawn and the type of hardware, software and the methods required for running the system are analysed. Keeping in mind the above considerations, the resource availability at this institution was observed. It was found that the institution has the efficient resources to develop the current project; hence the system is technically feasible.

2. ECONOMIC FEASIBILITY

This is judged by comparing the development cost against the income or benefit analysis, which is the basis for the economic justification of a system. In terms of benefits, we have to consider both tangible and intangible benefits here it I seen that no new software and hardware is needed for the development of the system. Thus, the project is economically feasible for development in this company.

3. OPERATIONAL FEASIBILITY

Operational feasibility I concerned with the working of the system after its installation. The company has a good record of development, installation and maintenance of systems for its clients. So, this system can be installed in the client environment and the company will help in maintenance of the system in future

3.5 REQUIREMENT SPECIFICATION

3.5.1 INTRODUCTION

Software requirement specification (SRS) is the requirement document that provides the technical specification for the design and development of the software. This document enhances the system's quality by formalizing communication between the system developer and the user and provides the proper information for accurate documentation. The produces a consequence of the analysis task at its culmination.

The introduction of the SRS states the goals and objectives of the software, describing it in the context of the computer-based system. It is nothing more than the software scope. The information description provides a detailed description of the problem that the software must solve. Information content, flow and structure are documented and hardware, software and human interfaces are described. A description of each function required to solve the problem is presented in the functional description. The behavioural description section of the specification examines the operation of the software as a consequence of external events and internally generated control characteristics.

Validation criteria are perhaps the most important and, ironically, the most often neglected section of the SRS. Specification of validation criteria act as an implicit review of all other requirement. Finally, the specification includes a Bibliography and Appendix. The bibliography contains references to all documents that relate to the software. The appendix contains information that supplements the specification. For example, tabular data, charts, description for algorithms etc.

SPECIFICATION REVIEW

A review of the SRS is conducted by both the software developer and the customer. The review is first conducted at a macroscopic level i.e.; reviewers attempt to ensure that the specification is completed, consistent and accurate when the overall information functional and behavioural domains are considered. Once the review is completed the SRS is "signed off" by both the customer and the developer. During the review changes to the specification may be recommended. Thus, it ensures that the developer and the customer have the same perception of the system.

3.5.2 SOFTWARE SPECIFICATION REQUIREMENT

This document describes the requirement of the system. It is meant for use by the developers, and will also be the basis for validating the final delivered system. Any changes made to the requirements in the future will have to go through a formal change approval process. The developer is responsible for asking the clarification, where necessary, and will not make any alterations without the permission of the client.

The developer is responsible for:

- Developing the site.
- Installing the software.
- Conducting demonstrations about the usage.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

System design involves translating information requirements and conceptual design into technical specification and general flow of processing. After the user requirements are identified, related information is gathered to verify the problem and after evaluating the existing system, a new system is proposed. The proposed system consists of various tables, their maintenance and report generation.

For design of get unsettled software, care has been given for developing an efficient system, which is user friendly as well as high in performance. It has been assured that the system will have the functions and promises of the proposed system. In the system, the various techniques are used to present a simple efficient system. Design phase acts a bridge between the software requirement specification and the implementation phase, which satisfies the requirements.

The major step in design is the preparation of input forms and the design of all major output forms in a manner acceptable to the user in all aspects. The base lies in the complete understanding of the system. The data flow diagrams explicitly specify the process flow. Table design or database design is the next major step. Extreme care has to be given here and several concepts of normalization have to be applied at many levels.

Program specification comes next. Here we specify various aspects of the program and also will in detail the major components used in the program. The overall process flow is also explained in much detail. Validation rules and checks come next. Several degrees of validation have to be applied to all outputs and various other operations made on the system. Deviation, if any, has to be checked from these validation rules, imposing the ‘not null’ constraint is one of the best examples. It has been used many aspects. Various other constraints are also used. Security checks refer to avoiding unnecessary access to data that is under use and guarding data from any malice.

Inputs, outputs have to be designed as per predefined guidelines. Effective and meaningful navigation has to be applied. In the input design, the user-oriented inputs are converted into computer-based formats whereas in the output design, the emphasis is on producing the hard copy or softcopy of the information requested for.

4.2 DATA FLOW DIAGRAM

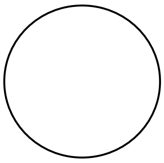
A Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. It′s a structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process-oriented system flowcharts.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

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

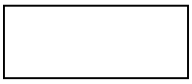
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Main symbols used in the data flow diagram are:

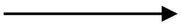
1. Circle represents a process that transforms incoming data flows in to outgoing data flows.



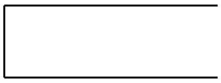
2. A square defines a source and destination of system data.



3. Arrow identifies data in motion.



4. An open rectangle defines a data store, data at rest or temporary repository of data.



Steps to Construct Data Flow Diagrams:

Four steps are commonly used to construct a DFD. They are

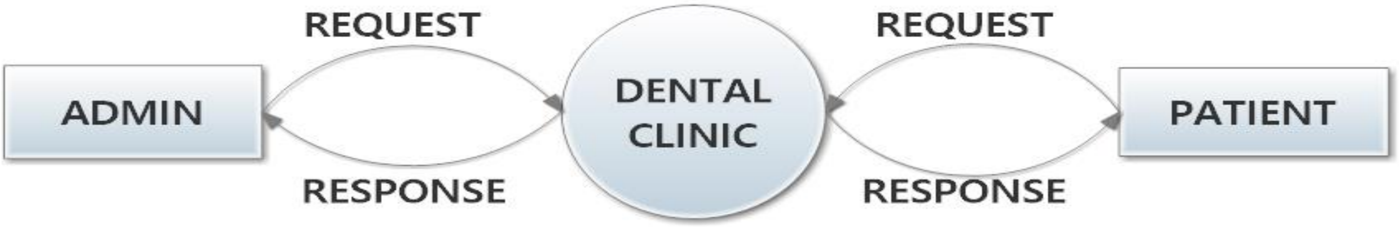
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The destination of flow is from top to bottom and from left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations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

Rules for constructing a Data Flow Diagram:

- Process should be named and numbered for easy reference.
- The direction of flow is from top to bottom and from left to right.
- When a process is explored into lower level details they are numbered.
- The name of data stores, sources and destinations are written in capital letter.

