INTRODUCTION

1.1ABOUT THE PROJECT

Our project is created to promote children's better health. For this, we provide healthy tips, diet tips, diet chart etc. Immunizations are crucial for a child's healthy development. So, we provide vaccine details and their notifications.

Parents can register their children in this that may be infant, toddler or adult. Parents can choose the plans through it. If the parent needs a dietitian for their children, they can select and pay for dietitians. User can easily communicate with dietitian through chat and can rate for them.

Kids can enjoy themselves with activities like stories, videos, picture identifications and lullabies with the help of their parents.

This project is developed using HTML, CSS, Bootstrap, Javascripts as front-end design tool. Python as server-side scripting language and sqlite3 as the backend.

Objectives

Main aim of this project is to help kids to bring healthy growth. Parents can know about the necessary details for their child's better growth. Through this web application we aim to make them happy and improving child's better health through diet details, dietitian and entertainment and brain development activities such as stories, videos, lullabies and picture identification.

The objectives includes:

- The web application can be useful for infant babies to adult children.
- Qualified dietitian can register.
- Users can receive vaccine details and notification.
- Children can entertain through activities.

Scope of the project

The mission in grow with us is to prevent both undernutrition and overnutrition to maintain optimal nutritional status of the population. A user from everywhere can easily access the application. Online accessing helps a user to register from anywhere. A user can access all the functionality after filling a simple registration form. Searching details of vaccination and online booking of dietitian etc. are enabled. If any problem occurs, there is a facility to post complain to the administrator. All the information are controlled by the administrator. All the booking details is available in the user profile.

1.2STATEMENT OF THE PROBLEM

MODULE DESCRIPTION

Grow with us consist of 3 modules. They are admin, user.

1. Admin

- Login
- View dietitian request
- Add and remove dietitian
- View dietitian services
- View user feedback
- Respond to the complaints
- View vaccine details
- Control notification of vaccination
- Manage activities

2. User

- Login
- Update profile
- View registration details
- View plans
- View dietitian’s payment plans
- Rate and chat dietitian
- View and write feedback
- View notifications of vaccination
- View healthy tips and diet tips
- View diet chart
- View Activities
- View details about the vaccines

3. Dietitian

- Login
- View details of dietitian
- Update profile
- Update payment plans
- View request to the user
- Respond user’s request
- Chat with user
- Consultation
- View feedback
- View user’s post
- View healthy tips

SYSTEM ANALYSIS

2.1 INTRODUCTION

Protecting and improving the health of children is of fundamental importance. Poor diets in early childhood can lead to deficiencies in essential vitamins and nutrients. For children, the right to health is vital because they are vulnerable beings, more at risk to illness and health complications. Among other encouraging statistics, the number of children dying before the age 5 was halved from 2000 to 2017, and more mothers and children are surviving today than ever before. However, a great deal of work remains to further improve the health outcomes for children. Children must also be given a stable environment in which to thrive, including good health and nutrition, Protection from threats and access to opportunities to learn and grow. So, children with a good dietitians can improve their better health. Investing in children is one of the most important things a society can do to build better future.

This project is developed using HTML, CSS, Bootstrap, and JavaScript as front end and sqlite3 as the backend.

- Identifying the drawback of existing system.
- Identify the need for conversion.
- Perform feasibility study.
- Identify hardware, software and database requirements.

2.2 DESCRIPTION OF EXISTING SYSTEM

Nowadays there are several child healthcare applications are available. But doesn’t get all health-related process in only one application. Existing system consist of only the dietitian details such as, contact number, rating etc. Registered users can only view the details about anything including into to the application. There is no security in existing system. It is difficult to find the better dietitian in a short period of time. If user doesn’t have money, then the application is useless. All available applications can’t get needs of whole health growth into one application.

2.3 LIMITATIONS OF EXISTING SYSTEM

- Needs continues searches in internet.
- More human effort.
- Lack of security.
- It is very tedious and time consuming.

2.4 PROPOSED SYSTEM

Through this project, we are planning to get the whole process of healthy growth for children from infant to adult in one. People could also be able to know about the dietitian and their rating. Our application provides a platform where the users could communicate with the dietitian, existing user and seek help. It provides vaccination details and notifications of vaccination. Also includes the healthy tips, diet chart and activities like stories, lullaby, videos etc. Beside this, the interface of the web application changes according to the age group changes.

2.5 ADVANTAGES AND FEATURES OF PROPOSED SYSTEM

Features and advantages of proposed system are: -

- Minimize human efforts and time saving.
- User can send suggestions and complaints.
- Attractive user interface.
- More secured.
- User friendly.

2.6 FEASIBILITY STUDY

The prime objective of feasibility study is to ensure that the problem is worth to be solved. At the stage a cost benefit analysis is performed to assertion that the benefit from the system will over rule the cost association with the whole analysis, design and development of the new system. An important outcome of the preliminary investigation determining whether the system required is feasible.

Steps in feasibility analysis

Feasibility Analysis involves eight steps

- Form a project team and appoint a project leader.
- Prepare a system flow chart.
- Enumerate potential candidate systems.
- Describe and identify characteristics of candidate systems.
- Describe and evaluate performance and cost effectiveness of each candidate systems.
- Weight system performance and cost data.
- Select the best candidate system.
- Prepare and report final project directive and management.

The proposed system is tested in all three aspects of feasibility

- Technical Feasibility study
- Operational Feasibility
- Financial and Economic Feasibility study
- Behavioural Feasibility

2.6.1Technical Feasibility

Technical feasibility center on existing system and to what extent it can support proposed modifications. It involves financial enhancement. This evaluation determines whether the technology needed for the proposed system is available or not. This is concerned with specifying satisfy the user requirements. The technical needs of the system may include front - end and back - end selection. An important issue for the development of a project is the selection of the suitable front - end and back - end. Based on some aspects, we select the most suitable platform that suits the needs of the organization, so in our system technically feasible.

2.6.2 Operational Feasibility

There is no difficulty in implementing the system. The proposed system is effective & user friendly. The user of the system must be completely unaware of the internal working of the system so that the users will not face any problem running the system. The system thus reduces the responsive time of computer thereby; the system is found to be operationally feasible. Operational feasibility is a measure of how people are able to work with the system.in our system consists of admin, ground admin, shop, clubs, users etc. So, all these users can easily to work with our portal. Since the proposed system fulfil all the requirements that it has supposed to do therefore, we can say that the system is operationally feasible.

2.6.3 Economical Feasibility

Economic and Financial analysis is used for evaluating the effectiveness of the system. This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditure must be justified. Comparison between the benefits and savings expected from the candidate system with cost incurred is done. If benefits outweigh cost, then decision will be to design and implement system. Otherwise, alterations will have to be made to the proposed system. The proposed system is economically feasible.

2.6.4 Behavioural Feasibility

The behavioural feasibility depends upon whether the system performed in the expected way or not. Feasibility study is a test of system proposal according to it workability, impact on feasibility study provides a useful starting point for full analysis. The behavioural feasibility depends upon whether the system performed in the expected way or not. Feasibility study is a test of system proposal according to it workability, impact on organization, ability to meet the user’s need and effective use of resources. However, a feasibility study provides a useful starting point for full analysis. The main problem faced during development of a new system is getting acceptance from the user. People are inherently resistant to changes and computers have been known to facilitate changes. It is mainly related to human organizational and political aspects.

The points to be considered are:

- What changes will be bought with the system?
- What new skills will be required? Do the existing staff members have these skills? If not can they be trained due course of time?

This feasibility study is carried out by small group of people who are familiar with information testing techniques, who understand the parts of the problems of existing system that are relevant to the project and are skilled in analysis and design process.

SYSTEM REQUIRE

2.7 SYSTEM SPECIFICATION

2.7.1 Minimum Software Requirements

Language	:	Python
IDE	:	Visual Studio
Framework	:	Django
Frontend	:	HTML, CSS, Bootstrap, JavaScript
Backend	:	Sqlite3
Web Browser	:	Google Chrome, Mozilla Firefox
Operating System	:	Windows 11

2.7.2 Minimum Hardware Requirements

Processor	:	11th Gen Intel® Core™
RAM	:	8GB
Hard Disk free space	:	10 GB
Display	:	1920 x 1080
Media	:	USB

2.7.3 Python

Python is a popular programming language. It was created by Guido van Rossum and released in 1991. It can be used on a server to create web applications and can be used alongside software to create workflows. It can connect to database systems and can also read and modify files. It can be used to handle big data and perform complex mathematics and can be used for rapid prototyping, or for production-ready software development. It works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). It has a simple syntax similar to the English language that allows developers to write programs with fewer lines than some other programming languages. It runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick. It can be treated in a procedural way, an object-oriented way or a functional way.

2.7.4 Visual Studio

Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs including websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code. Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, Typescript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past. The most basic edition of Visual Studio, the Community edition, is available free of charge.

2.7.5 DJANGO

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source. It was designed to help developers take applications from concept to completion as quickly as possible. It takes security seriously and helps developers avoid many common security mistakes.

2.7.6 HTML

HTML stands for Hyper Text Mark-up Language, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late1999. Though HTML 4.01 version is widely used but currently we are having HTML5 version which is an extension to HTML 4.01, and this version was published in2012. Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

2.7.7 CSS

CSS is used to control the style of a web document in a simple and easy way. CSS is the acronym for "Cascading Style Sheet". CSS handles the look and feel part of a web page. Bootstrap makes use of certain HTML elements and CSS properties that require the use of the HTML5 doctype. Bootstrap includes a responsive, mobile first UID grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes predefined classes for easy layout options, as well as powerful mixings for generating more semantic layouts. Grid systems are used for creating page layouts through a series of rows and columns that house your content.

2.7.8 JavaScript

JavaScript is a scripting language. A scripting language is easy and fast to learn. A scripting language is interpreted in run-time. It is not compiled like other languages as C++, C sharp, VB.net etc. JavaScript is a client-side language and it runs on a client browser. Netscape developed it and because of its simplicity it is one of the most known scripting languages. However, JavaScript can also be used on the server side. JavaScript can be used on all most known browsers. It can be easily used to interact with HTML elements. You can validate text fields, disable buttons, validate forms, or change the background color of your page. All this is possible with JavaScript. Like each programming language, it contains variables, arrays, functions, operators, objects and much more which can help you to create better script for your pages. On the server side you can use JavaScript for example to manage database entry. JavaScript code can be inserted directly in the HTML or you can place it in a separate file with the .js extension and link the webpage with the .js file.

2.7.9 SQLite3

SQLite is a database engine written in the C programming language. It is not a standalone app, rather, it is library that software developers embed in their apps. As such, it belongs to the family of embedded databases. It is the most widely deployed database engine, as it is used by several of the top web browsers, operating systems, mobile phones and other embedded system. Many programming languages have bindings to the SQLite library. It generally follows Postgre SQL syntax, but does not enforce type checking by default. SQLite was designed to allow the program to be operated without installing a database management system or requiring a database administrator, Unlike client server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, a linker integrates the SQLite library - statically or dynamically - into an application program which uses SQLite's functionality through simple function calls, reducing latency in database operations; for simple queries with little concurrency, SQLite performance profits from avoiding the overhead of inter process communication. Due to the server less design, SQLite applications require less configuration than client-server databases.

SYSEM DESIGN

3.1 INTRODUCTION TO SYSTEM DESIGN

System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. It implies a systematic approach to the design of a system. It may take a bottom-up or top-down approach.

- System design covers the following:
- Reviews the current physical system,
- Prepares output specifications,
- Prepares input specifications,
- Prepares edit, security and control specifications,
- Specifies the implementation plan,
- Prepares a logical design walk through of the information flow, output, input, control and implementation plan.

Design of a system can be defined as the process of applying various techniques and principles for the purpose of defining a device, a process or system is sufficient details to permit its physical realization. Thus, system design is a solution a "how to" approach to the creation of a new system. This important phase provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. The data design transforms the information domain model created during analysis into the data structure that will be required to implement the software. The architectural design defines the relationship among major structural components into a procedural detail necessary for implementing the system recommended in the feasibility study. The data design transforms the information domain model created during analysis into the data structure that will be required to implement the software. The architectural design defines the relationship among major structural components into a procedural description of the software. Source code is generated and testing is conducted to integrate and validate the software. Project management point of view software design is conducted in two into data and software architecture.

There are two levels of the system design:

- Logical design
- Physical design

Logical Design

Logical design aims at establishing the requirements of the users, which the new system must satisfy. In the stage, the system analyst has to identify the relationship between the various items of the data and the grouping if items of data together into records. This is known as logical data structure, which is required to produce outputs, which users require. Once this is agreed, the logical design can be turned into a physical system with more detailed design.

- Input design
- Output design
- Database design

Physical Design

Physical design takes the logical design blue print and produces the program specification. Physical design and user interface for a selected hardware and software.

Input Design

Input design is a part of overall system is design, which requires very careful attention. If data going into the system is incorrect, then processing and output will magnify these errors. Thus the designer has a number of clear in the objectives in the different stages of input design

- To produce a cost-effective method of input.
- To archive the highest possible level of accuracy.
- To ensure that input is acceptable to and understood by the user.

Output Design

- At the beginning of output design various type of outputs (external, internal, operational and turnaround) are defined. Then the format, content, location, frequency, volume and sequence of the output are specified. The content of the output must be defined in details. The system analyst has two specific objectives at these stages.
- To interpret and communicate the result of the computer part of a system to users in a form that they can understand and which meets their requirements.
- To communicate the output design specification to programmers in a way, this is unambiguous, comprehensive and capable of being translated into programming languages.

Database Design

Database design is the process of converting user-oriented inputs to a computer-based format. The database design phase is used to design the input with in the predefined guidelines. Inaccurate input data are the most common cause of errors in the data processing. Errors entered by data entry operators can be controlled by input design. Input design consist of developing specifications and procedures for data preparation and data validation. Database system consists of an important part of every project. The management of data involves both definition of structure storage of information and provision of mechanism for manipulation of information. The database designs provide more safety for the information stored, despite system crashes or attempts of unauthorized access to the database. Database files are the key source of information into the system. It is the process of designing database files, which are the key source of information to the system. The files should be properly designed and planned for collection, accumulation, editing and retrieving the required information. The organization of data in database aims to achieve three major objectives:

- Data integration
- Data integrity
- Data independency

3.2 STRUCTURED CHART

Structured chart shows variable pass between variable modules in a computer. Each task can be associated with structured chart representation. For large system several levels of structured will be needed to reflect the number and complexity of the module in system irrespective of whether multiprocessing or multitasking is in use. The relationship between are shown by call, the existence and direction of the call indicate that a module has means to call other module and at runtime the module a may call other module zero or more times. Typically the structured contain a hierarchy of modules which is used to show how one module will call another. As module calls normally involves the passage and return of parameter, the structured and chart these with couples of data and control which are provided that the passage of return of item of data and or sum value that control operation the recipient module.

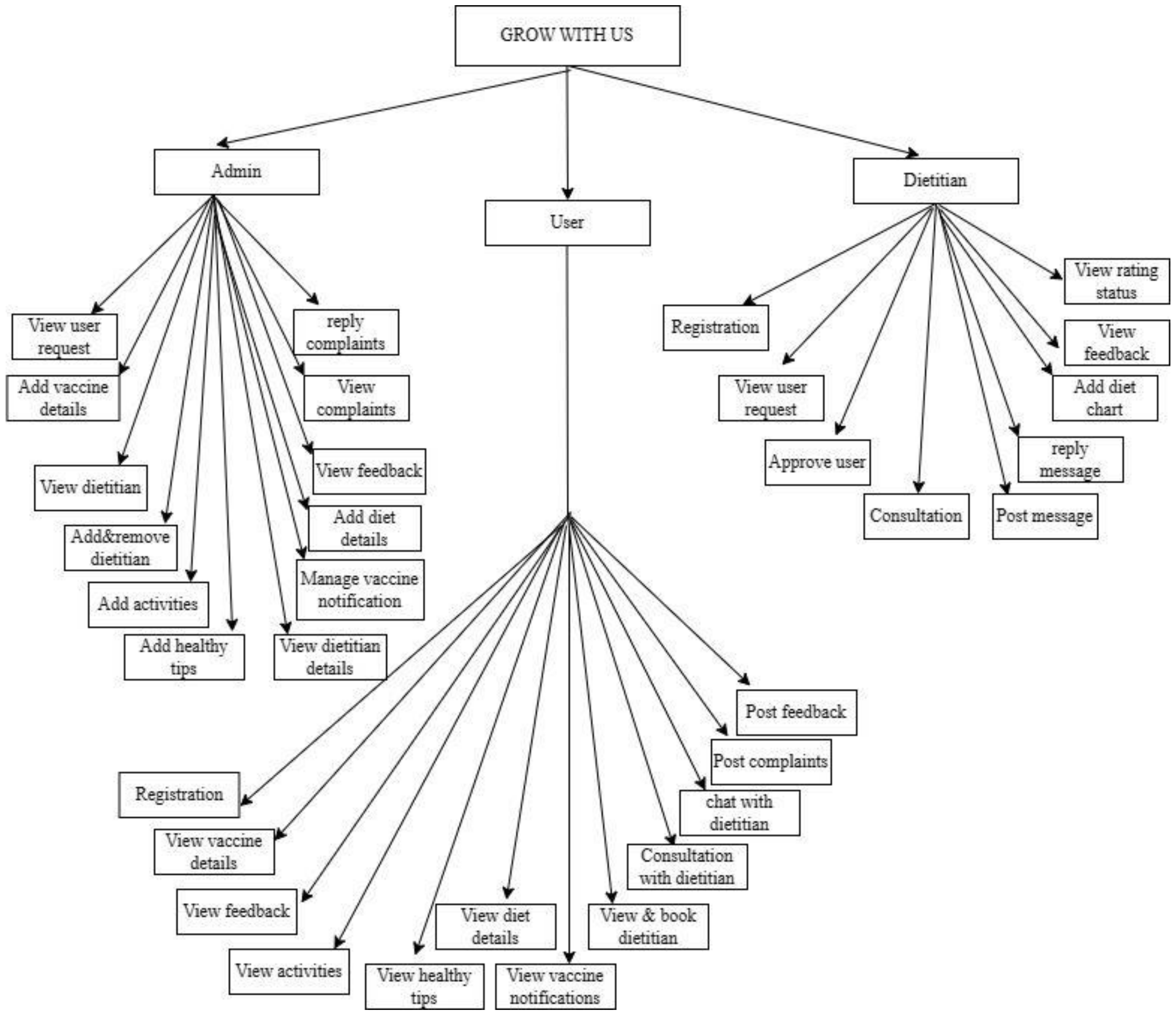
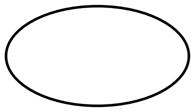


Fig 1: Structure chart

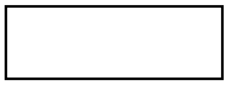
3.3 DATA FLOW DIAGRAM (DFD)

A DFD is a network that describes the flow of data throughout a system, data stores, and the process that change or transform data flows. Data Flow Diagrams are also known as Data Flow Graphs. DFDs are commonly used during the problem analysis stage. They are useful in understanding a system and can be effectively used for partitioning during analysis. The DFD network is a formal, logical abstract of a system that may have many possible physical configurations. This reason a set of symbols that do not imply a physical form are used to represent data source, data flows, data transformations and data storage. The basic element of DFD is:

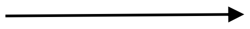
• **Process**: A process in the data flow diagram used to represent some amount of work being performed on data. The oval or bubble shape is used to represent the processes in a data flow diagram.


OVAL

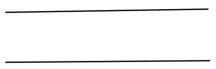
• **External Entity**: This represents any outside agency, which interact with the system. It represents the source and destination of the data for the system under consideration. The rectangle shape is used to represent the external entity in the data flow diagram.


RECTANGLE

• **Dataflow:** The data flow portrays an interface among different components in a DFD. It represents flow of data between a process and data store. The arrow shape is used to represent the data flow in a data flow diagram.


ARROW

• **Data store**: A data store is a place for holding information within the system. Here data is stored or referenced by a process in the system.