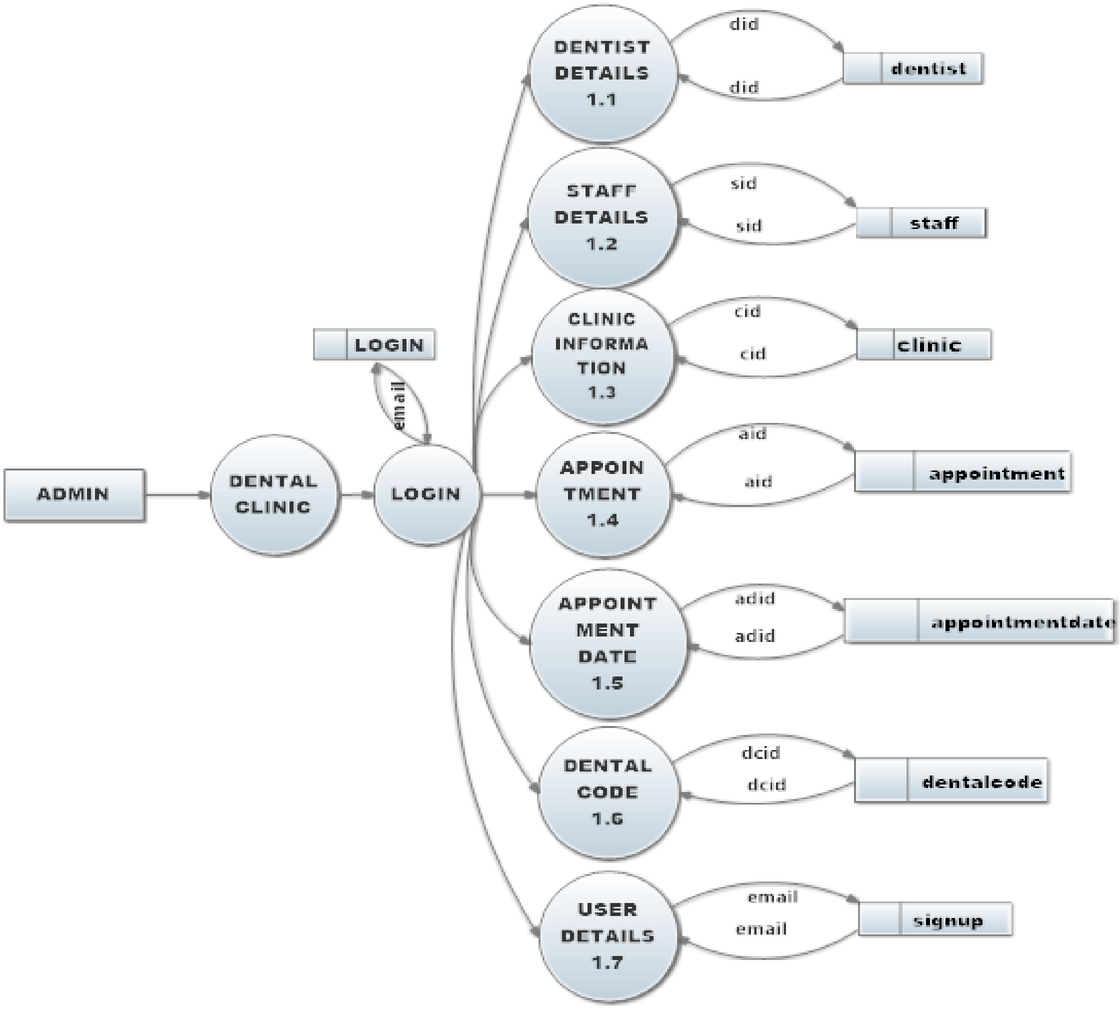
LEVEL 0

FIGURE 1



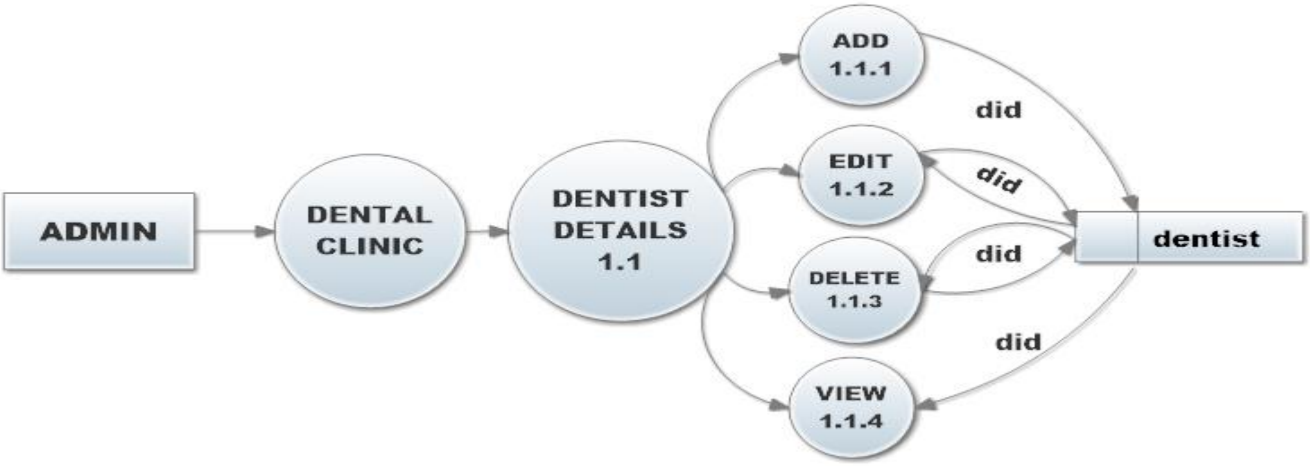
LEVEL 1-ADMIN

FIGURE 2



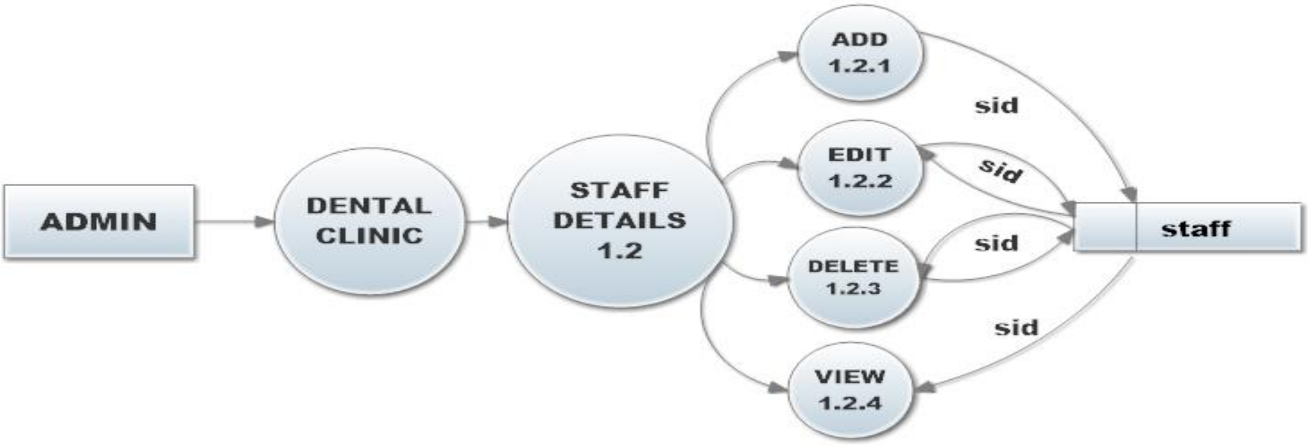
LEVEL 1.1

FIGURE 3



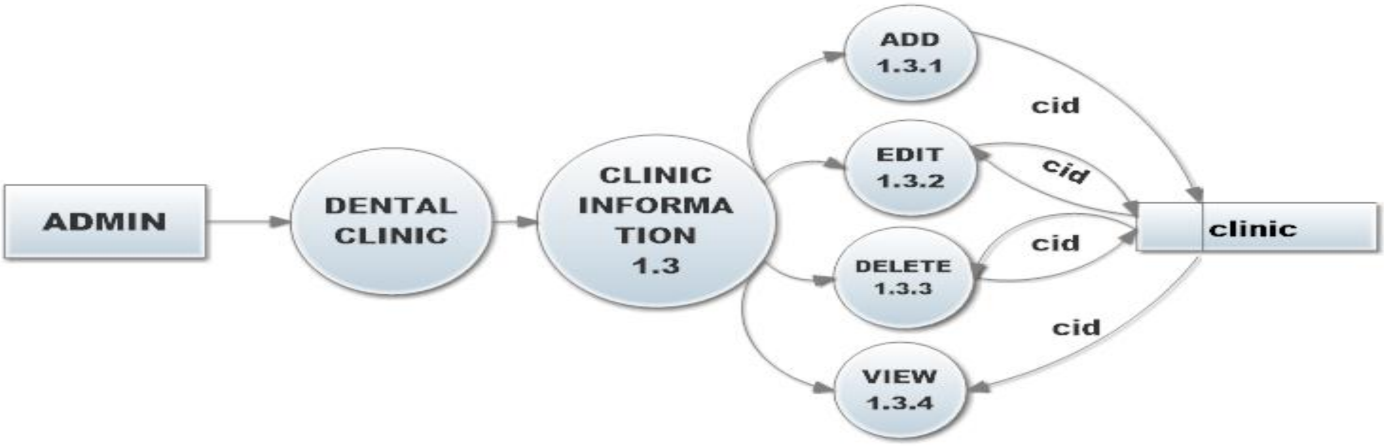
LEVEL 1.2

FIGURE 4



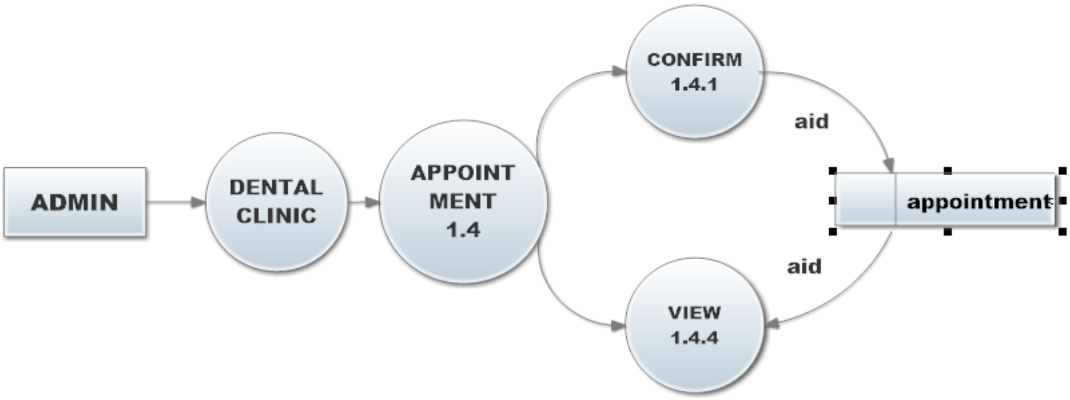
LEVEL 1.3

FIGURE 5



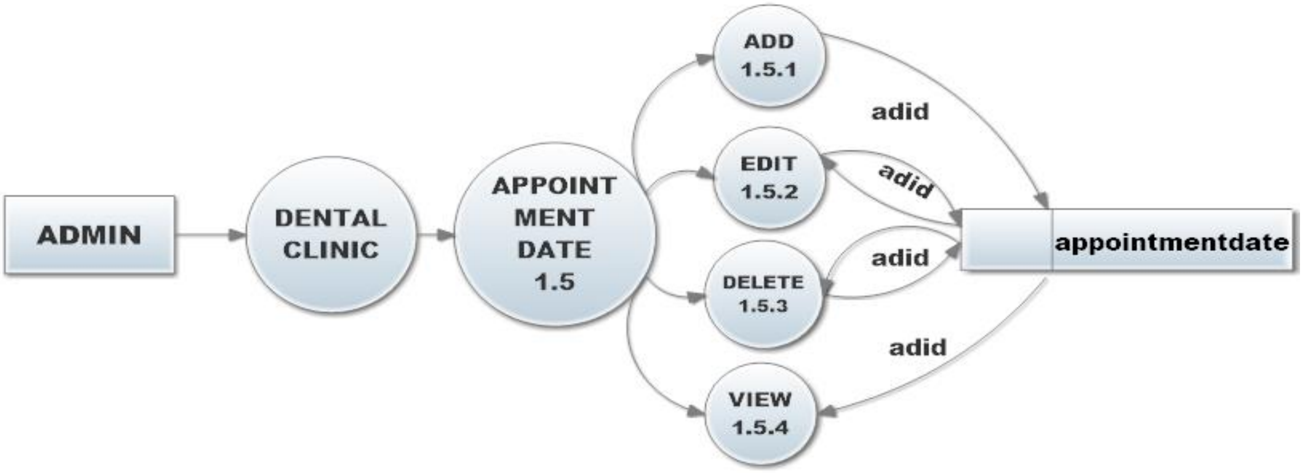
LEVEL 1.4

FIGURE 6



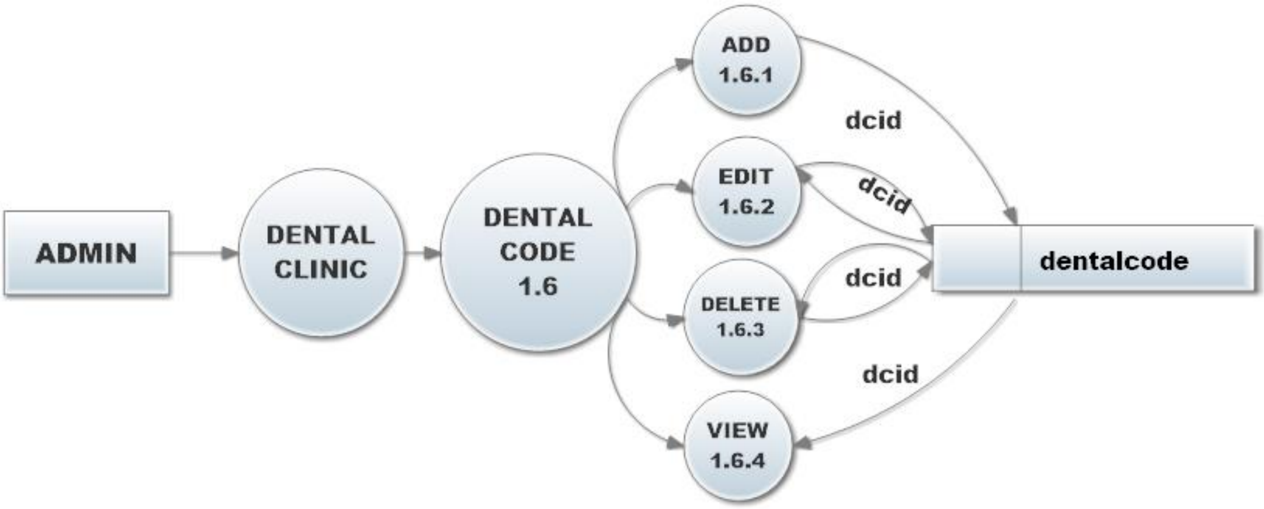
LEVEL 1.5

FIGURE 7



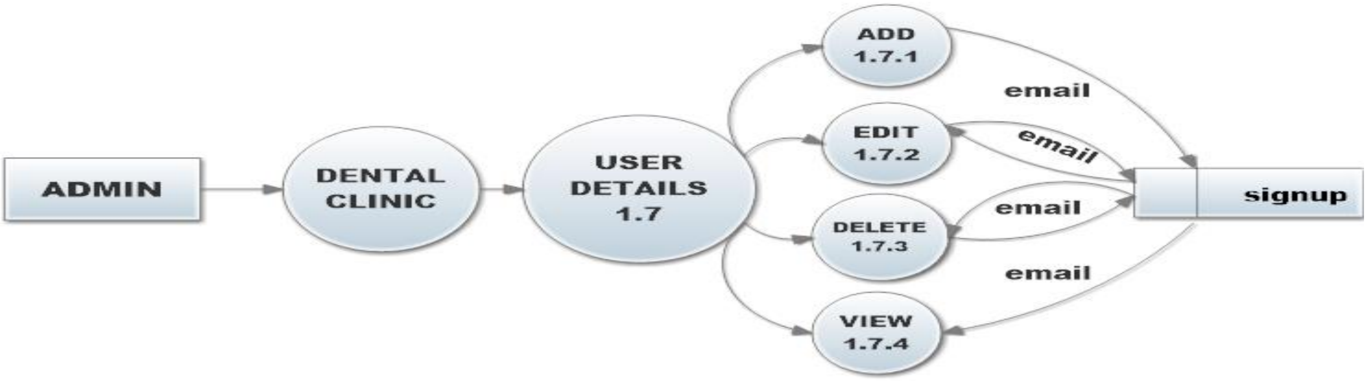
LEVEL 1.6

FIGURE8



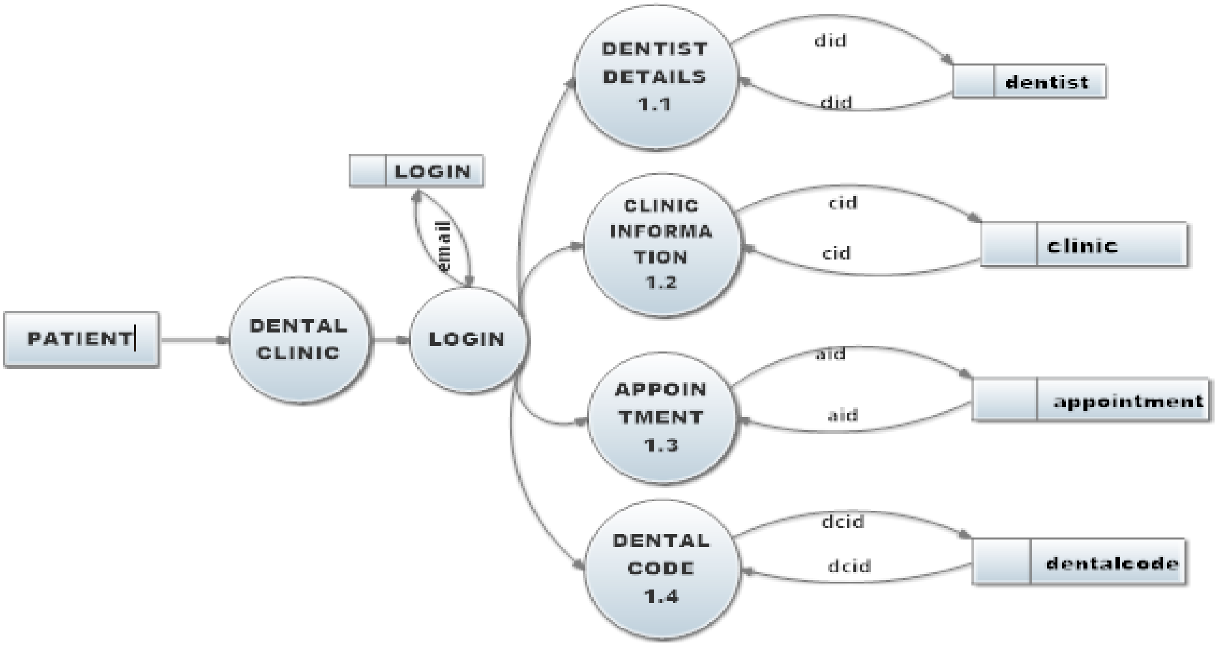
LEVEL 1.7

FIGURE 9



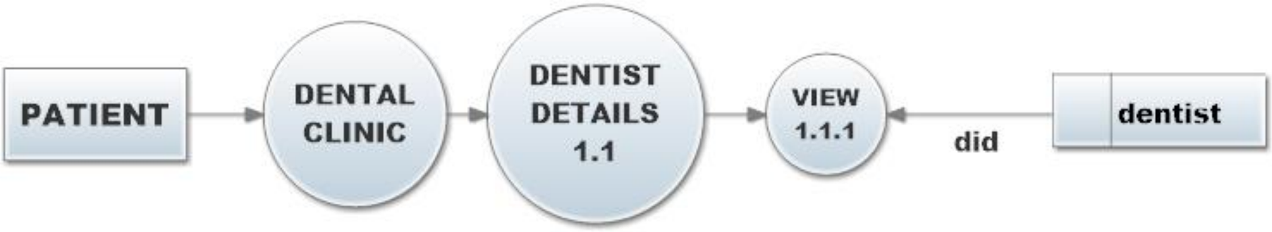
LEVEL 1-PATIENT

FIGURE 10



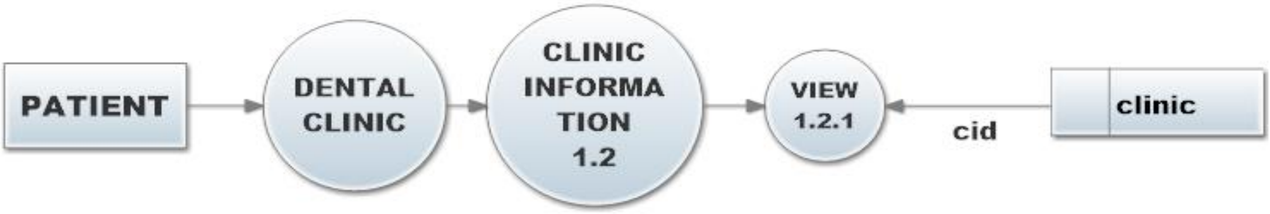
LEVEL 1.1

FIGURE 11



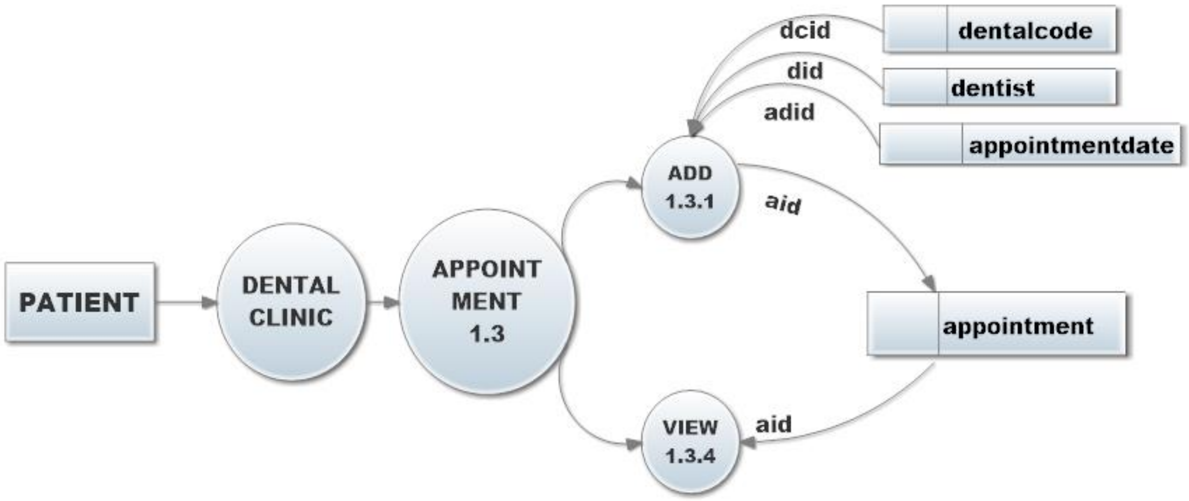
LEVEL 1.2

FIGURE 12



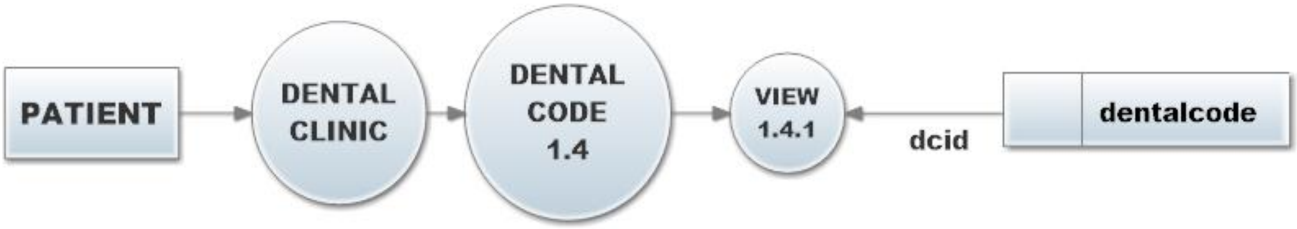
LEVEL 1.3

FIGURE 13



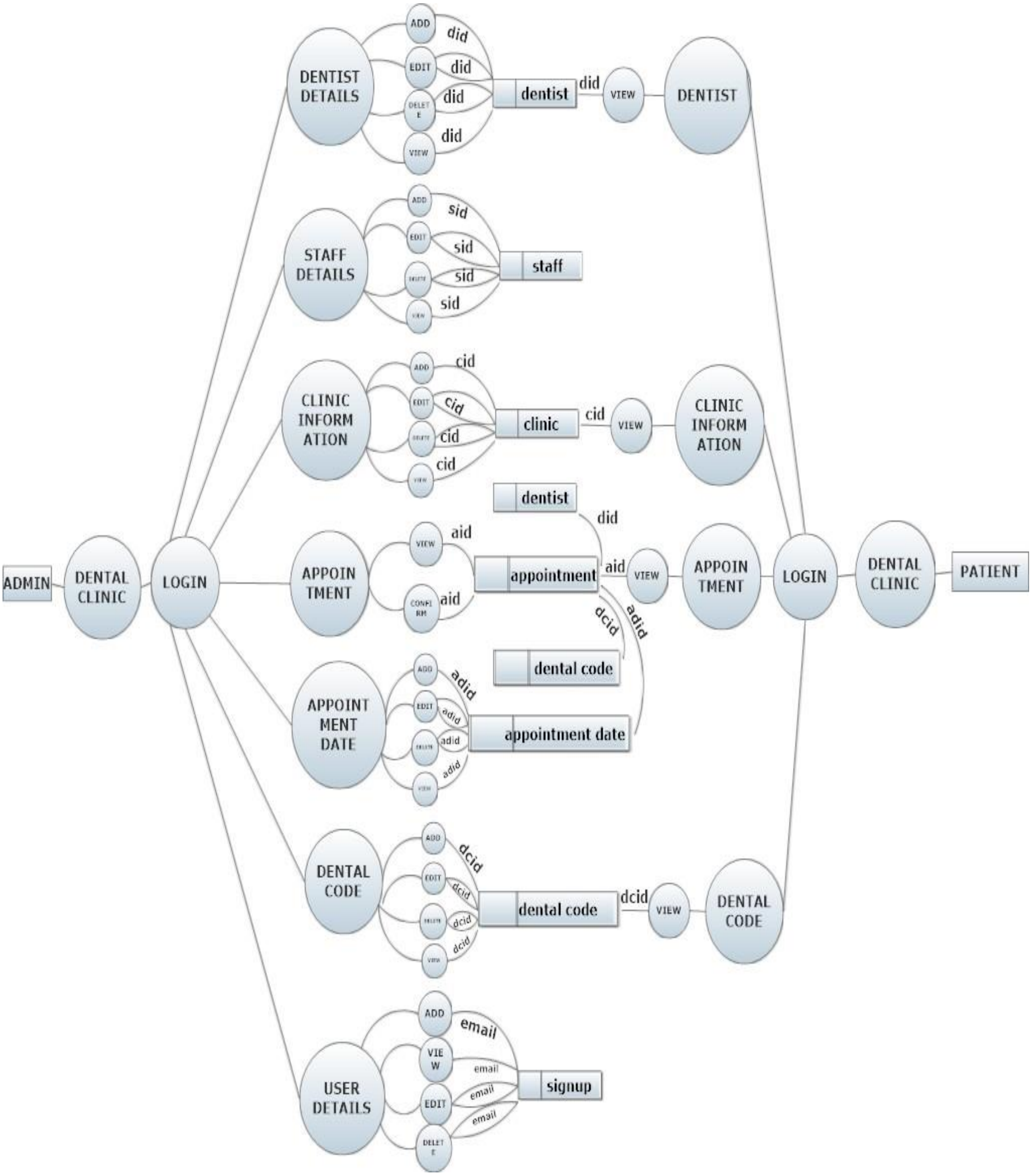
LEVEL 1.4

FIGURE 14



LEVEL 2

FIGURE 15



4.3 INPUT DESIGN

The input is the set of values that is provided by the user to the system. The input design must enable the user to provide the error free input to the system for efficient processing. The input design is the process of converting the user-oriented inputs into computer-based formats. The data fed into the system using simple interactive forms. The forms have been supplied with messages so that user can enter data without facing any difficulty. The data is validated wherever it requires in the project. The input data have to be validated, edited, organized, and accepted by the system before being proposed to produce the outputs.

The main objectives of input design are as follows:

- Produce effective method of input
- Achieve high level accuracy
- Ensure that the input is acceptable and understood by the user

The different types of input data handled by the system are:

EXTERNAL

They are the primary inputs to the system. The external input is what the user supplies to the system. The user can give different types of external inputs in this project such as add new threads, post reply

INTERNAL

When the external inputs are obtained from the user, these inputs are transferred to the system as messages. These messages are captured and handled as input for further processing. In this project the input design is done with PHP codes. The external inputs are data given to the system by the user such as username and password for authentication process. The external input also includes the request as per the users interest for displaying today’s, yesterdays and last week’s threads/posts and its replies. The internal input covers the fetching of data from the database and it will be the input for displaying the results of the screen

The necessary internal inputs are given to the system by Graphical User Interface (GUI) technology. The GUI system applied to this project enables the user to avoid error and conclusion arises while entering the input

4.4 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any systems results of the processing are communicated to the user and to the other systems through outputs design it is determined how the information is to be displayed for immediate need. It is the most important and direct source information to the user. Efficient and intelligent output design improves the systems relationship with the user and helps in decision making. The objectives of the output design is to convey the information of all the past activities, current status and to emphasize important events

The output generally refers to the results and information that is generated from the system. Outputs from computers are required primarily to communicate the results of processing to the users. The result for each query option that is submitted by the user, the system displays the output. The output that is obtained for each query submitted should be tested before conforming the result

4.5 DATABASE DESIGN

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for the users. The general theme behind a database is to integrate all the information. In database design several specific objectives are considered: -

- Controlled redundancy
- Ease of learning and use
- Data independence
- Accuracy and integrity
- Recovery from failure
- Performance

A database is an integrated collection of data which provides centralized access to the data. Usually the centralized data managing the software is called RDBMS and the other DBMS is the separation of data as seen by the program and data has stored in direct access to stores device. This is the difference between logical and physical data

DESIGN CONSIDERATION

The system is analysed to the requirements and possible tables and fields are identified.