OPEN ENDED RECTANGLE

Level 0: Context Level DFD

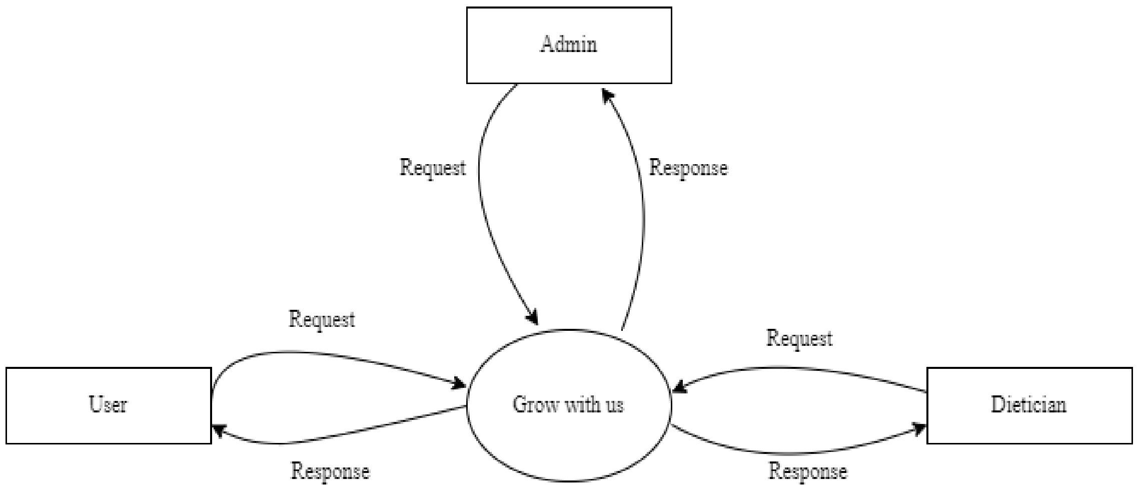


Fig 2: Level 0 dfd

Level 1 DFD

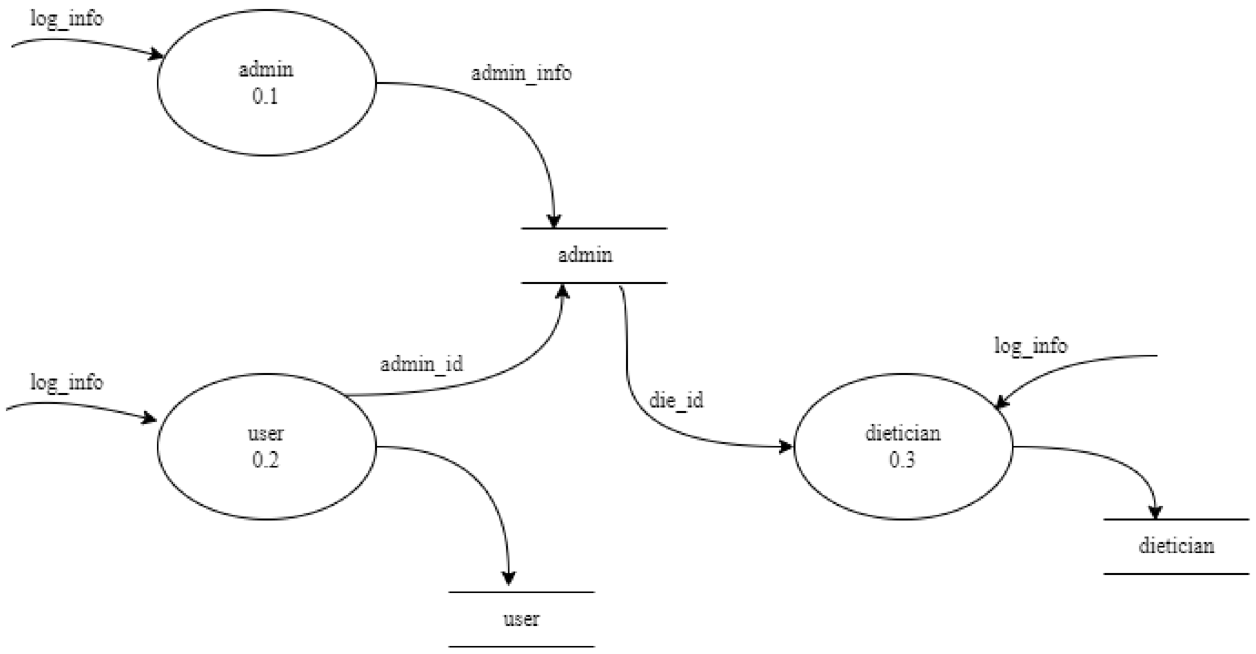


Fig 3: Level 1 DFD

Level 2 DFD-ADMIN

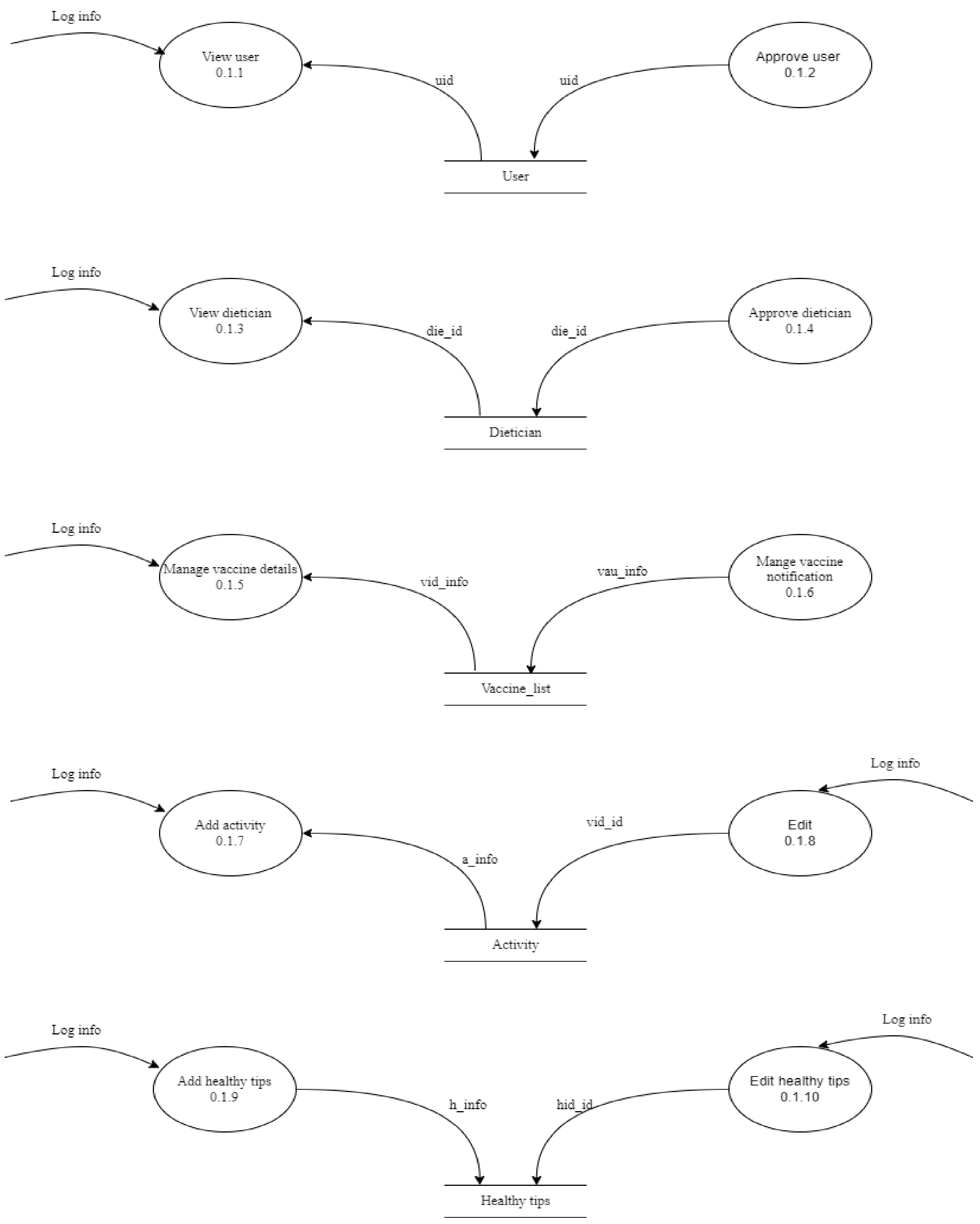


Fig 4: Level 2 dfd for Admin

Level 2 DFD-USER

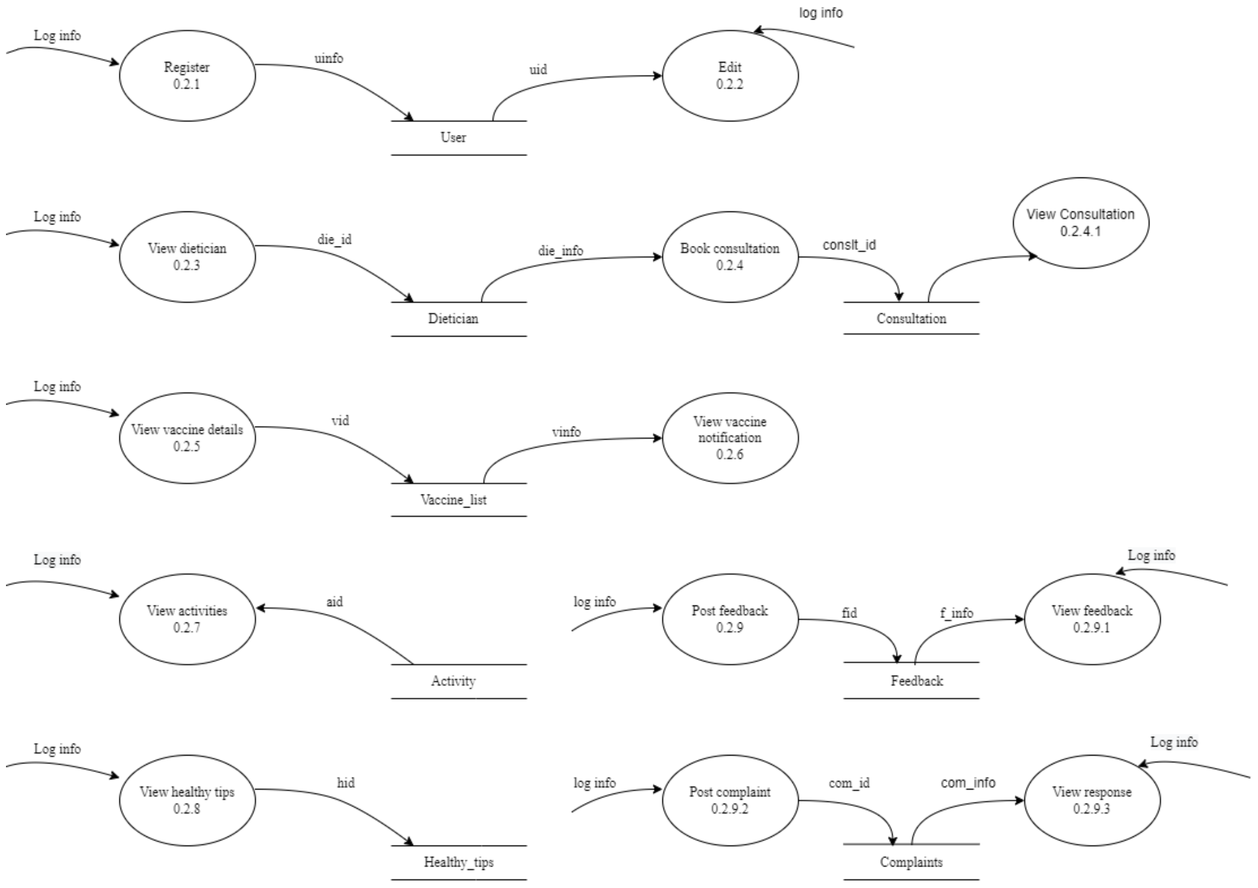


Fig 5: Level 2 dfd for User

Level 2 DFD_DIETICIAN

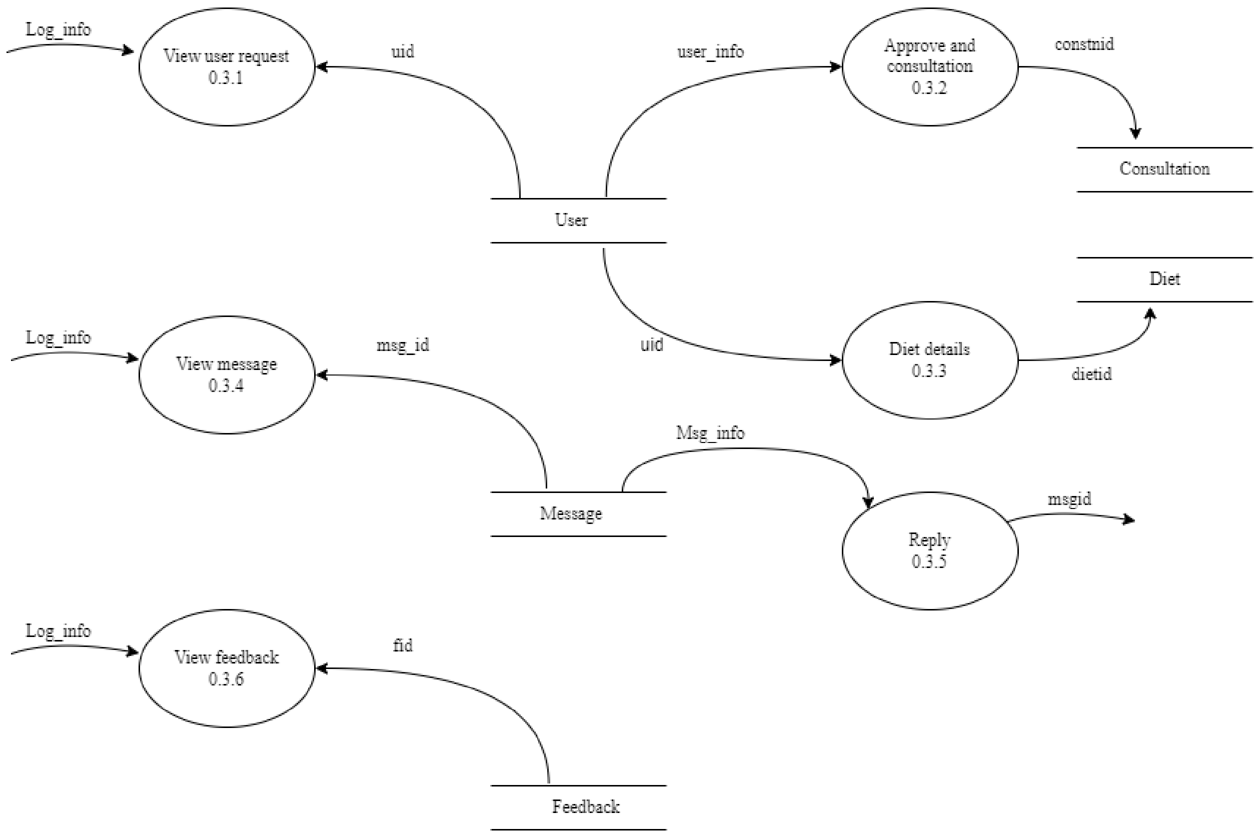


Fig 6: Level 2 dfd for Dietician

3.4 ER DIAGRAM

An Entity-Relationship (ER) diagram is a specialized graphic that illustrate the interrelationship between entities in the database. Boxes are commonly used to represent entity. Diamonds are normally used to represent relationship and ovals are used to represent attributes. An entity is a piece of data and object or concept about which data is stored. A relationship is how the data is shared between the entities.

Classifying Relationships

Relationships are classified by their degree, connectivity, cardinality, direction, type and existence.

Degree of a Relationship

The degree of a relationship is the number of entities associated with the relationship. The n-array relationship is the general form for degree n. Special cases are binary, ternary where the degree is 2 and 3 respectively.

Connectivity and Cardinality

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one or many". The cardinality of a relationship is the actual number of related occurrences for each of the two entities. The basic type of connectivity of a relation is: one-to-one, one-to-many, many-to-many.

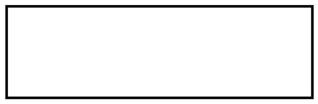
• A one-to-one is when at most one instance of an entity, entity A is associated with one instances of entity B.

• A one-to-many is when for an instance of an entity A, there are zero or many instances of entity B, but for one instances of the entity B, there is one instances of entity A.

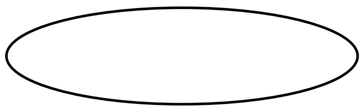
• A many-to-many relationship, sometimes call non-specific, is when for one instance of entity A.

SYMBOLS USED IN ER DIAGRAM

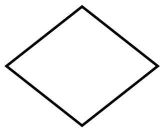
Entity :



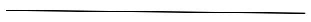
Attribute :



Relationship :



Lines :



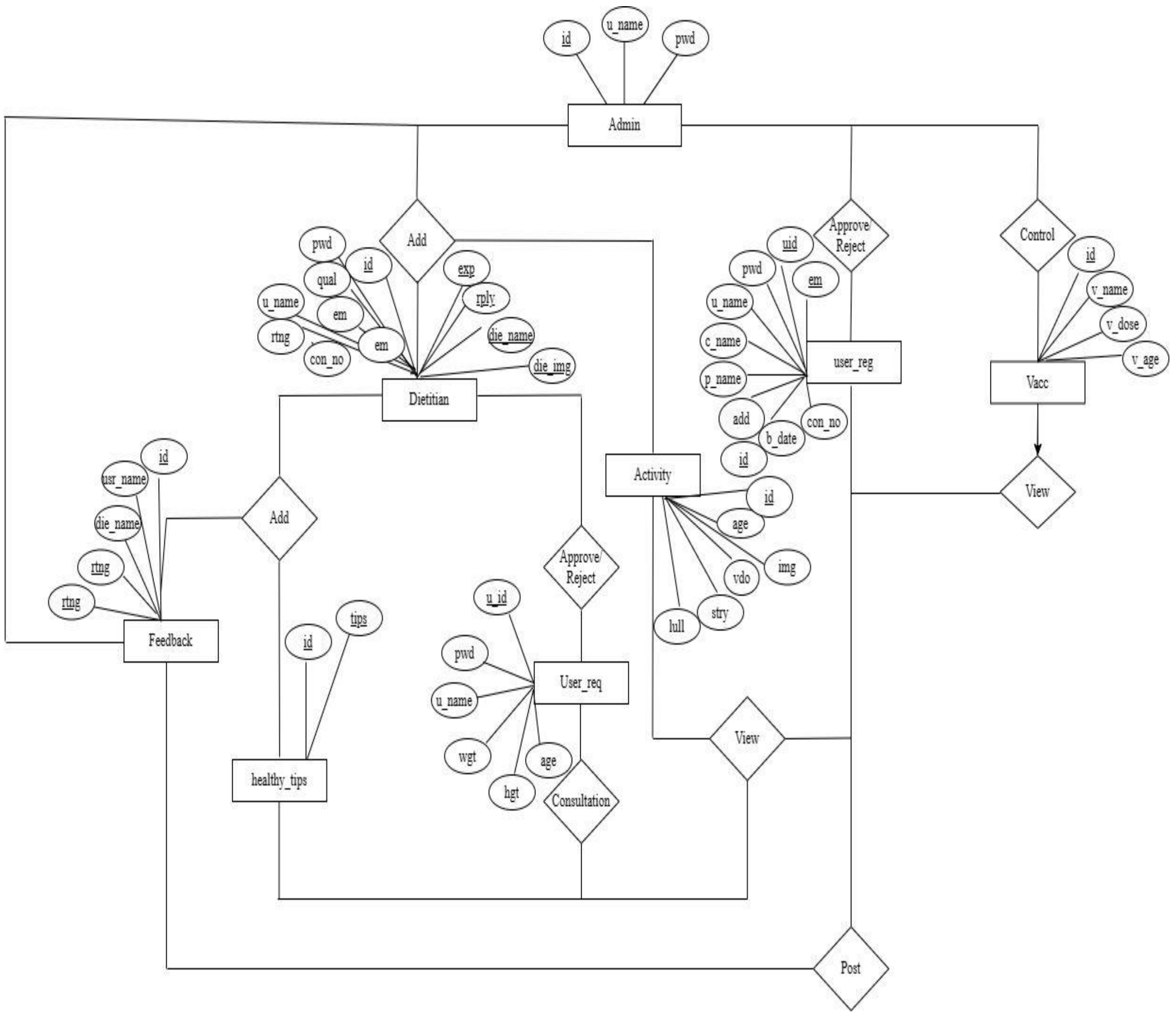
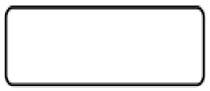
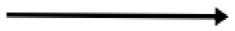
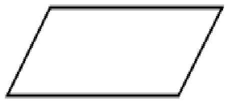
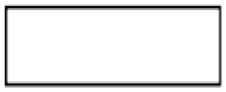
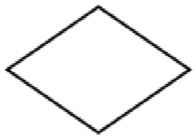
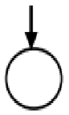


Fig 7: ER diagram

3.5 SYSTEM FLOWCHART

A System flowchart is a systematic graphical representation of an algorithm or a process. A flow is described as the graphical representation of the sequence of information. An information system flowchart shows how data flows from source document through computer to the final distribution of users. Program flow chart the sequence of instruction in a single program or subroutine. Different symbols are used to draw each flow chart. The flowchart shows the points or input and output or logic sequence of various processing steps in the system and the relationship of one element, the system to the other part of the system. Or to the other information of the system. A flowchart often symbolizes the most important steps of the process without detailing of the way the work is to be performed.

NAME	SYMBOL	USE IN FLOWCHART
Rounded Rectangle		Denotes the start and end of a program.
Flow Line		Denotes the direction of logic flow in the program.
Parallelogram		Denotes either an input operation or an output operation.
Rectangle		Denote a process to be carried out.
Diamond		Denotes a decision to be made.
Circle		Denotes a decision to be made.

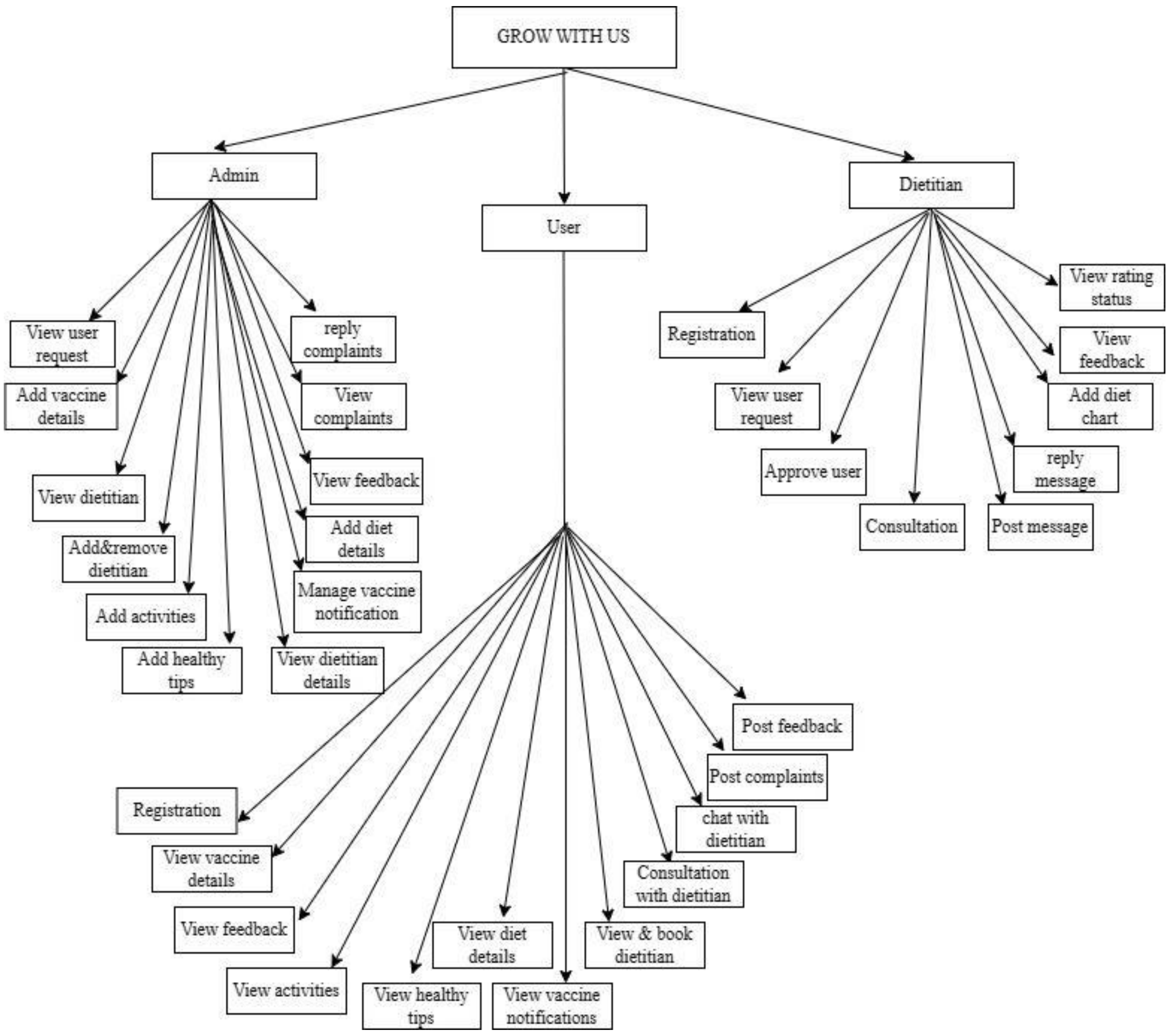


Fig 8: System Flow Chart

3.6 MENU TREE

Menu tree is a data flow methodology. The graphical representation of dataflow, communication and defining the modules and their relationship with each is known as menu chart. The method decomposes and modularizes the reasoning and promotes the maintainable provable systems.