- Identifying keys: Once we have drawn up the list of possible tables and fields, the next step in the logic database is to identify and set foreign keys for each table.
- Primary keys: A primary key consist of a field or a set of fields that uniquely identify each record in that table. The ʺprimary ʺfield defines the primary key.
- Foreign key: A foreign key comprises a field or multiple field that links to the primary key of another table.

DATABASE DESIGN AND TABLE

Database is recognized as standard of MIS and is available virtually for every computer system. The general theme behind a database is to integrate all the information. A database is an integrated collection of data and provides centralized access to the data. Databases are designed to manage large bodies of information. One of the major purposes of a database system is to provide users with an abstract view of data. A database is designed so that it can be used both to specify the overall logical structure of the database and provide a higher-level description of the implementation. The database is structured in fixed format records of several types. Each record type defines a fixed number of fields or attributes and each field usually of a fixed length.

4.6 TABLES

Table 1-Login

Fields	Data Type	Size	Constraints	Column Description
email	Varchar	30	Primary Key	Email Address
Password	Varchar	30	Not Null	Password

Table 2-Signup

Fields	Data Type	Size	Constraints	Column Description
name	Varchar	30	Not Null	Name
phone	Bigint	10	Not Null	Phone Number
age	Int	10	Not Null	Age
address	Varchar	30	Not Null	Address
email	Varchar	30	Primary Key	Email Address
password	Varchar	30	Not Null	Password

Table 3-Dentist

Fields	Data Type	Size	Constraints	Column Description
did	int	10	Primary key	Dentist id
name	Varchar	30	Not Null	Name
sex	Varchar	30	Not Null	Sex
age	Int	10	Not Null	Age
phone	Bigint	10	Not Null	Phone Number
address	Varchar	30	Not Null	Address
email	Varchar	30	Primary Key	Email Address
dtype	Varchar	30	Not Null	Dentist Type

TABLE 4-Staff

Fields	Data Type	Size	Constraints	Column Description
sid	int	10	Primary key	Staff id
name	Varchar	30	Not Null	Name
sex	Varchar	30	Not Null	Sex
age	Int	10	Not Null	Age
phone	Bigint	10	Not Null	Phone Number
address	Varchar	30	Not Null	Address
email	Varchar	30	Primary Key	Email Address

Table 5-Clinic

Fields	Data Type	Size	Constraints	Column Description
cid	int	10	Primary key	Clinic id
cname	Varchar	30	Not Null	Clinic Name
location	Varchar	30	Not Null	Location
openhr	Varchar	30	Not Null	Opening hrs
closehr	Varchar	30	Not Null	Closing hrs
rooms	Int	10	Not Null	Number of Rooms

Tables 6-Appointment

Fields	Data Type	Size	Constraints	Column Description
aid	Int	10	Primary key	Appointment id
dcid	Int	10	Foreign key	Dental code id
dentalcode	Varchar	30	Not Null	Dental code
did	Int	10	Foreign key	Dentist id
dentist	Varchar	30	Not Null	Dentist
adid	Int	10	Foreign key	Appointmentdate id
regdate	Varchar	30	Not Null	Appointment Date
regtime	Varchar	30	Not Null	Appointment Time
status	Varchar	30	Not Null	Status

Tables 7-Dental Codes

Fields	Data Type	Size	Constraints	Column Description
dcid	Int	10	Primary key	Dental code id
dentalcode	Int	10	Not Null	Dental code
Unitcost	Varchar	**30**	Not Null	Unit cost
description	Varchar	**30**	Not Null	Code Description

Tables 8-Appointment Date

Fields	**Data Type**	**Size**	**Constraints**	**Column Description**
adid	Int	10	Primary key	Dental code id
regdate	Varchar	30	Not Null	Appointment Date
cname	Varchar	**30**	Not Null	Clinic Name
cid	int	10	Foreign key	Clinic id

CHAPTER 5

SYSTEM DEVELOPMENT

5.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage the main workload, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion.

Implementation includes all those activities that take place to convert from the old system to new system. The new system may be totally new, replacing an existing manual or automated system or it may be a major modification to an existing system. Proper implementation is essential to provide a reliable system to meet the organization requirements. Successful implementation may not guarantee improvement in the organization using the new system, but improper installation will prevent it.

The implementation stage involves the following tasks:

- Careful planning
- Investigation of system and constrains
- Design of methods to achieve the changeover phase
- Training of staffs in the changeover phase
- Evaluation of the changeover method

The method of implementation and the time scale to be adopted are found out initially. Next the system is tested properly and the same time users are trained in the new procedures.

IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system, people who are not sure that the software is meant to make their job easier. In the initial stage, they doubt about the software but we have to ensure that the resistance does not build up as one has to make sure that

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

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

5.2 IMPLEMENTATION LOGIC

Implementation includes all those activities that take place to convert from the old system to the new one. The new system may be totally new, replacing an existing manual automated system. Proper implementation is essential to provide a reliable system to meet customer requirements.

The process of putting developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after thorough testing is done and if it is found to be working according to the specifications. The system personally checks the feasibility of the system.

The implementation stage involves following tasks:

- Investigation of system and constrains.
- Design of methods to achieve the changeover.
- Evaluation of the changeover method.

The newly proposed system is implemented after the successful testing of the system. The final step of the system approach recognizes that an implemented solution should be monitored and evaluated. This is called post implementation review process. Since the success of a solution is reviewed after it is implemented. The focus on this stepwise to determine if the implementation solution has indeed helped the institution and the organizers of the event, meet their system objectives.

5.3 CODING

Coding is the phase of a software development project where developer's actually in put the source code into a computer that will be compiled into the final software program. Source code is the high-level language like C#, java, python etc. that is typed into an IDE (Interactive Development Environment) and stored in the text file on the computer. This text file is compiled into machine code, which are the instructions actually understood by the computer.

5.4 CODING VALIDATION AND OPTIMIZATION

It is verified whether the data entered in each form is added to the corresponding fields of the table. On the press of ADD button, controls will appear on the form and on the press of SAVE button, the entered data is saved.

The lower keys letters entered are detected and changed to upper case. Also, numbers are not allowed to be entered in the text boxes.

- Validation is the status of the project when the theoretical designs turned into a working system.
- It is used to reduce the number of loops in the program.
- Optimization is the last part of the system development life cycle.
- If the number of loops increases no. of executions also increases. Then there may be a chance for the program to get stuck.

5.5 SAMPLE CODES

LOGIN

```
<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

        <div class="limiter">

        <div class="container-login100" style="background-image: url('images/bg-01.jpg');">

                <div class="wrap-login100 p-t-30 p-b-50">

                        <span class="login100-form-title p-b-41">

                                Patient Login

                        </span>


<form class="login100-form validate-form p-b-33 p-t-5" action="login.php" method="post"
class="form">

<div class="wrap-input100 validate-input" data-validate = "Enter username">

<input class="input100" type="email" name="email" placeholder="Email"

value="<?php if (isset($_POST['email'])) echo $_POST['email']; ?>" >

<span class="focus-input100" data-placeholder="&#xe82a;"></span>

</div>

<div class="wrap-input100 validate-input" data-validate="Enter password">

<input class="input100" type="password" name="password" placeholder="password"

value="<?php if (isset($_POST['password'])) echo $_POST['password']; ?>"
```

```
<span class="focus-input100" data-placeholder="&#xe80f;"></span>

</div>

<div class="container-login100-form-btn m-t-32">

                                        <button class="login100-form-btn" input id="submit">

                                                Login



                                        </button>

                                </div>

<br>&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp
&nbsp&nbsp&nbsp&nbsp&nbsp<p23> Create an account? <a href="signup.php">click to
register </a> </p23>

                        </form>

                </div>

        </div>

</div>
```

ADD DENTIST

```
<!doctype html>

<html lang=en>

<head>

<title>Register page</title>

<meta charset=utf-8><!--important prerequisite for escaping problem characters-->

<link rel="stylesheet" type="text/css" href="dentist.css">

</head>

<body>
```

```
<br>&nbsp
<div align="left">
<a href="admin.php"  class="col head" >Back</a>
        </div>
<div align="right">

<a href="index.html"  class="col head" >Logout</a>
</div>
<div id="container">
<div id="content"><!-- Registration handler content starts here -->
<?php
// The link to the database is moved to the top of the PHP code.
require ('connect-mysql.php'); // Connect to the db.
// This query INSERTs a record in the users table.
// Has the form been submitted?
if ($_SERVER['REQUEST_METHOD'] == 'POST')
{
        $errors = array(); // Initialize an error array.
        // Check for a first name:
        if (empty($_POST['name'])) {
        $errors[] = 'You forgot to enter your name.';
        } else {
        $name = mysqli_real_escape_string($dbcon, trim($_POST['name']));
        }
```

```
// Check for age

        if (empty($_POST['age'])) {

        $errors[] = 'You forgot to enter your age.';

        } else {

        $age = mysqli_real_escape_string($dbcon, trim($_POST['age']));

        }


// Check for sex

        if (empty($_POST['sex'])) {

        $errors[] = 'You forgot to enter your sex.';

        } else {

        $sex= mysqli_real_escape_string($dbcon, trim($_POST['sex']));

        }


// Check for an address

        if (empty($_POST['address'])) {

        $errors[] = 'You forgot to enter your address.';

        } else {

        $address = mysqli_real_escape_string($dbcon, trim($_POST['address']));

        }


// Check for phone no.

        if (empty($_POST['phone'])) {
```

```
        $errors[] = 'You forgot to enter your phone no.';

        } else {

        $phone = mysqli_real_escape_string($dbcon, trim($_POST['phone']));

        }


// Check for an email address

        if (empty($_POST['email'])) {

        $errors[] = 'You forgot to enter your email address.';

        } else {

        $email= mysqli_real_escape_string($dbcon, trim($_POST['email']));

        }


// Check for dentist type

        if (empty($_POST['dtype'])) {

        $errors[] = 'You forgot to select your dentist type';

        } else {

        $dtype= mysqli_real_escape_string($dbcon, trim($_POST['dtype']));

        }

if (empty($errors)) { // If it runs

        // Register the user in the database...

        // Make the query:

        $q = "INSERT INTO dentist (did, name, sex, age, phone, address, email, dtype,
registration_date)

        VALUES (' ', '$name', '$sex', '$age', '$phone', '$address', '$email','$dtype' , NOW() )";

        $result = @mysqli_query ($dbcon, $q); // Run the query.
```

```
    if ($result) { // If it runs

    header ("location: admin.php");

    exit();

    } else { // If it did not run

    // Message:

    echo '<h2 class="error">System Error</h2>

    <p class="error">You could not be registered due to a system error. We apologize

    for any inconvenience.</p>';

    // Debugging message:

    echo '<p>' . mysqli_error($dbcon) . '<br><br>Query: ' . $q . '</p>';

    } // End of if ($result)

    mysqli_close($dbcon); // Close the database connection.

    // Include the footer and quit the script:

    exit();

    } else { // Report the errors.

        echo '<h2 class="error">Error!</h2>

    <p class="error">The following error(s) occurred:<br>';

    foreach ($errors as $msg) { // Extract the errors from the array and echo them

    echo " - $msg<br>\n";

    }

    echo '</p><h3 class="error">Please try again.</h3><p><br></p>';

    }// End of if (empty($errors))

} // End of the main Submit conditional.

?>
```

```
<h2 class="title">Insert Dentist Information</h2>

<form action="dentist.php" method="post" class="form">

<p><label class="label" for="name">Name:</label>
<input  type="text" name="name" size="30" maxlength="30"
value="<?php if (isset($_POST['name'])) echo $_POST['name']; ?>"></p>

<p><label class="label" for="sex">Sex:</label>
<input  type="text" name="sex" size="30" maxlength="60"
value="<?php if (isset($_POST['sex'])) echo $_POST['sex']; ?>" > </p>

<p><label class="label" for="age">Age:</label>
<input  type="number" name="age" size="30" maxlength="60"
value="<?php if (isset($_POST['age'])) echo $_POST['age']; ?>" > </p>

<p><label class="label" for="phone">Phone No:</label>
<input  type="tel" pattern="[789][0-9]{9}" name="phone" size="30" maxlength="60"
value="<?php if (isset($_POST['phone'])) echo $_POST['phone']; ?>" > </p>

<p><label class="label" for="address">Address:</label>
<input  type="text" name="address" size="30" maxlength="60"
value="<?php if (isset($_POST['address'])) echo $_POST['address']; ?>" > </p>
```

```
<p><label class="label" for="email">Email:</label>
<input id="email" type="text" name="email" size="30" maxlength="60"
value="<?php if (isset($_POST['email'])) echo $_POST['email']; ?>" > </p>
<p><label class="label" for="dtype">Dentist Type:</label><select name="dtype"
value="<?php if (isset($_POST['dtype'])) echo $_POST['dtype']; ?>">
  <option>Permanent</option>
  <option>Trainee</option>
  <option>Visiting</option>

</select></p>
<p><input id="submit" type="submit" name="submit" value="Register"></p>
</form>
</p>
</div>
</div>
</body>
</html>
```

EDIT DENTIST

```
<!doctype html>
<html lang=en>
<head>
<title>Edit a record</title>
<meta charset=utf-8>
<link rel="stylesheet" type="text/css" href="dentist.css">
```

```
</head>

<body>

<br>&nbsp

<div align="left">

<a href="admin.php"  class="col head" >Back</a>

</div>

<div align="right">

<a href="index.html"  class="col head" >Logout</a>

</div>


<div id="container">

<div id="content"><!--Start of the edit page content-->

<h2></h2>

<?php

        // After clicking the Edit link in the found_record.php page, the editing interface appears

        // The code looks for a valid user ID, either through GET or POST #1

        if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) { // From view_users.php

        $id = $_GET['id'];

        }

        elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) { // Form submission

        $id = $_POST['id'];

        }

        else { // If no valid ID, stop the script

        echo '<p class="error">This page has been accessed in error</p>';
```

```
            exit();
            }
require ('connect-mysql.php');
// Has the form been submitted?
if ($_SERVER['REQUEST_METHOD'] == 'POST')
{
            $errors = array();
// Look for the dental codes
        if (empty($_POST['name'])) {
            $errors[] = 'You forgot to enter your first name';
            } else {
            $name = mysqli_real_escape_string($dbcon, trim($_POST['name']));
            }

// Look for the descriptions
            if (empty($_POST['age'])) {
            $errors[] = 'You forgot to enter your age.';
            } else {
            $age = mysqli_real_escape_string($dbcon, trim($_POST['age']));
            }

// Look for the descriptions
            if (empty($_POST['phone'])) {
            $errors[] = 'You forgot to enter the phone no.';
            } else {

            $phone = mysqli_real_escape_string($dbcon, trim($_POST['phone']));

            }


   // Look for the descriptions

            if (empty($_POST['email'])) {

            $errors[] = 'You forgot to enter the email id.';

            } else {

            $email = mysqli_real_escape_string($dbcon, trim($_POST['email']));

            }

 // Look for the descriptions

            if (empty($_POST['address'])) {

            $errors[] = 'You forgot to enter the address.';

            } else {

            $address = mysqli_real_escape_string($dbcon, trim($_POST['address']));

            }


// Look for the descriptions

            if (empty($_POST['dtype'])) {

            $errors[] = 'You forgot to enter the address.';

            } else {

            $dtype = mysqli_real_escape_string($dbcon, trim($_POST['dtype']));

            }
```

```
if (empty($errors))
    { // If everything is OK, make the update query
    // Check that the email is not already in the users table
    $q = "UPDATE dentist SET name='$name', age='$age' , phone='$phone'
,email='$email' , address='$address',dtype='$dtype' WHERE did=$id LIMIT 1";
    $result = @mysqli_query ($dbcon, $q);
    if (mysqli_affected_rows($dbcon) == 1) { // If it ran OK
    // Echo a message if the edit was satisfactory
    echo '<h3>The user has been edited.</h3>';
    } else { // Echo a message if the query failed
    echo '<p class="error">The user could not be edited due to a system error.
    We apologize for any inconvenience.</p>'; // Error message.
    echo '<p>' . mysqli_error($dbcon) . '<br />Query: ' . $q . '</p>'; // Debugging message.
    } // End of if ($result)
    mysqli_close($dbcon); // Close the database connection.
    // Include the footer and quit the script:


    exit();
    } else   { // Display the errors.
    echo '<p class="error">The following error(s) occurred:<br />';


    foreach ($errors as $msg) { // Extract the errors from the array and echo them
    echo " - $msg<br>\n";
  }
    echo '</p><p>Please try again.</p>';
```

```
            } // End of if (empty($errors))section
}       // End of the conditionals
        // Select the record
$q = "SELECT did,name,age,phone,email,address,dtype FROM dentist WHERE did=$id";
$result = @mysqli_query ($dbcon, $q);
if (mysqli_num_rows($result) == 1)
{   // Valid user ID, display the form.
// Get the user's information
$row = mysqli_fetch_array ($result, MYSQLI_NUM);
// Create the form
echo '<form action="edit_dentistinfo.php" method="post">
<p><label class="label" for="name">First Name:</label>
<input class="fl-left" id="name" type="text" name="name" size="30" maxlength="30"
value="' . $row[1] . '"></p>
<p><label class="label" for="age">Age:</label>
<input class="fl-left" type="text" name="age" size="30" maxlength="50"
value="' . $row[2] . '"></p>
<p><label class="label" for="phone">Phone:</label>
<input class="fl-left" type="text" name="phone" size="30" maxlength="50"
value="' . $row[3] . '"></p>
<p><label class="label" for="email">Email:</label>
<input class="fl-left" type="text" name="email" size="30" maxlength="50"
value="' . $row[4] . '"></p>
<p><label class="label" for="address">Address:</label>
```

```
<input class="fl-left" type="text" name="address" size="30" maxlength="50"

value="' . $row[5] . '"></p>

<label class="label" for="dtype">Dentist Type:</label><select name="dtype" value="' .
$row[6] . '">

      <option>Permanent</option>

      <option>Trainee</option>

      <option>Visiting</option>

      </select></p>


<p><input id="submit" type="submit" name="submit" value="Edit"></p>

<input type="hidden" name="id" value="' . $id . '" />

</form>';

}

else { // The record could not be validated

  echo '<p class="error">This page has been accessed in error</p>';

 }

mysqli_close($dbcon);

?>

</div>

</div>

</body>

</html>
```

DELETE DENTIST

```
<?php
session_start();
?>
<!doctype html>
<html lang=en>
<head>
<title>Delete a record</title>
<meta charset=utf-8>
<link rel="stylesheet" type="text/css" href="includes.css">
</head>
<body>
<br>&nbsp
<div align="left">

        <a href="admin.php"  class="col head" >Back</a>
        </div>

<div align="right">

        <a href="index.html"  class="col head" >Logout</a>
        </div>
<div id="container">

<div id="content"><!--Start of content for delete page-->
<h2>Delete a Record</h2>
<?php
// Check for a valid user ID, through GET or POST
if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) { //Details from view_users.php
$id = $_GET['id'];
```

```
} elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) { // Form submission #1
$id = $_POST['id'];
} else { // If no valid ID, stop the script
echo '<p class="error">This page has been accessed in error.</p>';

exit();
}
require ('connect-mysql.php');
// Has the form been submitted? #2
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
if ($_POST['sure'] == 'Yes') { // Delete the record
// Make the query
$q = "DELETE FROM dentist WHERE did=$id LIMIT 1";
$result = @mysqli_query ($dbcon , $q);
if (mysqli_affected_rows($dbcon ) == 1) { // If there was no problem
// Display a message
echo '<h3>The record has been deleted.</h3>';
} else { // If the query failed to run
echo '<p class="error">The record could not be deleted.<br>Probably
because it does not exist or due to a system error.</p>'; // Display error message
echo '<p>' . mysqli_error($dbcon ) . '<br />Query: ' . $q . '</p>';
// Debugging message
}
} else { // Confirmation that the record was not deleted
echo '<h3>The user has NOT been deleted.</h3>';
}
} else { // Display the form
// Retrieve the member's data #3
$q = "SELECT name FROM dentist WHERE did=$id";
$result = @mysqli_query ($dbcon , $q);
if (mysqli_num_rows($result) == 1) { // Valid user ID, show the form
// Get the member's data
$row = mysqli_fetch_array ($result, MYSQLI_NUM);
// Display the name of the member being deleted
echo "<h3>Are you sure you want to permanently delete $row[0]?</h3>";
// Display the delete page
echo '<form action="deletedentist.php" method="post">
<input id="submit-yes" type="submit" name="sure" value="Yes">
<input id="submit-no" type="submit" name="sure" value="No">
<input type="hidden" name="id" value="' . $id . '">
</form>';
} else { // Not a valid member’s ID
echo '<p class="error">This page has been accessed in error.</p>';
echo '<p>&nbsp;</p>';
}
} // End of the main conditional section
mysqli_close($dbcon );
?>
</div>
</div>
</body>
</html>
```

CONNECTION

```
<?php
DEFINE ('DB_USER','root');
DEFINE ('DB_PSWD',");
DEFINE ('DB_HOST','localhost');
DEFINE ('DB_NAME','dental');
```

```
$dbcon=mysqli_connect(DB_HOST,DB_USER,DB_PSWD,DB_NAME);
if(!$dbcon){
        die('error connecting to database');
}
echo '';
?>
```

SIGNUP

```
<!doctype html>
<html lang=en>
<head>
<?php
// The link to the database is moved to the top of the PHP code.
require ('connect-mysql.php'); // Connect to the db.
// This query INSERTs a record in the users table.
// Has the form been submitted?
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
$errors = array(); // Initialize an error array.
// Check for a name:
if (empty($_POST['name'])) {
$errors[] = 'You forgot to enter your Name.';
} else {
$name = mysqli_real_escape_string($dbcon, trim($_POST['name']));
}
// Check for phone number
if (empty($_POST['phone'])) {
$errors[] = 'You forgot to enter your phone number.';
} else {
$phone = mysqli_real_escape_string($dbcon, trim($_POST['phone']));
}
```

```
// Check for age
if (empty($_POST['age']) ) {
$errors[] = 'You forgot to enter your age.';
}
if($_POST['age'] > 90)
{
        $errors[] = 'age should be below 80.';
}
 else {
$age = mysqli_real_escape_string($dbcon, trim($_POST['age']));
}
// Check for an address
if (empty($_POST['address'])) {
$errors[] = 'You forgot to enter your address.';
} else {
$address = mysqli_real_escape_string($dbcon, trim($_POST['address']));
}
// Check for an email address
if (empty($_POST['email'])) {
$errors[] = 'You forgot to enter your email address.';
} else {
$email = mysqli_real_escape_string($dbcon, trim($_POST['email']));
}
// Check for a password and match it against the confirmed password
if (!empty($_POST['psword1'])) {
if ($_POST['psword1'] != $_POST['psword2']) {
$errors[] = 'Your two passwords did not match.';
} else {
$p = mysqli_real_escape_string($dbcon, trim($_POST['psword1']));
}
} else {
```

```
$errors[] = 'You forgot to enter your password.';
}
if (empty($errors)) { // If it runs
// Register the user in the database...
// Make the query:
$q = "INSERT INTO signup (userid, name, age, phone, address,email, password, registration_date)
VALUES (' ', '$name', '$age', '$phone','$address','$email', '$p', NOW() )";
$result = @mysqli_query ($dbcon, $q); // Run the query.
if ($result) { // If it runs
header ("location: index.html");
exit();
} else { // If it did not run
// Message:
echo '<h2>System Error</h2>
<p class="error">You could not be registered due to a system error. We apologize □
for any inconvenience.</p>';
// Debugging message:
echo '<p>' . mysqli_error($dbcon) . '<br><br>Query: ' . $q . '</p>';
} // End of if ($result)
mysqli_close($dbcon); // Close the database connection.
// Include the footer and quit the script:
include ('footer.php');
exit();
} else { // Report the errors.
        echo '<h2 class="error">Error!</h2>
<p class="error">The following error(s) occurred:<br>';
foreach ($errors as $msg) { // Extract the errors from the array and echo them
echo "<p class='error'> - $msg<br></p>\n";
}
echo '</p><h3 class="error">Please try again.</h3><p><br></p>';
```