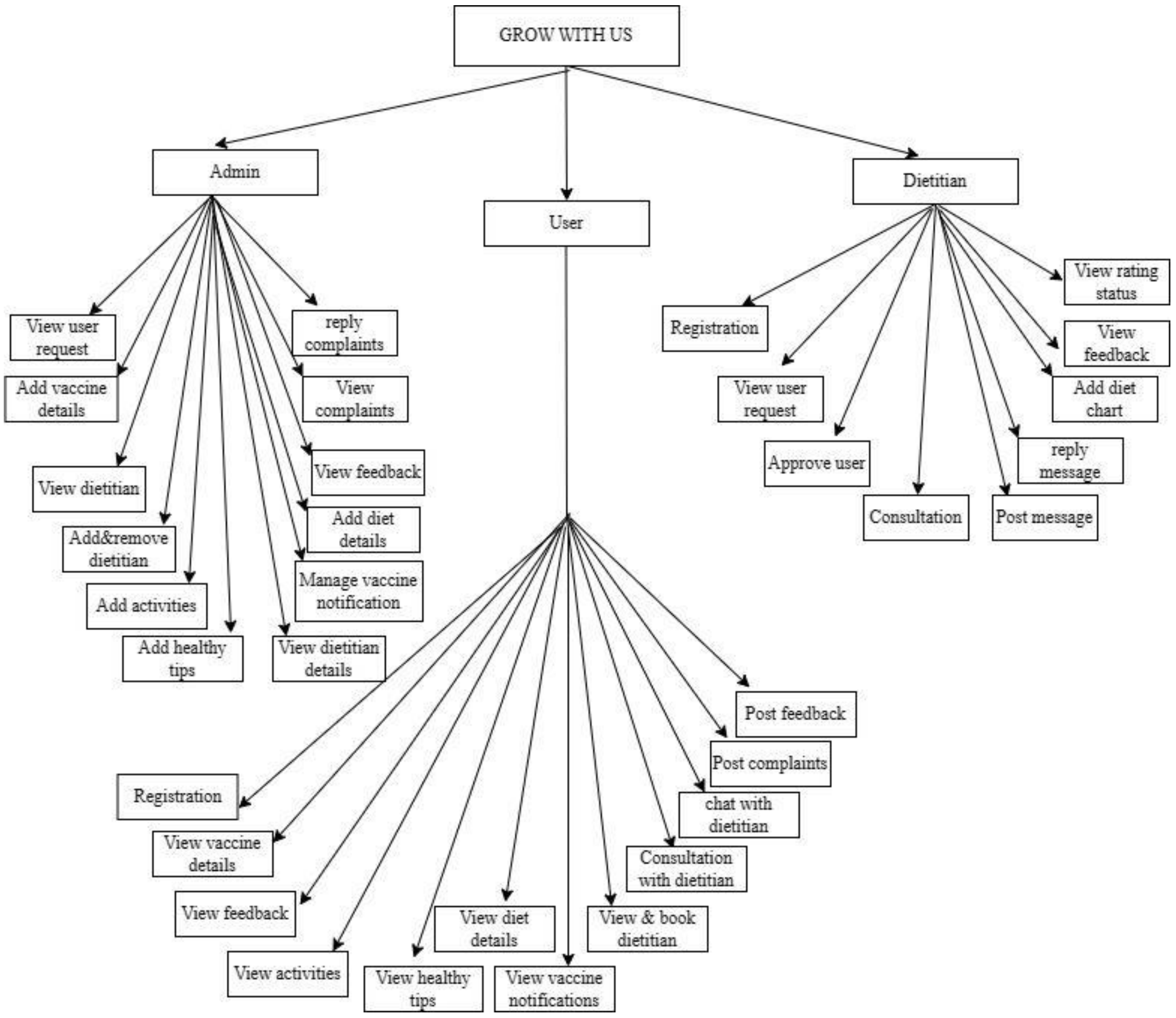


Fig 9: Menu tree

3.7 DATABASE DESIGN

A database design is a collection of interrelated data stored with the maximum redundancy to serve many users quickly and effectively. The general theme behind a database is to handle information as an integrated whole. The general objective is to make information access easy, quick, inexpensive and flexible for other user. The database serves as the repository of data, so a well-designed database can lead to better program structure and reduces procedural complexity. Database design is considered as a standard for management information system and is available virtually for every computer system.
Specific objectives in a database are:

- **Controlled Redundancy**: A unique aspect of database is storing data only once, which control redundancy and improve and system performance.
- **Ease of Learning and Use**: Database should be modifying without interfering with established ways of using data.
- **Data Independence**: It refers to the ability to add new data without rewriting an application program.
- **More Information Low Cost**: Using storing and modifying more information at low cost is important.
- **Accurate and Integrating**: The accuracy of database ensures that data quality and content remain constant.
- **Recovery from Failure**: Integrity with multi user access to a database, the system must recover after it is down with no loss transaction.
- Privacy and Security: Database should be prevented from unauthorized access. Users must be positively identified and their action monitored.
- **Performance**: This emphasizes on the response time to enquire suitable to the use of the data.
- Keys: A key is a column or columns used to identify rows. Various types of keys are:
- **Primary key**: A primary key is a column used to uniquely identify a particular row in a table. Every database table should have or more columns designed as the primary key. The value of this they holds should be unique for each record in the database.
- **Candidate key**: A candidate key is a combination of one or more columns, the values of which uniquely identifies each row of the table.
- **Foreign key**: A foreign key is one or more columns whose values are based on the primary or candidate key of another table. These keys are used to create relationship between tables. Natural relationship exists between tables in most database structures.
- **Super keys**: A super key is a combination of attributes that can be uniquely used to identify a database records. A table might have many super key. Candidate keys are special subset of super keys that do not have any extraneous information in them.

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
u_name	varchar(50)	Not Null	Name
pwd	varchar(50)	Not Null	Password

Table.1. admin

Fieldname	**Datatype**	**Constraints**	**Description**
u_id	Int(10)	primary	Id
u_name	varchar(30)	Not Null	User Name
c_name	varchar(30)	Not Null	Child name
p_name	varchar (30)	Not Null	Parent name
add	varchar (30)	Not Null	Address
b_date	date	Not Null	Birth date
age	integer	Not Null	Age
em	varchar (30)	Not Null	Email
con_no	int (10)	Not Null	Contact number
pswd	int (10)	Not Null	Password

Table.2. user_reg

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
uname	varchar(30)	Not Null	User name
die_name	varchar(30)	Not Null	Dietitian name
b_date	date	Not Null	Birth date
qual	varchar(30)	Not Null	Qualifications
exp	varchar(30)	Not Null	Experience
con_no	int (10)	Not Null	Contact number
pwd	int(10)	Not Null	Password
appr	varchar(30)	Not Null	Approvement
rply	varchar(30)	Not Null	reply
rtng	varchar(30)	Not Null	rating
die_img	varchar(30)	Not Null	Dietitian image

Table.3. dietitian

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
name	varchar(50)	Not Null	Name
am	varchar(50)	Not Null	Amount

Table.4. payments

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
st_t_m	int(10)	Not Null	Standard three months
st_s_m	int(10)	Not Null	Standard six months
st_o_y	int(10)	Not Null	Standard one year
ex_t_m	int(10)	Not Null	Exclusive three months
ex_s_m	int(10)	Not Null	Exclusive six months
ex_o_y	int(10)	Not Null	Exclusive one year

Table.5. plan

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
crd_name	varchar(30)	Not Null	Card name
crd_no	int(10)	Not Null	Card number
sec_code	int(10)	Not Null	Security code
pos_code	int(10)	Not Null	Postal code
exp_date	varchar(30)	Not Null	Expiry date

Table.7. paid_details

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
ch_hd	varchar(100)	Not Null	Chart heading
exp	datetime	Not Null	explanation

Table.8. diet_chart

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
rply	VarchR(30)	NULL	Reply

Table.10.reply

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
age	int(10)	Not Null	Age
img	varchar(500)	Not Null	Image
vd	varchar(500)	Not Null	Video
stry	varchar(500)	Not Null	Story
lull	varchar(500)	Not Null	Lullaby

Table.11. age_upto_five

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
age	int(10)	Not Null	Age
img	varchar(500)	Not Null	Image
vd	varchar(500)	Not Null	Video
stry	varchar(500)	Not Null	Story
lull	varchar(500)	Not Null	Lullaby

Table.12. age_upto_ten

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
age	int(10)	Not Null	Age
img	varchar(500)	Not Null	Image
vd	varchar(500)	Not Null	Video
stry	varchar(500)	Not Null	Story
lull	varchar(500)	Not Null	Lullaby

Table.13. age_upto_sixteen

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
tps	Varchar(50)	Not Null	Tips

Table.14. healthy_tips

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
age	Int(10)	Not Null	Age
img	varchar(500)	Not Null	Image
vdo	varchar(500)	Not Null	Video
stry	varchar(500)	Not Null	story
lull	varchar(500)	Not Null	lullaby

Table.15. Activity

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
usr_name	Int(10)	Not Null	User name
die_name	varchar(30)	Not Null	Dietitian name
fdbck	varchar(100)	Not Null	Feedback
rtng	varchar(30)	Not Null	Rating

Table.16. feedback

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
usr_name	Int(10)	Not Null	User name
die_name	varchar(30)	Not Null	Dietitian name
ad_rply	varchar(100)	Not Null	Admin reply

Table.17. complaints

Fieldname	**Datatype**	**Constraints**	**Description**
id	int(10)	Primary	Id
v_age	Int(10)	Not Null	Vaccine age
v_name	varchar(30)	Not Null	Vaccine name
v_dose	varchar(100)	Not Null	Vaccine dose

Table.18. vacc

3.8 NORMALIZATION

Normalization is a refinement process to resolve the issues like inconsistency, ambiguity and redundancy. It is also used to avoid insertion, deletion and updating anomalies. All the tables have been normalizing up to the third normal form. Designing a database is a complex task the normalization theory is useful aid in this designing process. A bad database may lead to certain undesirable situations such as:

- Repetition of information
- Inability to represent certain information
- Loss of information

This is important that a database using that we are using may free from data redundancy and inconsistency. For this need we maintain the table in a normalised manner. A normalized data can also encompass many related activities of an organisation thereby minimizing the need for rewriting the application programs. Thus normalization helps one to attain a database design and thereby ensures efficiency of database.

Purpose of normalization:

- Helps to simplify the structure of tables.
- To structure the data so that there is no repetition of data that helps in saving space.
- To permit simple retrieval of data in response to query and report requests.
- To simplify the maintenance of data through updates, insertion and deletion.

To minimize these anomalies, normalization may be used. Each step in the process of normalization is known as normal form. The four normal forms in the process of normalization are:

First Normal Form (1NF)

A relation is in First Normal Form (1NF), if and only if all attributes are based on a single domain. The objective of normalizing a table is to removes its repeating groups and to ensure that all entries of the resulting table have almost single value. The objective of 1NF is to divide the database into logical unit called tables. When each table has been designed, the primary key is assigned to most or all tables. Hence all tables in the system are in first normal form.

Second Normal Form (2NF)

A relation is said to be in 2NF, when it is 1 NF and every non key attribute is fully dependant on a key. The objective of 2NF is to take data that is partially dependant on the primary key, enter the data in another table. Now consider the database of this system. In this, there is a total of eight tables and all tables are in second normal form.

Third Normal Form (3NF)

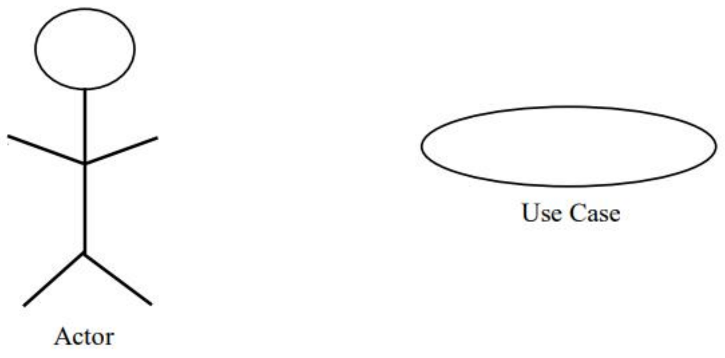
A relation is said to be in 3NF, when it is already in 2NF and every non key attribute is functionally dependant only on primary key. The objective of 3NF is to remove data in a table that is not dependant on primary key and hence all the tables of this system are in third normal form.

Benefits of Normalization

- To permit simple retrieval of data in response to query and report request.
- Help to simplify the structure of tables.
- Data modification anomalies are reduced.
- Data consistency within the database.

3.9 USE CASE DIAGRAM

A Use case is a set of scenarios that describing an interaction between user and system. A use case diagram displays the relationship among the actors and use cases. The two main components of a use case diagram are cases and actors. A use case defines the interaction between external actors and system under consideration to accomplish goals. Actors must be able to make decision, but need not to be human: An actor must be a person, a company or organisation.



A Use case represents a user or another system that will interact with the system you are modelling. A use case is an external view of the system that represents actions that user might perform in order to complete a task. Use cases are used in almost every project. They are helpful in exposing requirements and planning the project. During the initial stage of a project most use case should be defined, but as the project continuous might become visible.