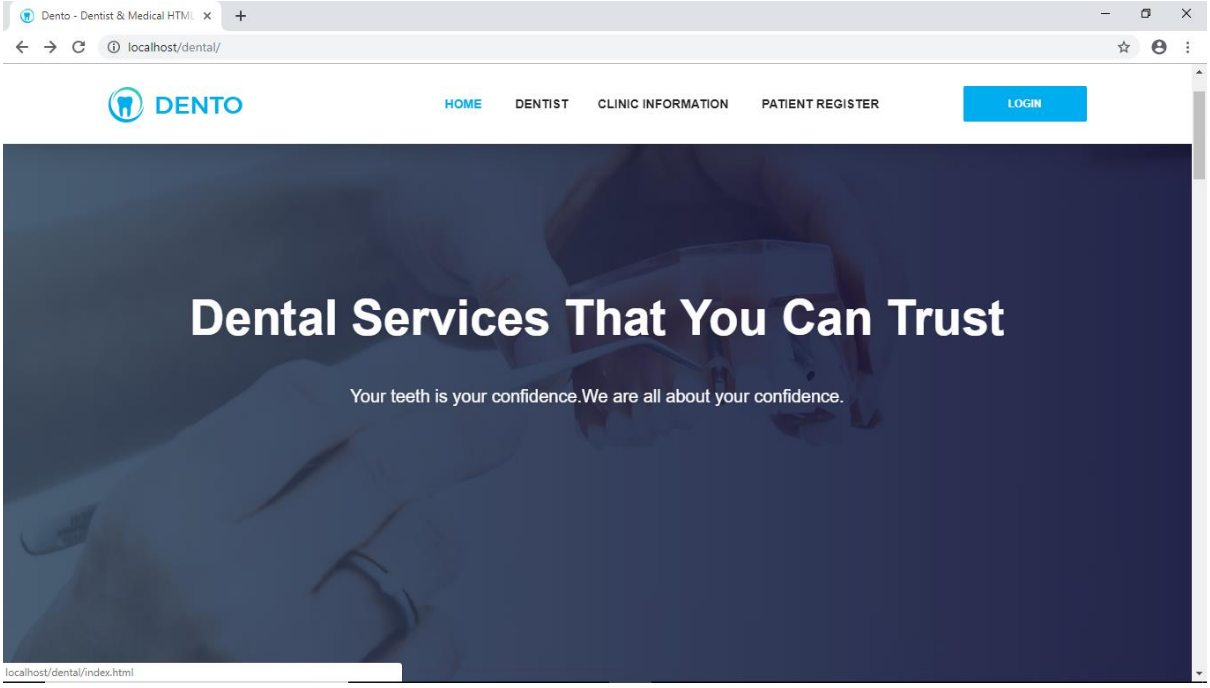
```
}// End of if (empty($errors))

} // End of the main Submit conditional.

?>

<body>


        <div class="limiter">

                <div class="container-login100" style="background-image: url('images/bg-
01.jpg');">

                        <div class="wrap-login100 p-t-30 p-b-50">

                                <span class="login100-form-title p-b-41">

                                        Create Account

                                </span>


                                <form class="login100-form validate-form p-b-33 p-t-5"
action="signup.php" method="post" class="form">

                                        <div class="wrap-input100 validate-input" data-
validate = "Enter username">

                                                <input class="input100" type="name"
name="name" placeholder="Name"

                                                value="<?php if (isset($_POST['fname'])) echo
$_POST['fname']; ?>"></p>

                                                <span class="focus-input100" data-
placeholder="&#xe82a;"></span>

                                        </div>




                                        <div class="wrap-input100 validate-input" data-
validate = "Enter username">

                                                <input class="input100" type="phone"
name="phone" placeholder="Phone" pattern="[789][0-9]{9}"

                                                value="<?php if (isset($_POST['phone'])) echo
$_POST['phone']; ?>"></p>

                                                <span class="focus-input100" data-
placeholder="&#xe82a;"></span>
```

```
                    </div>


<div class="wrap-input100 validate-input" data-validate = "Enter username">
                        <input class="input100" type="age"
name="age" placeholder="Age"
                        value="<?php if (isset($_POST['age'])) echo
$_POST['age']; ?>"></p>
                        <span class="focus-input100" data-
placeholder="&#xe82a;"></span>
                    </div>



<div class="wrap-input100 validate-input" data-validate = "Enter username">
                        <input class="input100" type="address"
name="address" placeholder="Address"
                        value="<?php if (isset($_POST['address']))
echo $_POST['address']; ?>"></p>
                        <span class="focus-input100" data-
placeholder="&#xe82a;"></span>
                    </div>



<div class="wrap-input100 validate-input" data-validate = "Enter username">
                        <input class="input100" type="email"
name="email" placeholder="Email"
                        value="<?php if (isset($_POST['email'])) echo
$_POST['email']; ?>" > </p>
                        <span class="focus-input100" data-
placeholder="&#xe82a;"></span>
                    </div>


<div class="wrap-input100 validate-input" data-validate = "Enter username">
                        <input class="input100" type="password"
name="psword1" placeholder="Password"
```

```
                        value="<?php if (isset($_POST['psword1']))
echo $_POST['psword1']; ?>" ></p>
                        <span class="focus-input100" data-
placeholder="&#xe82a;"></span>
                    </div>
<div class="wrap-input100 validate-input" data-validate="Enter password">
                        <input class="input100" type="password"
name="psword2" placeholder="password"
                        value="<?php if (isset($_POST['psword2']))
echo $_POST['psword2']; ?>" ></p>
                        <span class="focus-input100" data-
placeholder="&#xe80f;"></span>
                    </div>


                    <div class="container-login100-form-btn m-t-32">
                        <button class="login100-form-btn" input
id="submit">
                            Register


                        </button>



                        </div>
<br>&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp
&nbsp&nbsp&nbsp&nbsp&nbsp<p23> Already our patient? <a href="login.php">click to
login </a> </p23>
                </form>
            </div>
        </div>
    </div>
    </form>
</body>
</html>
```

5.6 SCREENSHOTS

HOMPAGE



Dento - Dentist & Medical HTML

localhost/dental/

DENTO

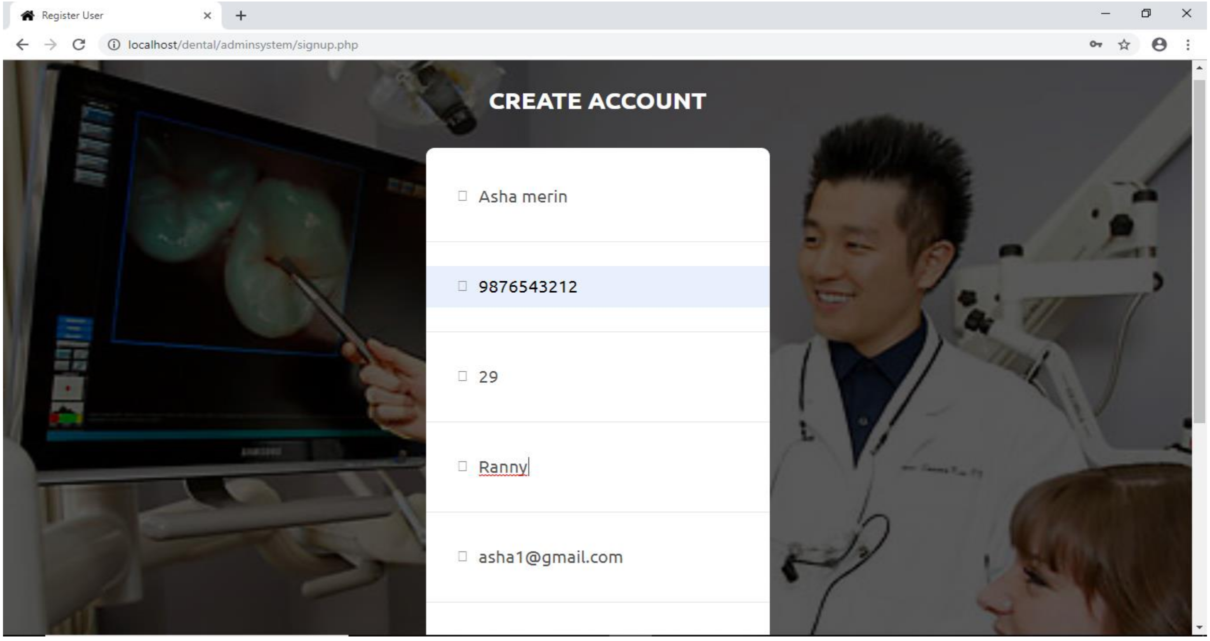
HOME DENTIST CLINIC INFORMATION PATIENT REGISTER LOGIN

Dental Services That You Can Trust

Your teeth is your confidence.We are all about your confidence.