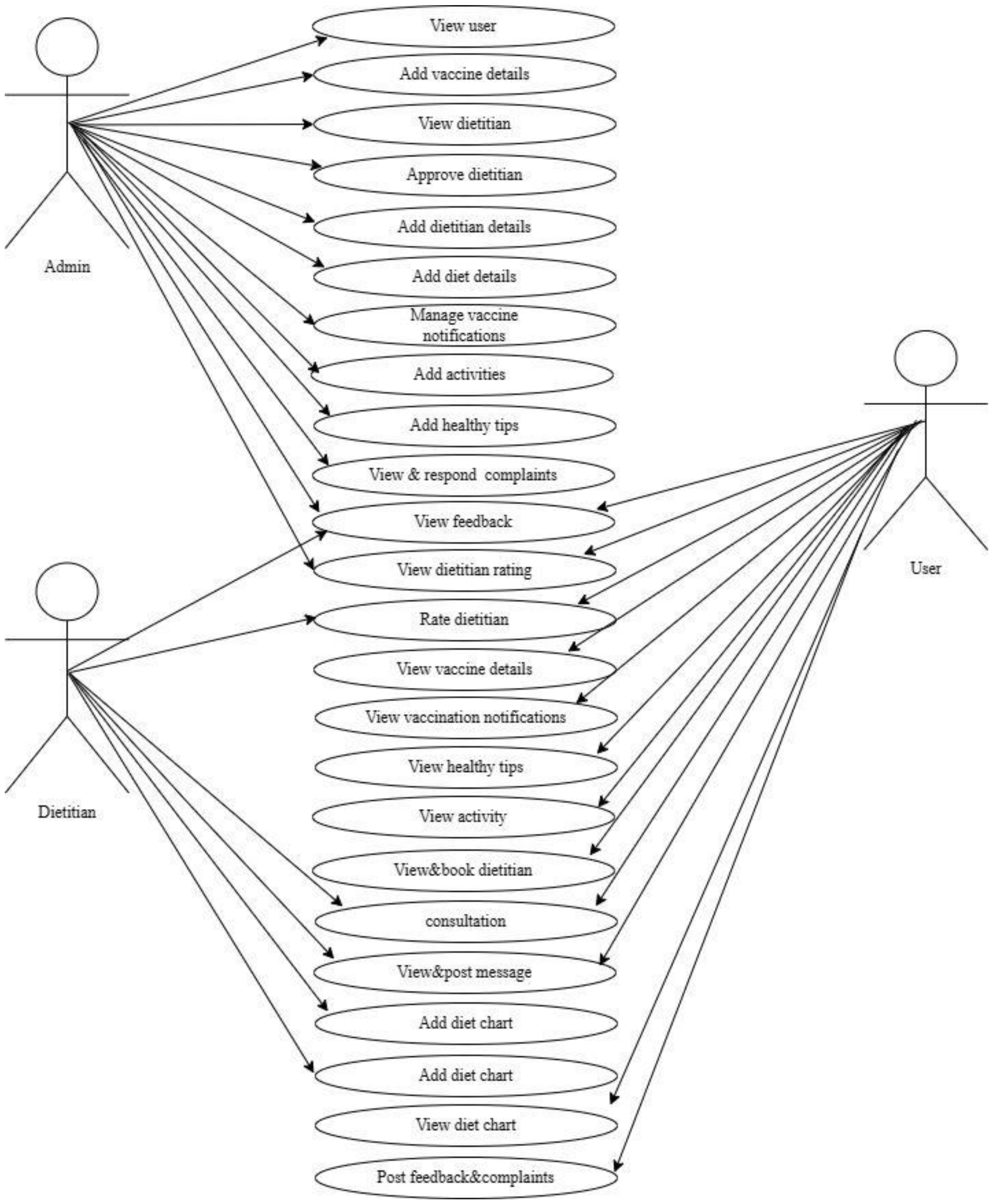


Fig 10: Use case diagram

CODING AND IMPLEMENTATION

4.1 INTRODUCTION

When considered as a step-in software engineering, coding is viewing as a natural consequence after design. However, programming language characteristics and code style can profoundly affect software quality and maintainability. The coding step translates a detailed design representation into a programming language realization. The translation process continues when a compiler accepts source code as output. The initial translation step in detail design to programming language is a primary concern in the software engineering context. Improper interpretation of a detailed design specification can lead erroneous source code. Style is an important attribute of source code and can determine the intelligibility of a program. The element of a style includes internal documentation, method for data declaration, procedures for statement construction, I/O coding, and declaration. In all cases simplicity and clarity are characteristics.

In this section a brief introduction of the technologies of the software used in the proposed system are discussed.

FEATURES OF LANGUAGE

- Backend and maincode designed using python.
- Frontend using django framework
- Database used is SQLite3

PROGRAMMING LANGUAGES USED

- **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.

Python works on different platforms. It can be treated in a procedural way, an object-oriented way or functional way. Python can be used on server to create web applications. It is also used for web development, software development, system scripting etc.

- **HTML**

HTML, or *HypertextMarkupLanguage*, is used by web programmers to describe the contents of a web page. It is not a programming language. You simply use HTML to indicate what a certain chunk of text is-such as a paragraph, a heading or specially formatted text. All HTML directives are specified using matched sets of angle brackets and are usually called *tags*. Every HTML tag has an individual meaning associated with it. The tag adds a meaning to the document content. The webpage formatting is possible using the various HTML tags. For example <B> means that the following text should be displayed in bold. To stop the bold text, use the </B> directive. Most HTML directives come in pairs and surround the affected text.

- **CSS**

CSS stands for Cascading Style Sheets. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

CSS is designed to make style sheets for the web. It is independent of HTML and can be used with any XML-based markup language. Now let’s try to break the acronym:

Cascading: Falling of Styles

Style: Adding designs/Styling our HTML tags
Sheets: Writing our style in different documents

- **Javascript**

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.). There are also more advanced server-side versions of JavaScript such as Node.js, which allow you to add more functionality to a website than downloading files (such as Realtimecollaboration between multiple computers). Inside a host environment (for example, a web browser),
JavaScript can be connected to the objects of its environment to provide programmatic control over them.JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997. ECMA-262 is the official name of the standard. ECMAScript is the official name of the language.

a. Frameworks used

- **Bootstrap**

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

- **Django**

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source. It was designed to help developers take applications from concept to completion as quickly as possible. Django takes security seriously and helps developers avoid many common security mistakes.

b. RDBMS

- **SQLite3**

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects. It is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file.
SQLite is a compact library. With all features enabled, the library size can be less than 750KiB, depending on the target platform and compiler optimization settings. (64-bit code is larger. And some compiler optimizations such as aggressive function in lining and loop unrolling can cause the object code to be much larger.) There is a trade-off between memory usage and speed. SQLite generally runs faster the more memory you give it. Nevertheless, performance is usually quite good even in low-memory environments. Depending on how it is used, SQLite can be faster than direct filesystem I/O.

4.2 INTRODUCTION TO IMPLEMENTATION

Implementation is the state in the project where the theoretical design is put into real test. All the theoretical and practical works are now implemented a working system. This is the most crucial stage in the lifecycle of a project. The system goes for implementation only after passing through some rigorous testing, especially when it comes to operation system and other system software, the testing and implementation phase assumed greater significance.

4.3 IMPLEMENTATION OF PROPOSED SYSTEM

The implementation is the final stage, and it is an important phase. It involves the individual programming, system testing and operational running of the developed system that constitute the application system. One major task of implementation phase is the education of users, which should really have been taken place much earlier during the investigation and design work. It has been observed that many of the users resist advent of new technology or systems. This is one of the important factors of the considered for the actual implementation of the project. During the implementation phase the system takes physical shape. For the system implementation, planning is necessary. The implementation phase of the software development is concerned while design specification into a source code. The user tests the developed system and changes are made according to suit his/her needs. Our system has been successfully implemented. Before implementation, several tests have been conducted to ensure that no errors are encountered during the operation, in case of errors they must be rectified effectively. Errors can be of various types mainly minor or major, requiring the corresponding effort. Even a dot or comma and sometimes causes major errors. The implementation phase ends with an evaluation of the system after placing into operation for a period.

4.4 INSTALLATION PROCEDURE

For implementing the project the different software must be installed for its fast and better execution. To install the system, the primary need is java without which the system will not have a proper utilization. Not much installation is required to run the system. The following components are needed for the installation procedure:

- HTML or any other web server which is compiled with python
- MySQL(for database)
- A preferable operating system like windows/Linux

Copy all the required files to the web server root folder and create the database. In the field of computer software, the term software build refers either to the process of converting source code file into standalone software artefacts that can be run on a computer, or the result of doing so. One of the most important steps of a software building is the compilation process where source code files are converted into executable code.

- Before installing the software make sure that Team Viewers is installed in your system.
- Also install MySQL in your system.
- You have to make sure that all applications have to be closed before the installation.

- Install the software in all the client system in the network.

The project web content management system is installed by the following steps:

- First host this application into the web.
- After that URL will obtain.
- Log on to the site using the given or specified URL.
- Go to this site directly and view various options.
- Select various options that must be necessary for a user.
- Inset the secondary devices that contain the project.

For implementing the project the different software must be installed for its fast and better execution. To install the system, the primary need is web based environments without which the system will not have a proper utilization. To install the system, it is a must to setup a centralized server which can hold social networking web system including the user's information database. The database is accessed through web pages using browser at the client end.

- Python
- SQLite3
- A preferable Operating System like Windows 7/8 /10.

On the client side part we have to install HTML and java. Different sort of browsers can be used at the client end.

TESTING

5.1 INTRODUCTION

Software testing is critical element of software quality assurance the ultimate review of specification, design and coding. Testing represents an interesting anomaly for the software. During the earlier definition and development phase it was attempted to build software from abstract concepts to a tangible implementation. The testing phase involves the testing of the developed system using various test data. After preparation of the test data the system under study is tested using those test data. While testing the system by using the test data, errors were found and corrected. Thus a series of test layer performed for the proposed system before the system was already for implementation.

5.2 TESTING METHODS

Software system testing methods are traditionally divided into Black box testing and White box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

- Black box testing (is also called functional testing) is testing that ignores the internal mechanism of the system or component and focuses solely on the output generated in responses to selected input and execution conditions.
- White box testing (is also called structural testing or glass box testing) is testing that takes into account the internal mechanism of the system or component.

Black box testing

This testing methodology looks at what are available for an application and what expected outputs are should be get from each inputs. It is not concerned with the inner working of the applications under takes to achieve particular outputs or any other internal aspects of the application that may be involved in the transformation of an input into an output. Most black box testing tools employee either coordinate based on interaction with the application Graphical User Interface (GUI) or images recognition an example of a black box would be search engine.

White box testing

This testing methodology looks under the covers and into the subsystem of application whereas black box concerns it exclusively with inputs and outputs of an application. White box testing enables you to see what is happening inside the application. White box testing provides a degree of sophistication that is not available with black box testing as the tester is able to refer to and interact with the objects that comprise and application rather than only having access to user interface.

5.3 LEVELS OF TESTING

5.3.1Unit testing

A unit testing focuses verification effort on the smallest unit of software design using theunit test plan prepared in the design phase of the system, improvement control paths are test to uncover the error within the module. This testing was carried out during the coding itself. Each module is going to be working satisfactory as the expected output the module.

5.3.2Integration testing

Integration testing is the systematic technique for constructing the program structure while at the same time, conducting it's the programming structure while at the same time design conducting test to uncover errors associated with the interface. The objectives are to take unit tested modules and build the program structure that has been directed by design. All modules are combined in this testing step. The entire program is tested as a whole. If a set of errors is encountered correction is difficult because the isolation of causes is completed by vastness of the entire program. Using integrated test plans prepared in the design phase of the system developed as a guided, the integration was carried out. All the errors found in the system were corrected for the next testing steps.

5.3.3 Validation testing

At the end of the integration testing software is completely assembled as a package interfacing errors have been uncovered and corrected and final series of software validation testing begins. Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software fractions in a manner that can be reasonably accepted by the user customer. Software validation is achieved thought a series of black box test that demonstrates conformity with requirements. After the validation test has been completed one of following two possible conditions exists.

- The functions or performance characteristics confirm to specification and are accepted. A deviation from specification is uncovered and a deficiency list is created.
- Deviation or errors discovered at this project is corrected prior to the completion of the help users by negotiating to establish a method resolving deficiencies.

5.3.4 Output testing

No system could be useful if it does not produce the required output in the specific format. The output user interface was tested to ensure if the system provide correct, accurate output in the specified format. The output generated or displayed by the system is tested asking the users about the format required by them.

- Conditional testing

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

- Hybrid testing

Hybrid testing examines class operation at an algorithmic granularity but only examines public methods and variables. From the standard point of the application, hybrid testing qualifies as white box testing since the classes being tested may not be exposed to the application. From the class level, hybrid testing qualifies as black box testing since private methods and variables are not exposed and how the results are produced is never called into question.

SECURITY, BACKUP AND RECOVERY MECHANISMS

6.1 SECURITY

6.1 SECURITY

Security is an important consideration in an application. There are many possibility threats to the security and integrity of any system where more than one user is associated with the system. Software integrity of any system has more than one user is associated with the system. Software integrity has become increasingly important. The attributes measures a system system's ability to withstand on security. Attacks can be done on all three components of software: programs, data and documents. Security test attempts to verify the protection mechanism built into the system, which will protect it from improper penetration. The system is designed in such a way that only authorised user can access it. Security concepts:

- Authentication
- Authorization

Authentication

This is the process of determining a user's identity and forcing users to prove they are who they claim to be; usually this involves entering credential in some sort of login page or windows.