localhost/dental/index.html

SIGNUP



Register User

localhost/dental/adminsystem/signup.php

CREATE ACCOUNT

Asha merin

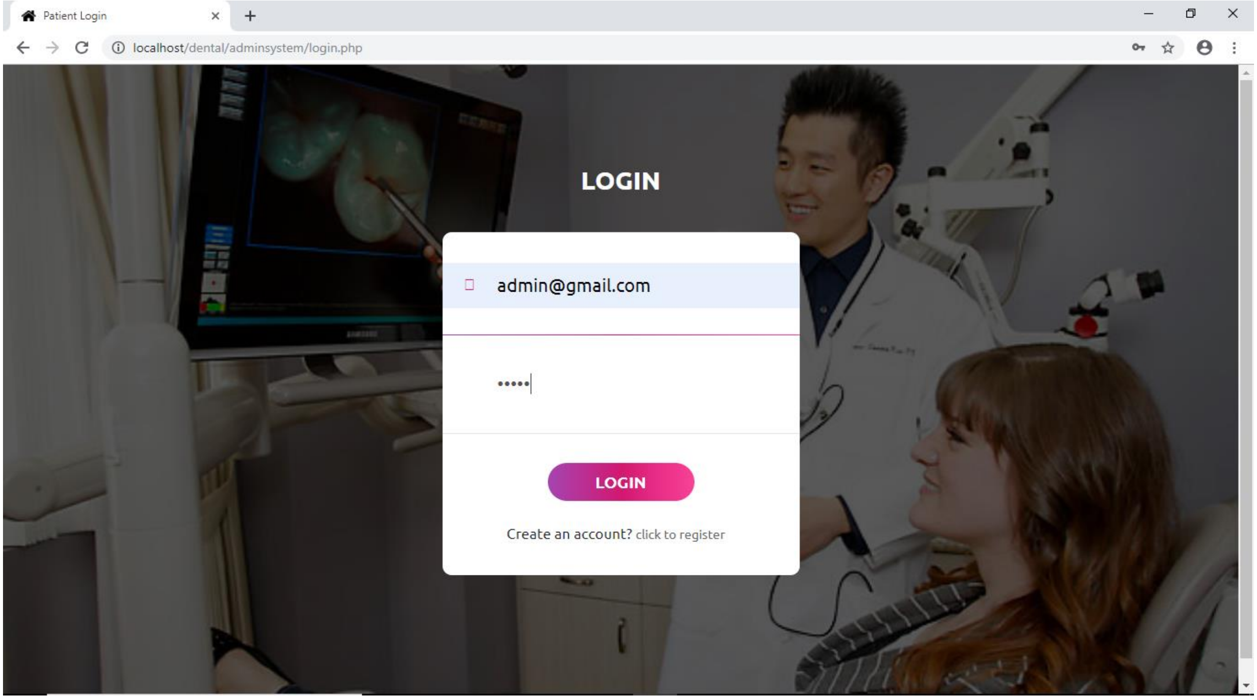
9876543212

29

Ranny

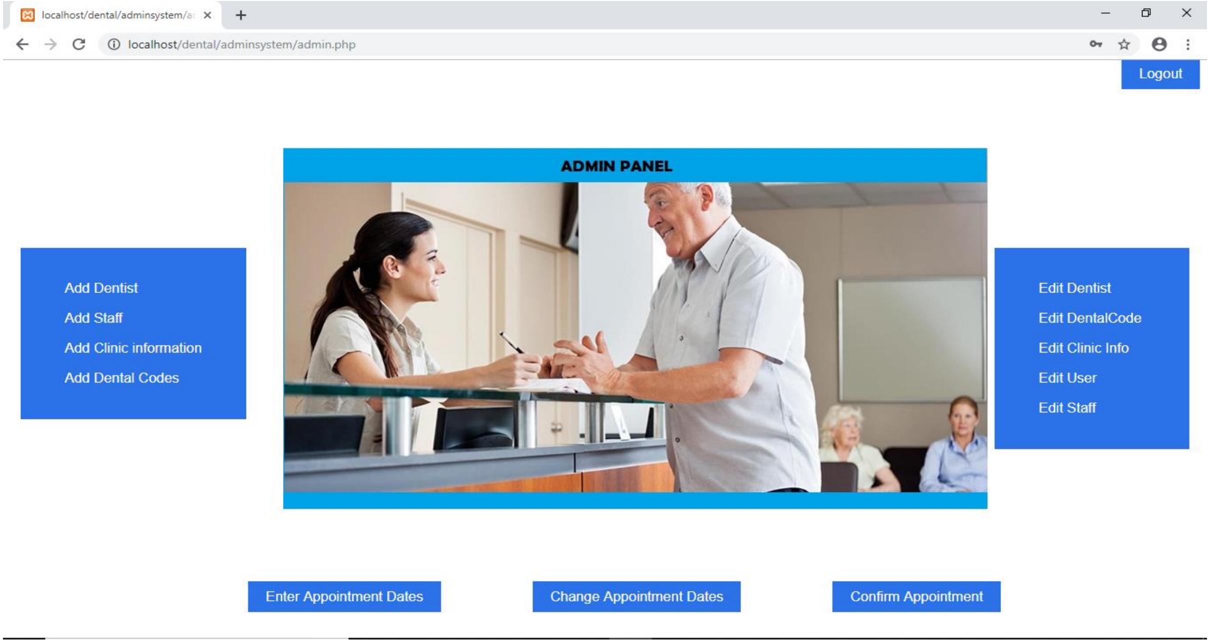
asha1@gmail.com

ADMIN LOGIN

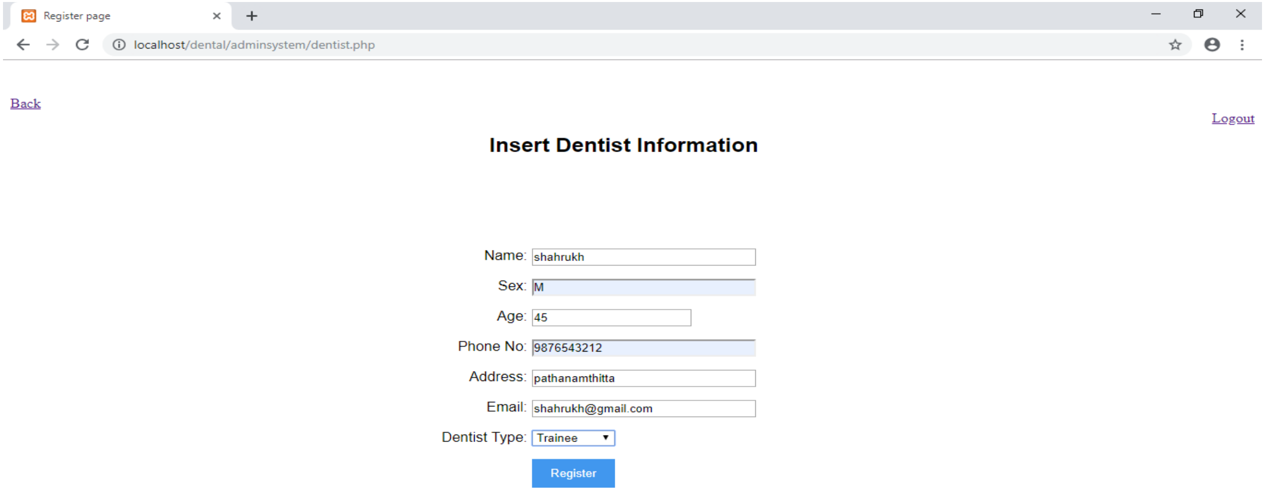


Patient Login
localhost/dental/adminsystem/login.php
LOGIN
admin@gmail.com
•••••
LOGIN
Create an account? click to register

ADMIN PAGE



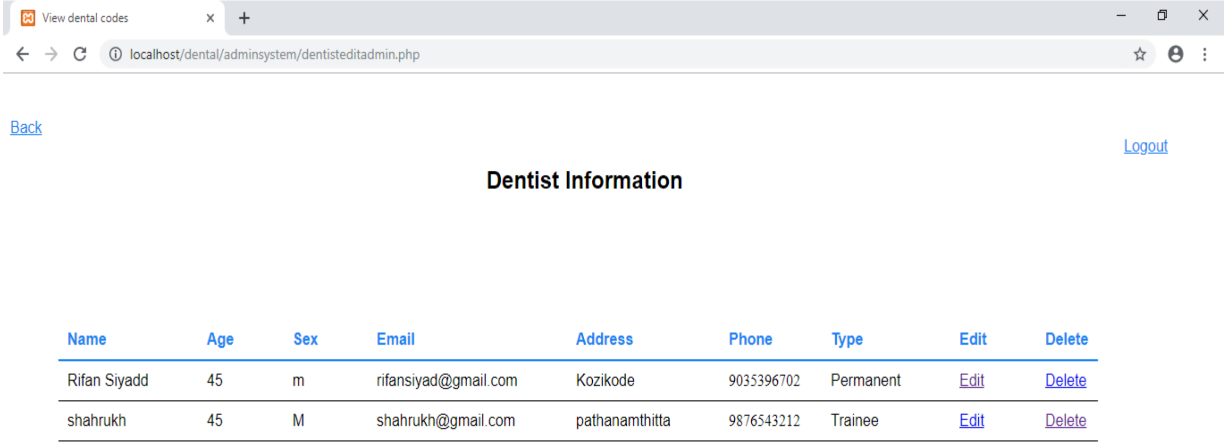
ADD DENTIST



Insert Dentist Information

Name:	shahrukh
Sex:	M
Age:	45
Phone No:	9876543212
Address:	pathanamthitta
Email:	shahrukh@gmail.com
Dentist Type:	Trainee

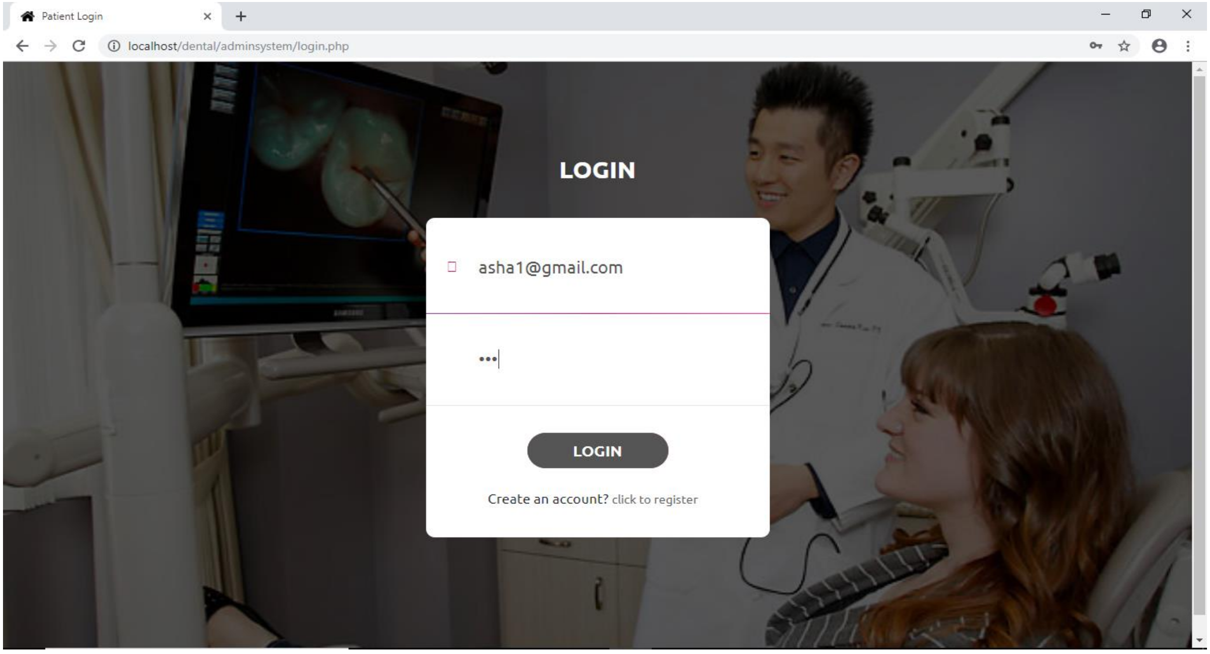
EDIT/DELETE DENTIST



Dentist Information

Name	Age	Sex	Email	Address	Phone	Type	Edit	Delete
Rifan Siyadd	45	m	rifansiyad@gmail.com	Kozikode	9035396702	Permanent	Edit	Delete
shahrukh	45	M	shahrukh@gmail.com	pathanamthitta	9876543212	Trainee	Edit	Delete

PATIENT LOGIN

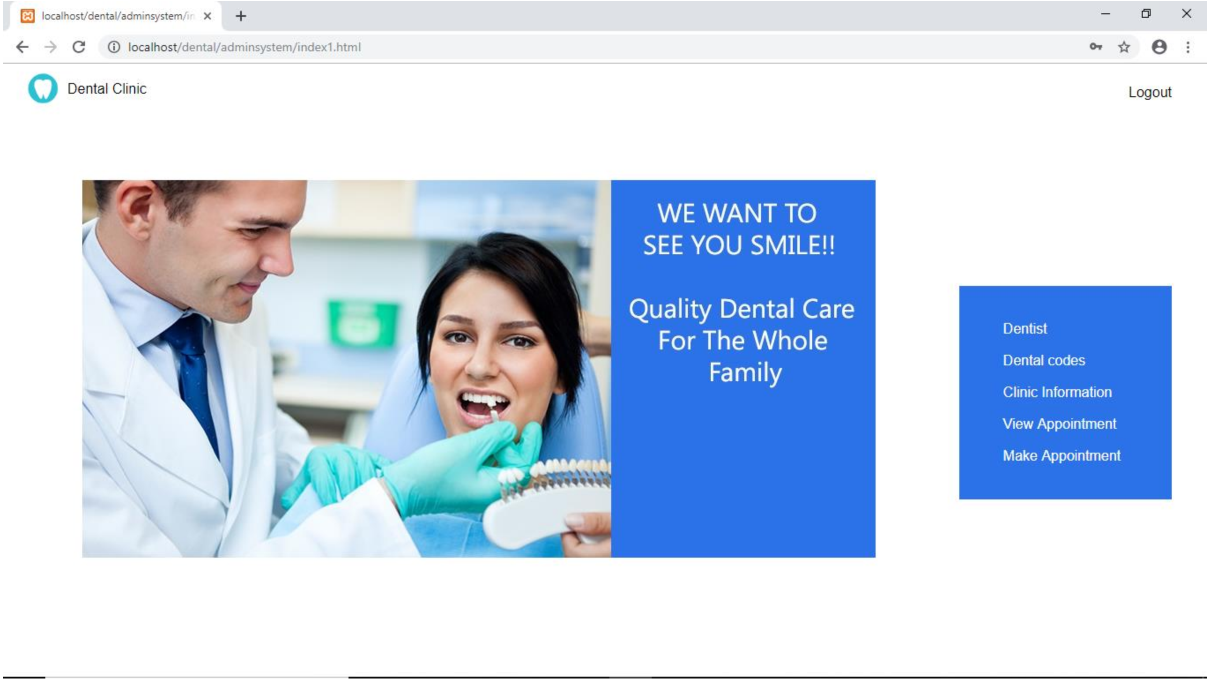


asha1@gmail.com

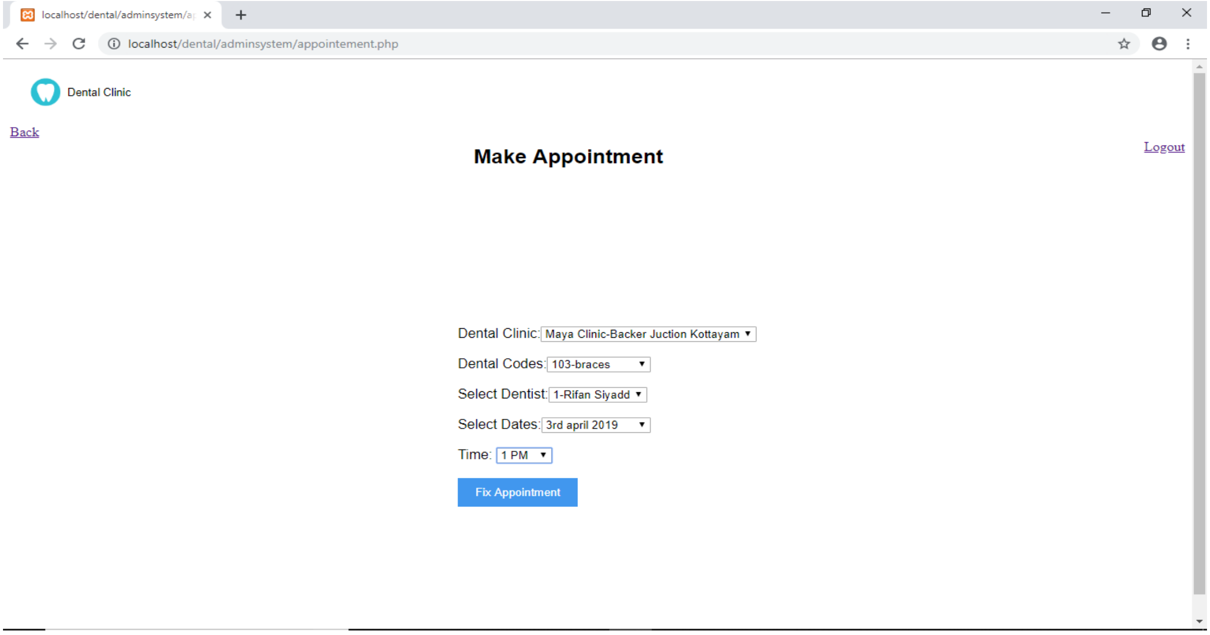
•••

Create an account? click to register

PATIENT PAGE


localhost/dental/adminsystem/in
localhost/dental/adminsystem/index1.html
Dental Clinic
Logout
WE WANT TO SEE YOU SMILE!!
Quality Dental Care For The Whole Family
Dentist
Dental codes
Clinic Information
View Appointment
Make Appointment

ADD APPOINTMENT


localhost/dental/adminsystem/a
localhost/dental/adminsystem/appointement.php
Dental Clinic
Back
Logout
Fix Appointment

Make Appointment

Dental Clinic: Maya Clinic-Backer Juction Kottayam

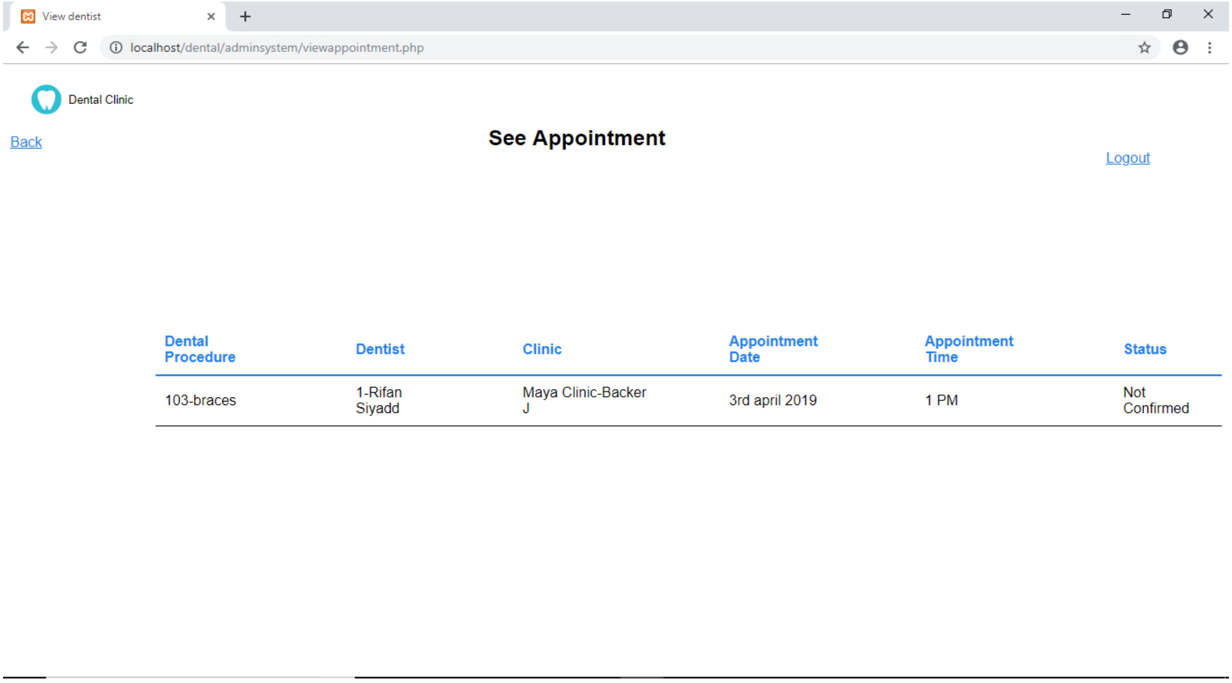
Dental Codes: 103-braces

Select Dentist: 1-Rifan Siyadd

Select Dates: 3rd april 2019

Time: 1 PM

VIEW APPOINTMENT



See Appointment

Dental Procedure	Dentist	Clinic	Appointment Date	Appointment Time	Status
103-braces	1-Rifan Siyadd	Maya Clinic-Backer J	3rd april 2019	1 PM	Not Confirmed

CHAPTER 6

SYSTEM TESTING

6.1 TESTING METHODOLOGIES AND STRATEGIES

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

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

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

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

BLACK BOX TESTING

Black box testing, also called behavioural testing, focuses on the functional requirements of software. This testing approach enables the software engineer to derive the input conditions that will fully exercise all requirements for a program. Black box testing attempts to find the errors like

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access

- Behaviour or performance errors
- Initialization and termination errors

In Black box testing software is exercised over a full range of inputs and outputs are observed for correctness.

WHITE BOX TESTING

White box testing is also called Glass box testing is a test case design control; structure of the procedural design to derive test cases using White box testing method, the software engineer can derive the test cases that guarantee that all independent paths within the module have been exercised at least once. Exercise all logic decisions on their true or false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structure to ensure their validity.

The first level of test is unit testing. The purpose of unit testing is to ensure that each program is fully tested.

6.2 UNIT TESTING

In the unit test case will be testing the separate modules of the software. We will carry out black box testing where each module or component of software is tested individually. We will test the component by passing data through it and we will be monitoring data to find the errors. We will make sure that the component work without any troubles. The test primarily is carried out by the programmer who designed and implemented the module. Lead tester is carried out by the programmer who test the modules to finalize the testing.

6.3 INTEGRATION TESTING

In the Integration testing we will combine the different tested modules and we will test the bundle of module. This is to ensure that the entire modules are working correctly in conjunction with the other modules. Data can be lost across any interface; one module can have adverse effect on another. Sub function when combined, may not produce the desired major function. Integration testing is a systematic testing for conducting test to uncover errors associated within the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here correction is difficult because vast expense of the entire program complicates the isolation of causes.

6.4 USER ACCEPTANCE TESTING

System validation checks for equality of the software in both simulated and live environments. First, the software goes through a phase, in which errors and failures based on simulated user requirements are verified and studies. This is called alpha testing.

CHAPTER 7

SYSTEM IMPLEMENTATION

7.1. INTRODUCTION

Implementation is the stage in the project where the theoretical, design I turned into a working system and is giving confidence on the new system for the users, which it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation, of change over methods. Apart from planning major task of preparing the implementation are education and training of users the more complex system being implemented, the more involved will be the system analysis and the design effort required just for implementation.

An implementation co-ordination committee based on policies of individual organization has been appointed. The implementation process begins with preparing a plan for them implementation of the system. According to this plan, the activities are to be carried out, discussions made regarding the equipment and the additional equipment and resources and the addition equipment has to acquire to implement the new system.

Implementation is the final and more important phase. The system can be implemented only after through testing is done and if it found to work according to the specification. This method also offers the greatest security since his old system can take over if the errors are found or inability to handle certain type of transactions while using the new system.

Implementation involves careful planning to avoid any unwelcome consequences. The effort spends on developing any system results in success only when the system implemented properly.

System implementation involves actual installation, evaluation of the installation, organizational impact and finally the equality assurance.

The implementation plan consists of:

- Testing the developed system with the sample data
- Detection and correction of errors
- Making necessary changes in the system
- Checking it with the existing system

CHAPTER 8

FUTURE ENHANCEMENT OF THE PROJECT

8.1 FUTURE ENHANCEMENT

As all the projects have their future expansion. The entire project may be changed in future by the other person. We also have our future expansion and we can add new features as and when we require. There is flexibility in all modules. In future somebody might buy this project.

Also, some might add additional features to these projects and can expand these projects. But these is yet created in any of the college so we decided to make these projects. Also, in future some industries might be interested in buying this project. And also, we will be in future we will try to make these projects more interesting by adding some futures in it. Its cost is under the budget and make within given time period. It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy the entire requirement. Scope of this document is to put down the requirements, clearly identifying the information needed by the user, the source of the information and outputs expected from the system.

CHAPTER 9

CONCLUSION

9.1. CONCLUSION

Dental Clinic is a Graphical User Interface based website. The main objective of this website is to make it easy for the clinic to perform their daily tasks. The product will take care of Registrations, Appointments, and Dental services etc. It also gives an abstract idea about the clinic and its atmosphere.

Dental clinic is a website that manages registrations, appointment details, etc. It makes the job of the dentists and patients easy and efficient. A patient can make an appointment with the doctors of their choice for different services. Normally it was done through documentations, it requires a lot of time and man power by dentists and patients. This project implementing a real time system, through which patients can register his/her details, view appointments and communicate with corresponding coordinators.

The project has been developed in **PHP** as front end and **MYSQL Server** as back end which develop to help powerful software.

.

CHAPTER 10

APPENDIX

10.1 LIST OF TABLES

10.2 LIST OF FIGURES

\

CHAPTER 11

BIBLIOGRAPHY

11.1 BIBLIOGRPHY

BOOKS/REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, ʺSystem Analysis and Designʺ, 2009.
- Roger S Pressman, ʺSoftware Engineeringʺ, 1994.
- Pankaj Jalote, ʺSoftware engineering: a precise approachʺ, 2006.
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.codemaster.com
- www.stackoverflow.com
- www.w3layouts.in