Authorization

Once the user is authenticated, authorization is the process of determining where that user has sufficient permission to perform a given action, such as viewing a page or retrieving information from the database.

- Security measures are taken by reporting the contents of encrypted files to an external media
- Each time, system maintenance is done.
- Error reporting are collected and updated regularly.

6.2 BACKUP AND RECOVERY MECHANISM

Backup facility is used in the software for backing of data. If any error occurs in the database due to any database error or software error or the database is deleted in any fault operation, you can copy the backup file to solve this problem.
For protecting the system from any kind of loss or damage, backup facility is offered. The entire program and associated database can be saved into a floppy disk or CD for the purpose of the failure use. If the program encounters as unexpected problem due to a corrupt database, the backup in the floppy disk or CD can be used. The data do not lost from that device due to a usual failure. The entire database is recommended to be back daily. So there is no problem for the recovery of data.

ONLINE HELP AND USER MANUALS

7.1 USER MANUALS

The User Manuals provides the detailed description regarding the usage of the software. The main user tips are:

- All the required operations are specified in various links.
- Never share your user id and password.
- Do not send extremely confidentially information as autographs to any user.
- Change your password periodically

CONCLUSION

8.1 CONCLUSION

By reviewing the project, we presented an approach that enhances a virtual agent by the ability to interpret and respond to social cues of users participating in a simulated job interview. In order to achieve seamless credible interaction, our system automatically recognizes the user’s social cues in real time. Based on these, the virtual recruiter reacts and adapts to the user’s behaviour. Furthermore, the interaction with the virtual agent can be recorded and presented to the user to enhance the learning effect, for example, by identifying critical incidents during the simulated interview. The scenario manager was used to model the virtual recruiter’s interactive behaviour allowing the character to react to various social users recognized by the social cue recognition module. More precisely, we modelled mirroring and turn taking behaviour. Despite several reported problems, such as the realism of the character’s appearance, all participants’ reactions were mainly positive saying they would use such a system to train for real job interviews.

UPGRADABILITY POSSIBILITIES

9.1 UPGRADABILITY POSSIBILITIES

The development system is capable of efficiency performing routine activities. Although the system developed most of the functionalities the system can be enhanced by making little changes. Proper documentation of the code helps in easy addiction or modification of the code. If the code needs modification at a later time, it can be done the help of its creator. Enhancement can be made with simplicity and without the complexity.

The enhancement can be done for adding more functionality to the application, adding more special options for users. Here we can add option to automatically calculate the amount for booking, add on additional services etc. we can add event reminder to the portal.

APPENDIX

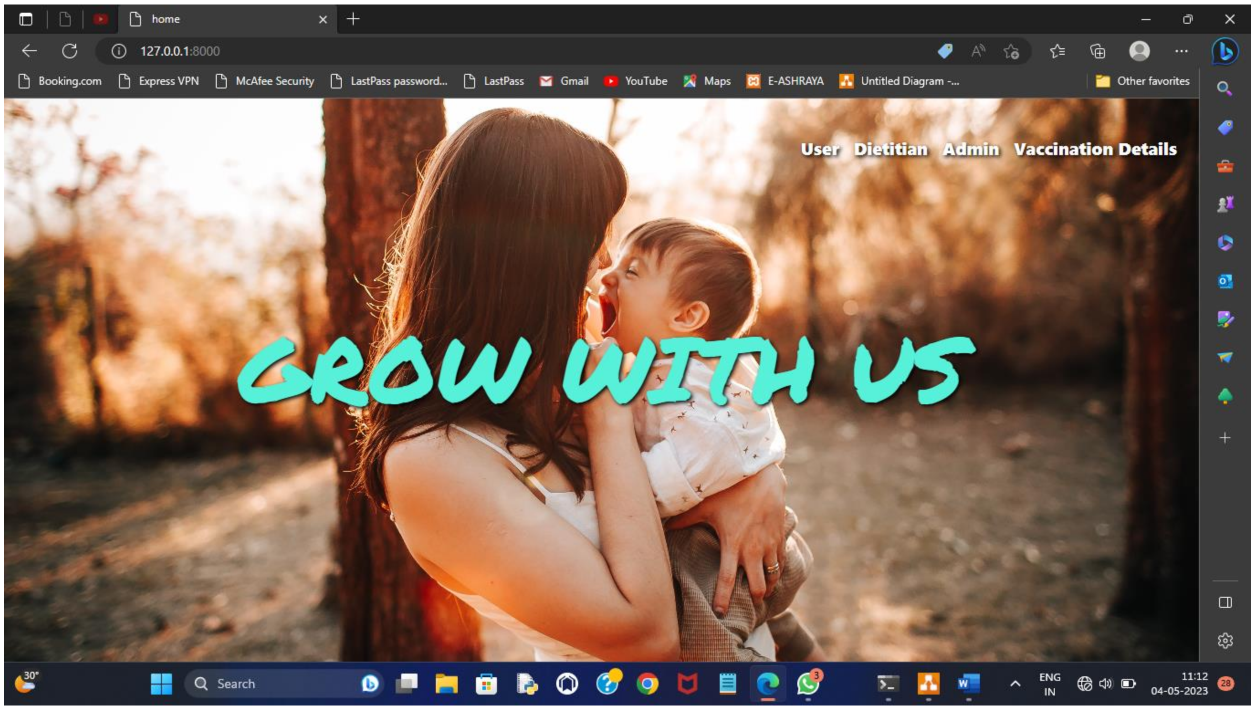
10.1 SOURCE CODE:

```
app.views
from django.shortcuts import render,redirect
from .models import*
from django.contrib.auth.models import auth,User
def home(request):
    return render(request,'home.html')
def user_reg(request):
    if request.method=="POST":
        user_names=request.POST['user_name']
        child_names=request.POST['child_name']
        parent=request.POST['parent_name']
        addresses=request.POST['user_address']
        date_of_birth=request.POST['dob']
        agee=request.POST['child_age']
        emaill=request.POST['mail']
        ph_number=request.POST['contact_number']
        password1=request.POST['u_password']
        password2=request.POST['uu_password']
 if password1==password2:
            if User.objects.filter(username=user_names).exists():
                print("already exists")
                return render(request,'user_registration.html')
            elif User.objects.filter(email=emaill).exists():
                print("invalid")
                return render(request,'user_registration.html')

      else:

usr=User.objects.create_user(username=user_names,password=password1,email=emaill)
                usr.save()



usave=users_reg(userr=usr,user_name=user_names,child_name=child_names,parent_name=pare
nt,address=addresses,birth_date=date_of_birth,age=agee,email=emaill,contact_number=ph_num
ber,password=password1)
                usave.save()
                print("saved")
                return redirect(user_login)
        else:
            return render(request,'user_registration.html')
    return render(request,'user_registration.html')

def user_login(request):
    if request.method=='POST':
        username=request.POST['uname']
        passwordd=request.POST['passwordd']
        if users_reg.objects.filter(user_name=username,password=passwordd).exists():
            user=auth.authenticate(username=username,password=passwordd)
            if user is not None:
                auth.login(request,user)
                print("login successfully")
                return redirect(user_apply)
            else:
                print("invalid login")
                return render(request,'user_login.html')
        else:
            print("username not found")

            return render(request,'user_login.html')
    return render(request,'user_login.html')


def user_apply(request):
    return render(request,'users_apply.html')

def user_profile_views(request):
    username=request.user
    n=users_reg.objects.filter(userr=username).all()
    di={'view':n}
    return render(request,'user_profile_view.html',di)


def update_user_profile(request,up):
    updates=users_reg.objects.get(id=up)
    if request.method=="POST":
        updates.user_name=request.POST['up_username']
        updates.child_name=request.POST['up_child']
        updates.parent_name=request.POST['up_parent']
        updates.address=request.POST['up_address']
        updates.birth_date=request.POST['up_dob']
        updates.age=request.POST['up_age']
        updates.email=request.POST['up_mail']
        updates.contact_number=request.POST['up_phn']
        updates.password=request.POST['up_psw']

        updates.save()
        return redirect(user_profile_views)
    return render(request,'user_profile_update.html',{'updates':updates})

def dietition(request):
    if request.method=="POST":
        dietition_username=request.POST['d_username']
        dietition_name=request.POST['d_name']
        dietition_dob=request.POST['d_dob']
        dietition_qulification=request.POST['d_qul']
        dietition_experience=request.POST['d_exp']
        dietition_no=request.POST['d_phn']
        password1=request.POST['password']
        password2=request.POST['c_password']
        dietition_img=request.FILES['d_img']
        dietition_approvement=request.POST['dietition_approvement']
        if password1==password2:
            user=None
            if User.objects.filter(username=dietition_username).exists():
                user=User.objects.get(username=dietition_username)
                if Dietition.objects.filter(user=user).exists():
                    print("diet already exits")
                    return render(request,'dietition_register.html')
            else:
                User.objects.create_user(username=dietition_username,password=password1).save()
                user=User.objects.get(username=dietition_username)
            if user:
                if Dietition.objects.filter(diet_Username=dietition_username).exists():
                    print("diet already exits")
                    return render(request,'dietition_register.html')
                else:

Dietition(user=user,diet_Username=dietition_username,diet_Name=dietition_name,diet_Date_of
_Birth=dietition_dob,diet_Qualifications=dietition_qulification,diet_Experience=dietition_exper
ience,diet_Mobile_no=dietition_no,diet_password=password1,diet_Image=dietition_img,diet_ap
provement=dietition_approvement).save()
                    return render(request,'dietition_register.html')
    else:
        return render(request,'dietition_register.html')
    return render(request,'dietition_register.html')

def dietitionlog(request):
    if request.method=="POST":
        dietition_username=request.POST["username"]
        password1=request.POST["password"]
        print(dietition_username)
        if
Dietition.objects.filter(diet_Username=dietition_username,diet_password=password1).exists():
            d=Dietition.objects.get(diet_Username=dietition_username)
            if d.diet_approvement=="yes":
                u=auth.authenticate(username=dietition_username,password=password1)
                if u is not None:
                    auth.login(request,u)
                    return redirect(dietition_apply)
                else:
                    return render(request,'dietition_login.html')
            else:
                print("waiting for admin approval")
        else:
            print('invalid')
            return render(request,'dietition_login.html')
    return render(request,'dietition_login.html')

def d_profile_view(request):

    username=request.user
    u=Dietition.objects.filter(user=username).all()
    di={'view':u}
    return render(request,'dietition_profile.html',di)


def dietition_apply(request):
    return render(request,'dietition_apply.html')

def update_dietition_profile(request,upd):
    updates=Dietition.objects.get(id=upd)
    if request.method=="POST":
        updates.diet_Username=request.POST['up_username']
        updates.diet_Name=request.POST['up_name']
        updates.diet_Date_of_Birth=request.POST['up_dob']
        updates.diet_Qualifications=request.POST['up_qualification']
        updates.diet_Experience=request.POST['up_experience']
        updates.diet_Mobile_no=request.POST['up_phn']
        updates.diet_password=request.POST['up_password']
        updates.diet_Image=request.POST['up_image']

        updates.save()
        return redirect(d_profile_view)
    return render(request,'dietition_profile_update.html',{'updates':updates})

def dietition_list_views(request):
    s=Dietition.objects.all()
    d={'view':s}
    return render(request,'dietition_list_views.html',d)

def add_plan(request):
    print(request.user)
```

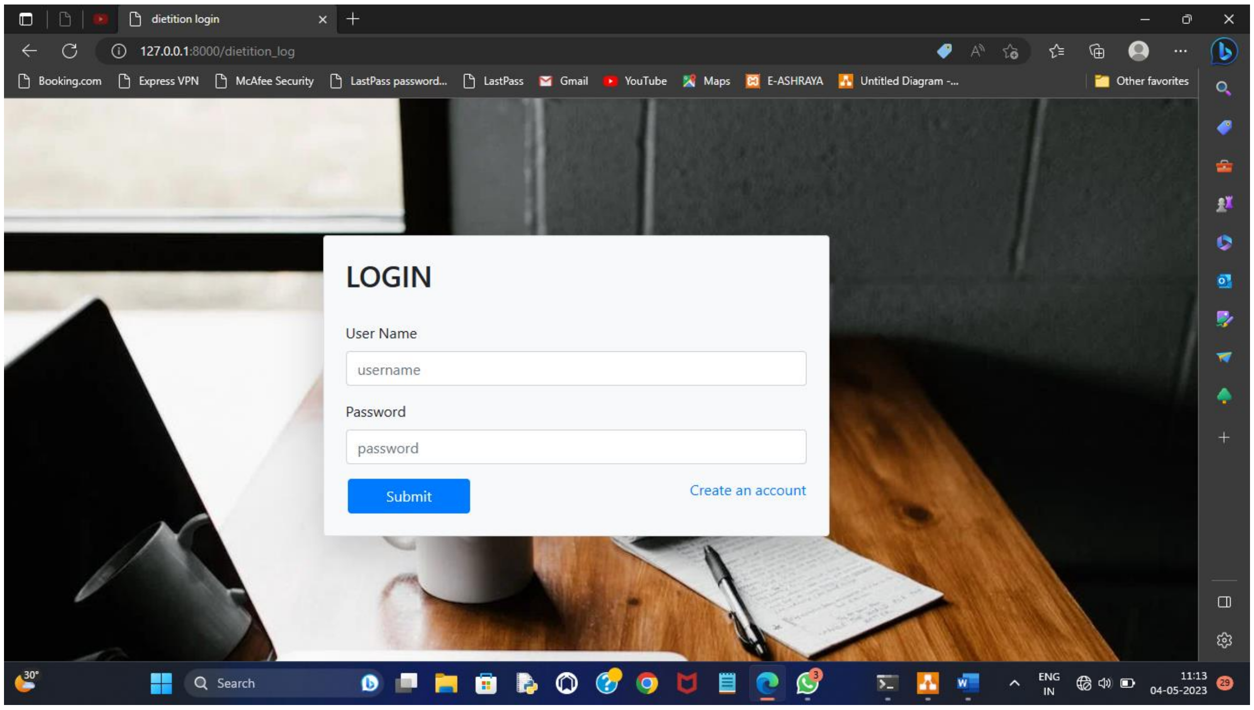
```
users=User.objects.get(username=request.user)
print(users)
if request.method=="POST":
   std_trials=request.POST['trial']
   std_three_months=request.POST['three_month']
   std_six_months=request.POST['six_month']
   std_one_yrs=request.POST['one_year']
   ex_three_months=request.POST['three_months']
   ex_six_months=request.POST['six_months']
   ex_one_yrs=request.POST['one_years']
```

INPUT AND OUTPUT FORMS

11.1 SCREENSHOTS



Home page



LOGIN

User Name

username

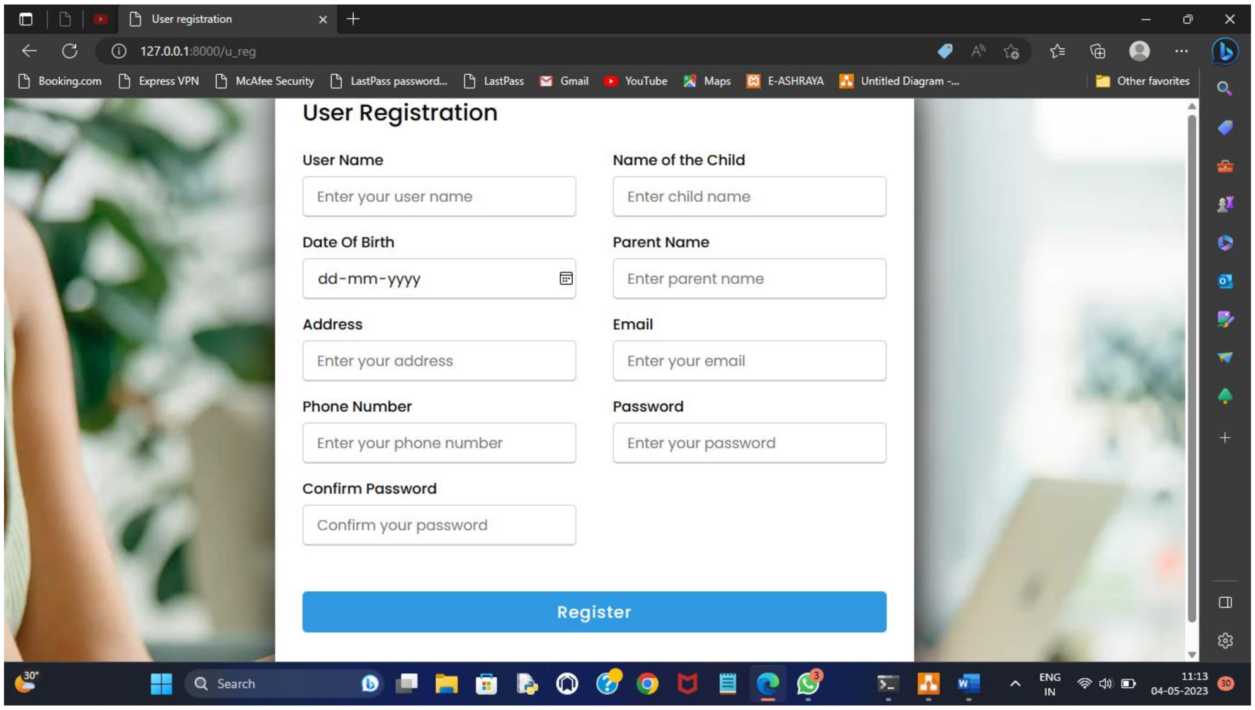
Password

password

Submit

Create an account

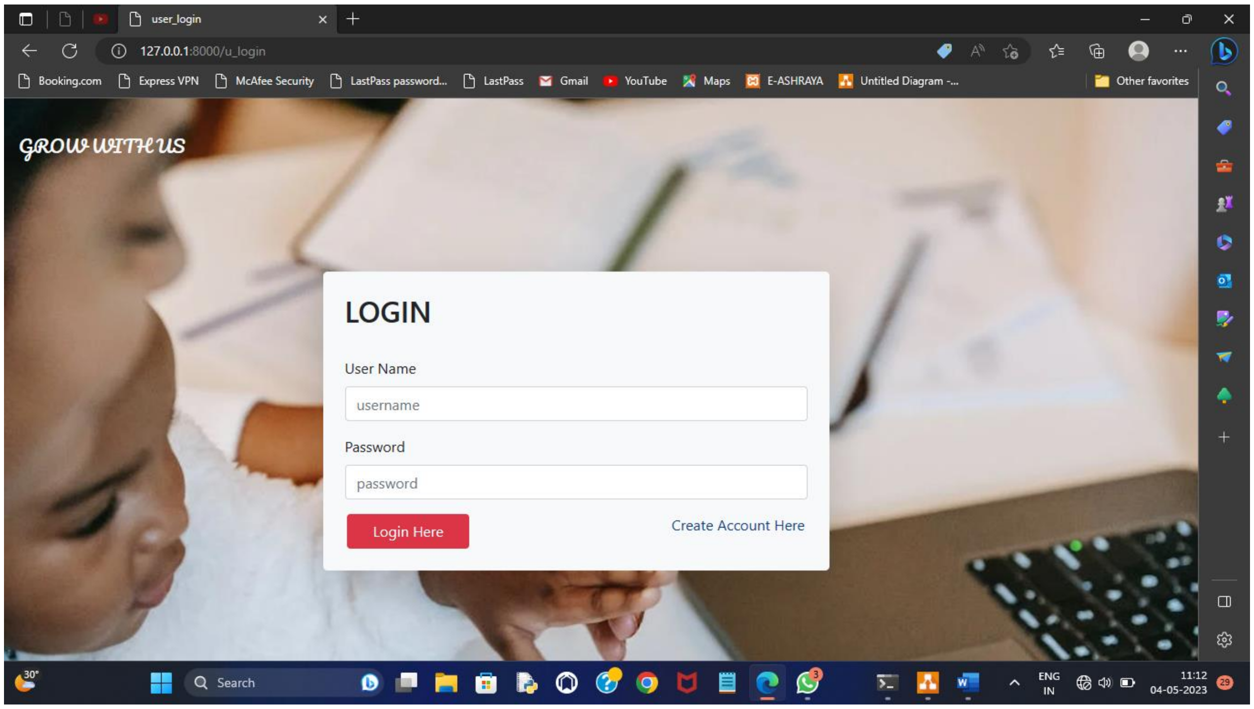
Dietitian login



User Registration

User Name	Name of the Child
Enter your user name	Enter child name
Date Of Birth	Parent Name
dd-mm-yyyy	Enter parent name
Address	Email
Enter your address	Enter your email
Phone Number	Password
Enter your phone number	Enter your password
Confirm Password	
Confirm your password	

User registration



LOGIN

User Name: username

Password: password

User login

GANTT CHART

12.1 GANTT CHART

Gantt chart shows time relationship between events of the production program has regarded as revolutionary in management. Gantt chart recognize the total program goals and it should be regarded as a series of inter-related supporting plan (or events), that people can comprehend and follow. The following figure is the gantt chart of "**GROW WITH US**". The plan explains the task versus the time it will take to complete.

	Task Name	**Start date**	**Due date**	**Duration**	**January**				**February**				**March**			
					W 1	**W 2**	**W 3**	**W 4**	**W 1**	**W 2**	**W 3**	**W 4**	**W 1**	**W 2**	**W 3**	**W 4**
1	System Analysis	15/01/23	30/01/23	16d		🟩	🟩	🟩								
2	System Design	31/01/23	11/02/23	12d				🟩	🟩	🟩						
3	Code Design	12/02/23	04/03/23	21d						🟩	🟩	🟩				
4	System Testing	05/03/23	16/03/23	12d									🟩	🟩		
5	Documentation	17/03/23	29/03/23	13d											🟩	🟩

TEAM MEETING MINUTES

13.1 TEAM WEEKLY MINUTES 1

Date: 13-01-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
Discussion and Selection of the Topic
Scheduled next meeting
The next meeting will be on 20-01-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 2

Date: 20-01-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 28-01-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 3

Date: 28-01-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 01-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 4

Date: 01-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
Abstract submission of the project
Scheduled next meeting
The next meeting will be on 03-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 5

Date: 03-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 07-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 6

Date: 07-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Thasleema N R
Individual progress report
Zeroth review of the project
Scheduled next meeting
The next meeting will be on 10-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 7

Date: 10-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 11-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 8

Date: 11-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 17-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 9

Date: 17-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 18-02-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 10

Date: 18-02-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Achyuthan Asok, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 03-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 11

Date: 03-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 11-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 12

Date: 11-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Achyuthan Asok, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 16-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 13

Date: 03-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
First review of the project
Scheduled next meeting
The next meeting will be on 17-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 14

Date: 17-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 18-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 15

Date: 18-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 23-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 16

Date: 23-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
Second review of the project
Scheduled next meeting
The next meeting will be on 25-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 17

Date: 25-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
To study the language used in the system.
Scheduled next meeting
The next meeting will be on 30-03-2023 at Nyeste Venture Technologies, Kowdiar

TEAM WEEKLY MINUTES 11

Date: 30-03-2023
Time: 10am to 1pm
Location: Nyeste Venture Technologies, Kowdiar
Present: Abhinav S Nair, Achyuthan Asok, Akhilesh B Kurup, Thasleema N R
Individual progress report
Third and Final review of the project

BIBLIOGRAPHY

15.1 BIBLIOGRAPHY

Textual References:

- Python Pocket Reference 5 edition by Mark Lutz
- Web Design: The complete Reference-Powell Thomas
- An Integrated Approach to Software Engineering - Pankaj Jalote
- Software Engineering-Pankaj Jalote
- System Analysis and Design Methods - Whitten, Bentley and Dittman
- Software Engineering Roger S Pressman System Analysis and Design- James A Senn

Websites:

- http://en.wikipedia.org/wiki/Internet_Information_Services
- www.freecstemplate.org
- https://www.github.com
- http://en.wikipedia.org/wiki/Database
- https://www.w3schools.com
- https://www.tutorialspoint.com

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx