5. DESIGN

5.1 System Design

System design is a reduction of an entire system by studying the various operations performed and their relationships within the system and the requirements of its success. One aspect of design is defining the boundaries of the system and determining whether or not the candidate system should consider other related system.

System can be defined, as an orderly grouping of interdependent components can be simple or complex. The most creative and challenging phase of the system life cycle is system design. The term design describes a final system and the process by which it is developed .It refers to the technical specifications that will be applied in implementing the candidate system .It also includes the construction of programs and program testing.

The first step in the system design is to determine how the output is to be produced and in what format. Samples of the output and the inputs are also presented .In the second step, input data and master files are to be designed to meet requirement of the proposed output .The processing phase's system's objectives and complete documentation.

System design has two phases:

- Logical
- Physical

The logical design reviews the present physical system, prepares the input and output and also prepares a logical design walk- through .We have to deal with how to take entries required and whether and how to process the user data. Also we have to deal with how to present the data

in an informative and appealing format .This design also involves the methodology to store, modify and retrieve data from the data base as per the requirement.

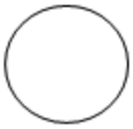
Physical design maps out the details of the physical system, plans the system implementation, devices a test and implementation plan and new hardware and software. We have to decide how and where to store the input data and how to process it so as to present it to the user in an easy, informative and attractive manner.

5.1.1 Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

Data Flow Diagrams Notations

Process: 

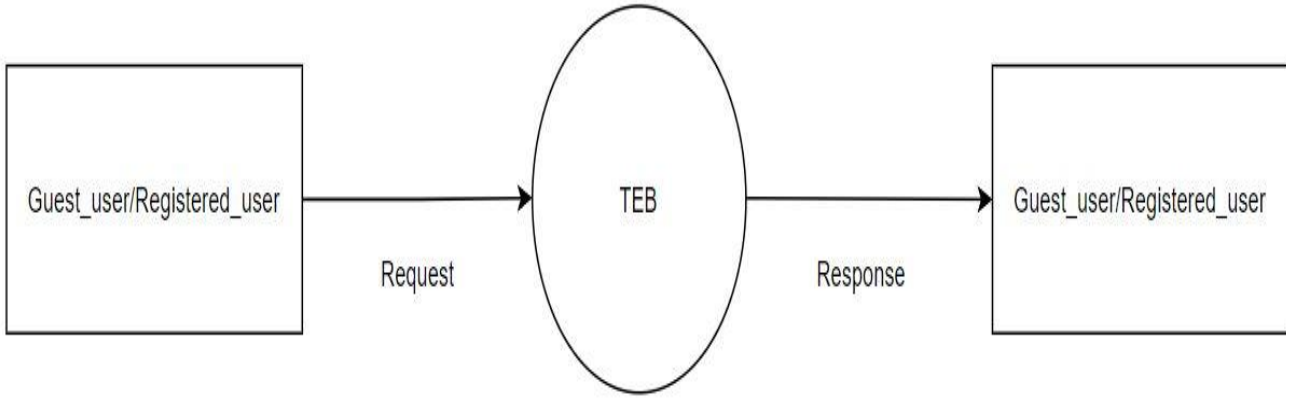
Input/output: 

Flow of direction: 

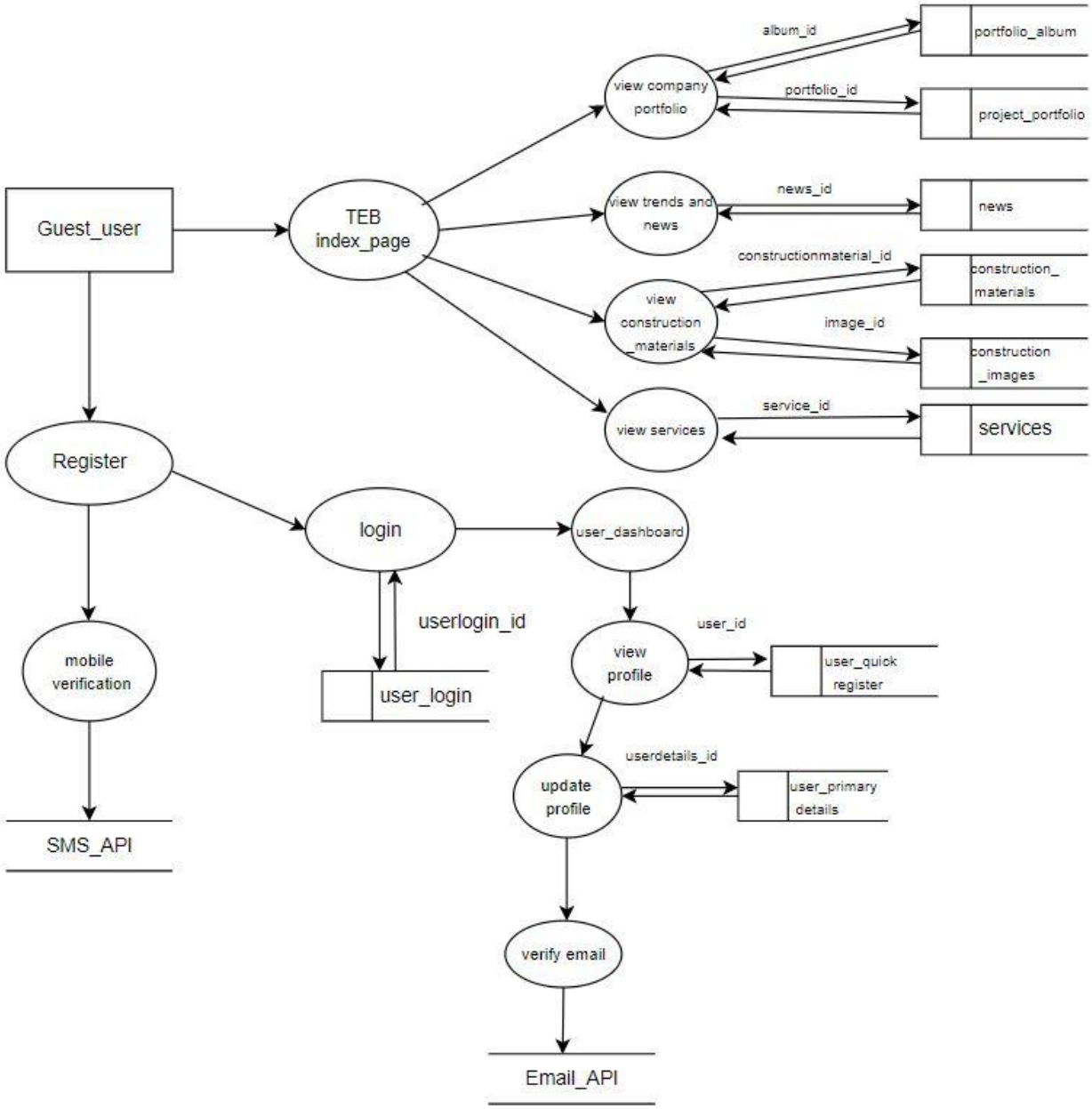
Database/File: 

5.2.2 Project DFD

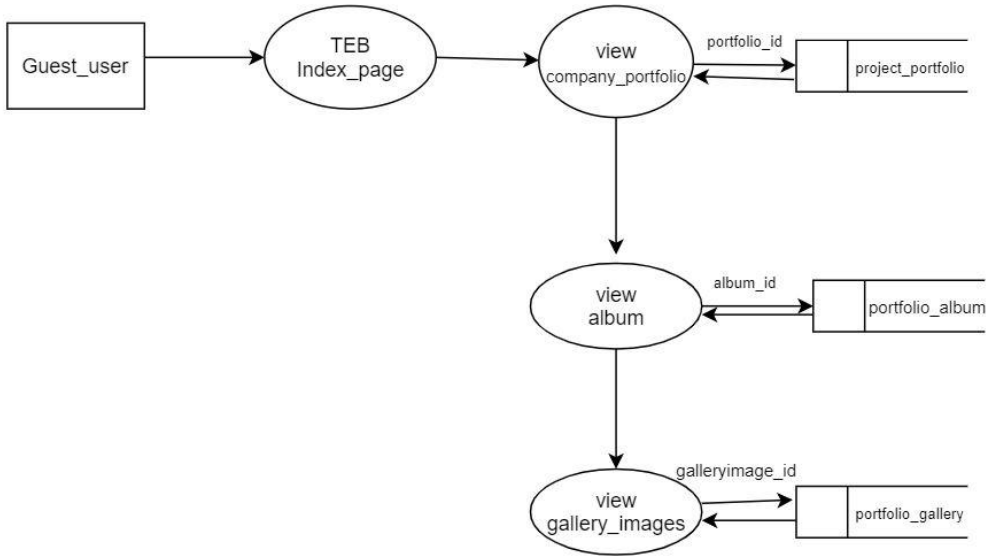
CONTEXT LEVEL / LEVEL 0



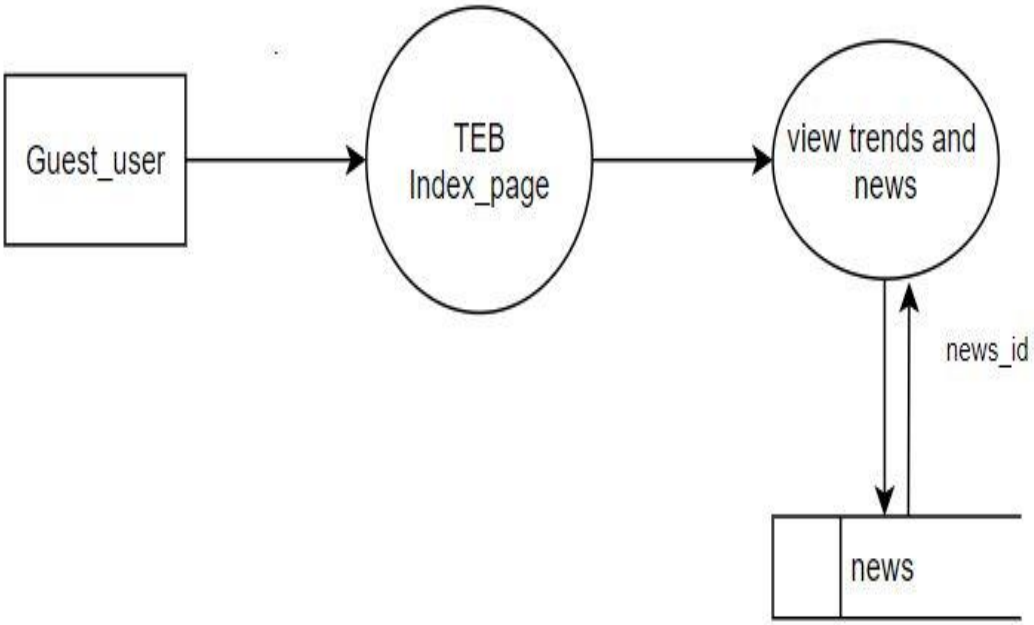
LEVEL 1 of Guest_user


Guest_user
TEB
index_page
view company
portfolio
album_id
portfolio_album
portfolio_id
project_portfolio
view trends and
news
news_id
news
view
construction
_materials
constructionmaterial_id
construction_
materials
image_id
construction
_images
view services
service_id
services
Register
login
user_dashboard
mobile
verification
userlogin_id
user_login
view
profile
user_id
user_quick
register
update
profile
userdetails_id
user_primary
details
SMS_API
verify email
Email_API

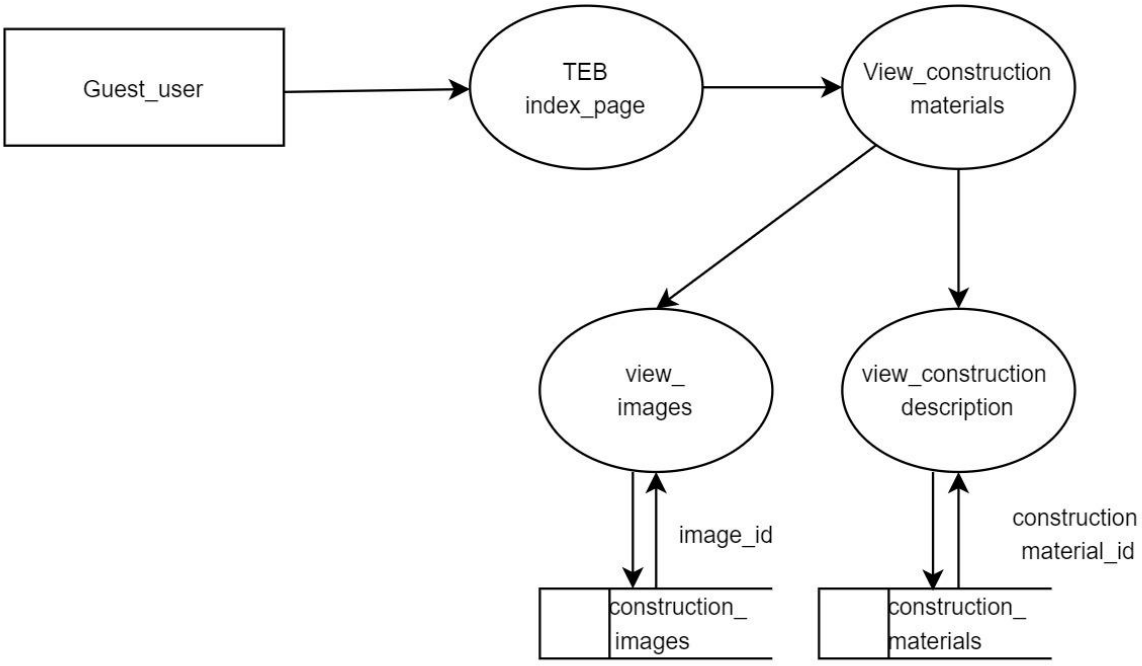
1.1


Guest_user
TEB
Index_page
view
company_portfolio
portfolio_id
project_portfolio
view
album
album_id
portfolio_album
view
gallery_images
galleryimage_id
portfolio_gallery

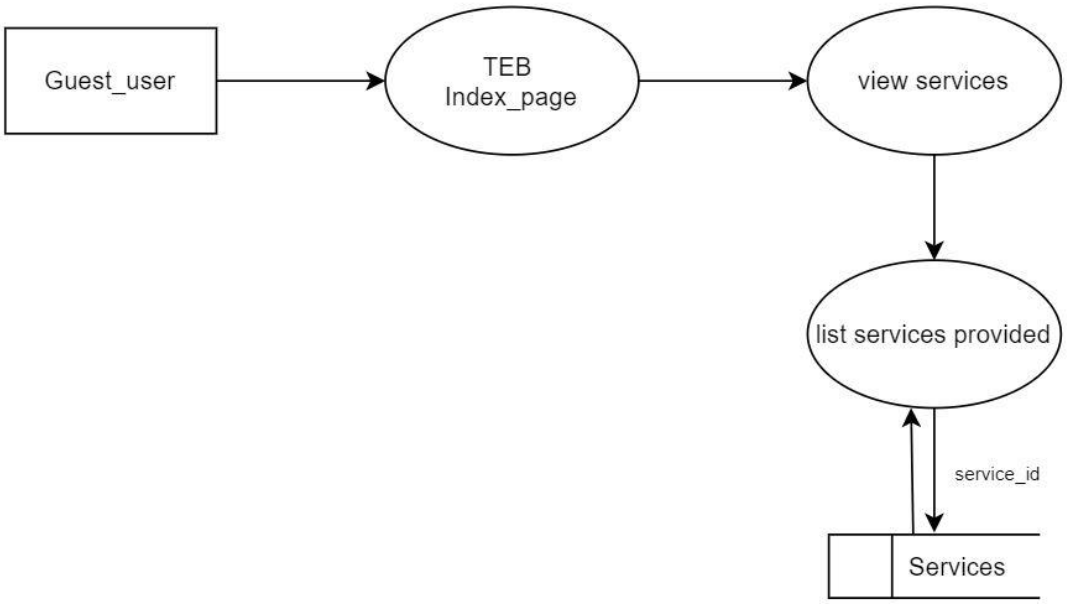
1.2



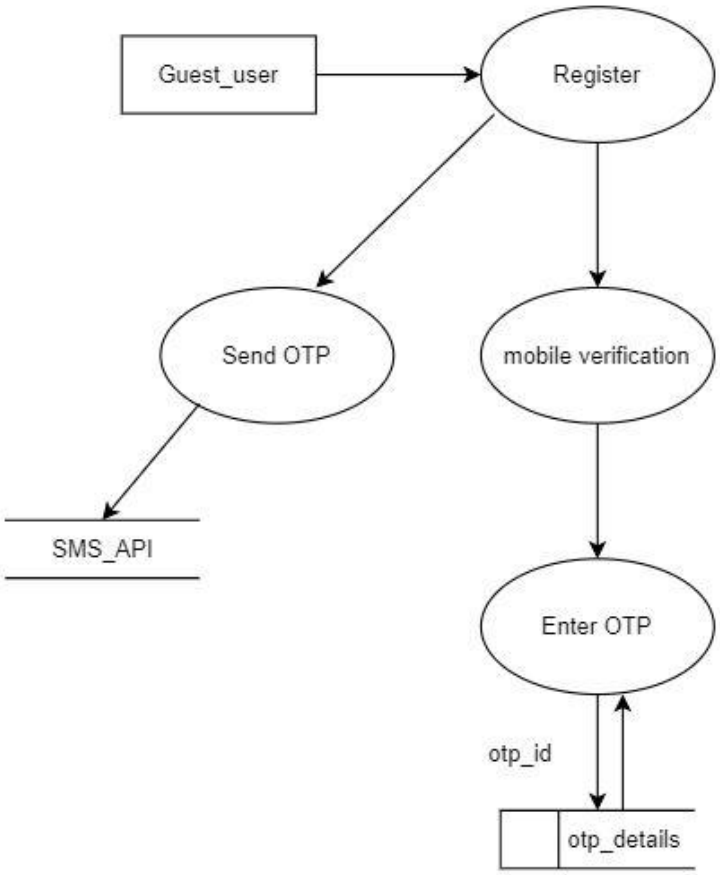
1.3


Guest_user
TEB
index_page
View_construction
materials
view_
images
view_construction
description
image_id
construction
material_id
construction_
images
construction_
materials

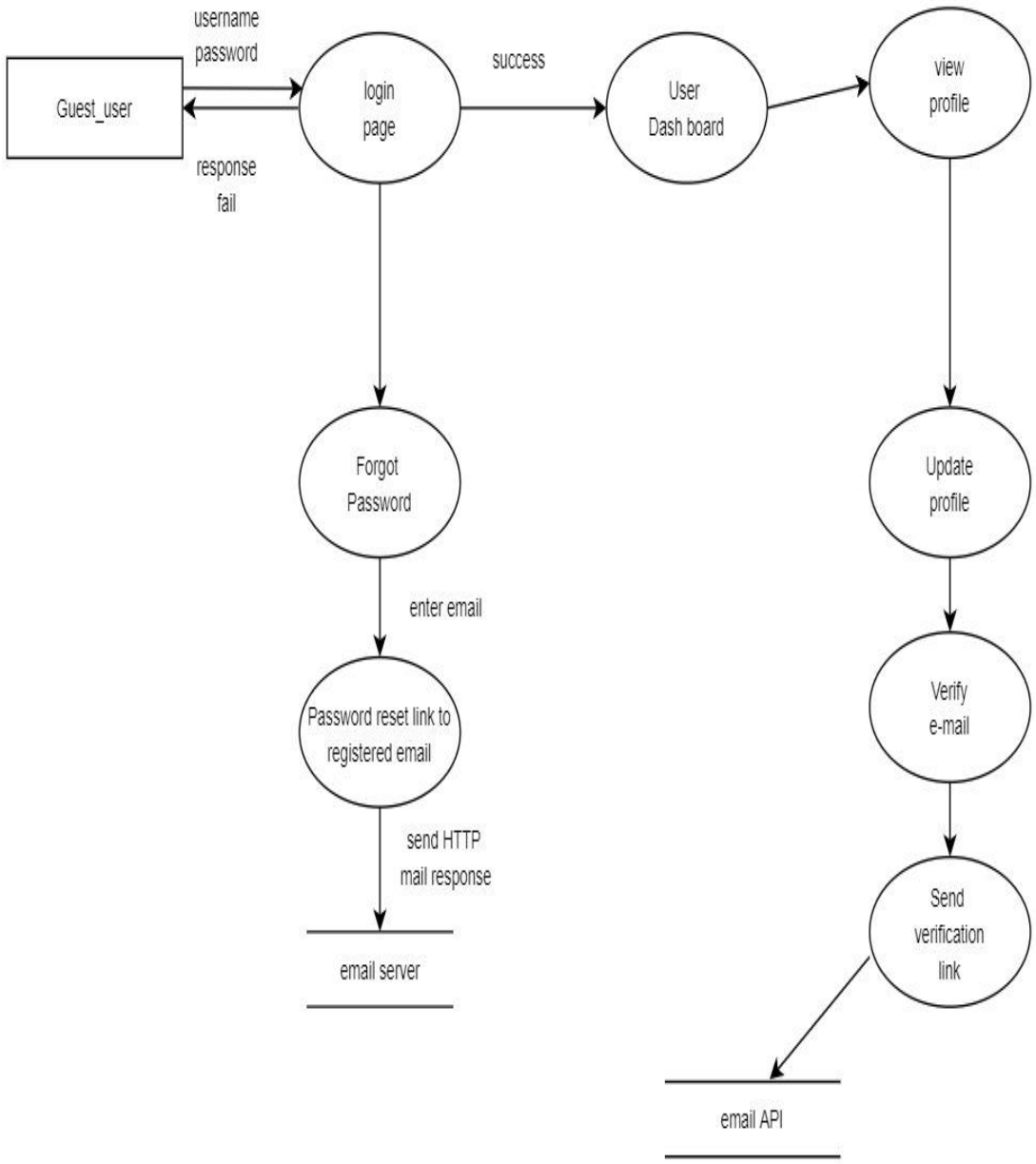
1.4


Guest_user
TEB
Index_page
view services
list services provided
service_id
Services

1.5

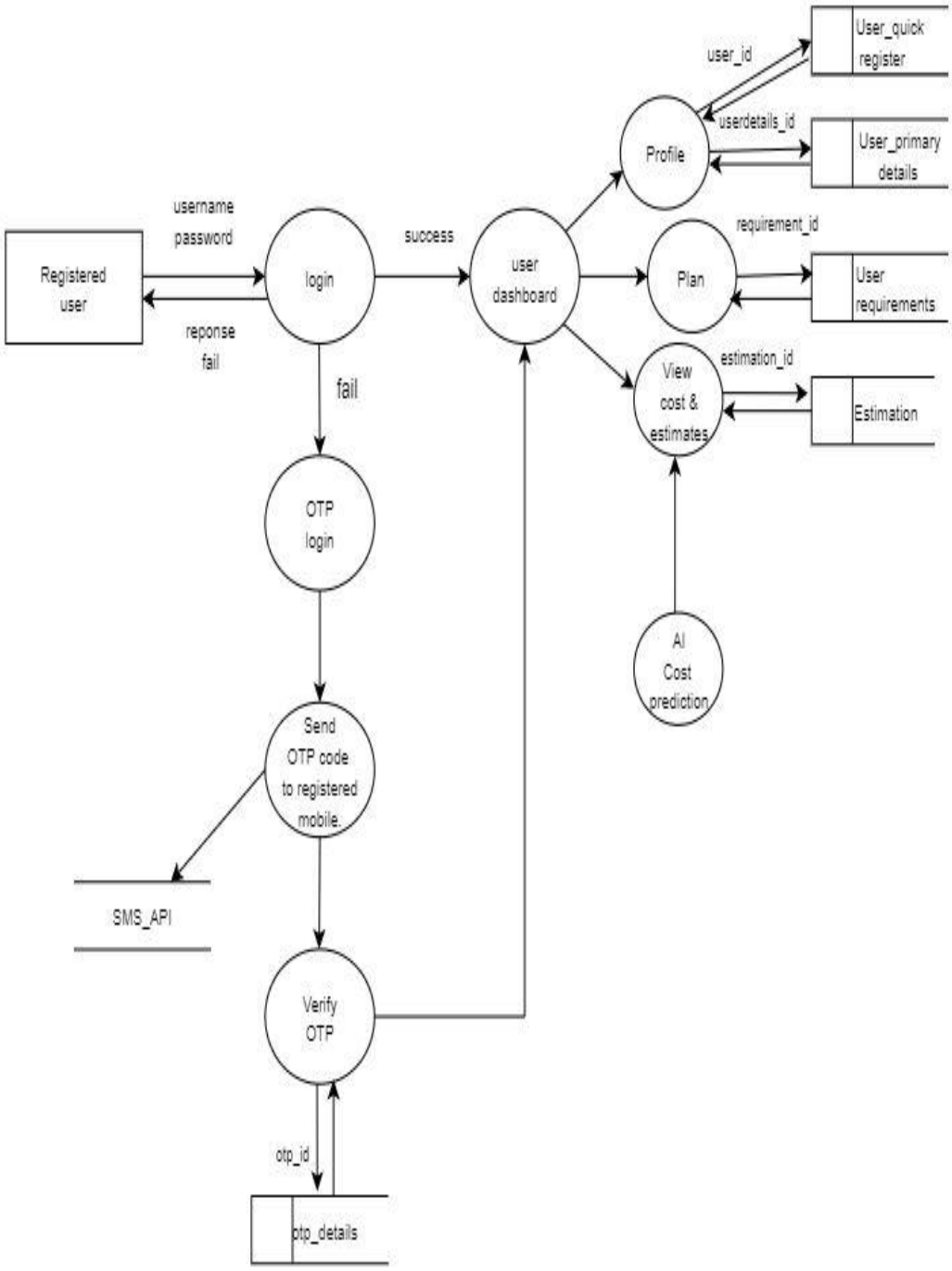

Guest_user
Register
Send OTP
mobile verification
SMS_API
Enter OTP
otp_id
otp_details

1.6

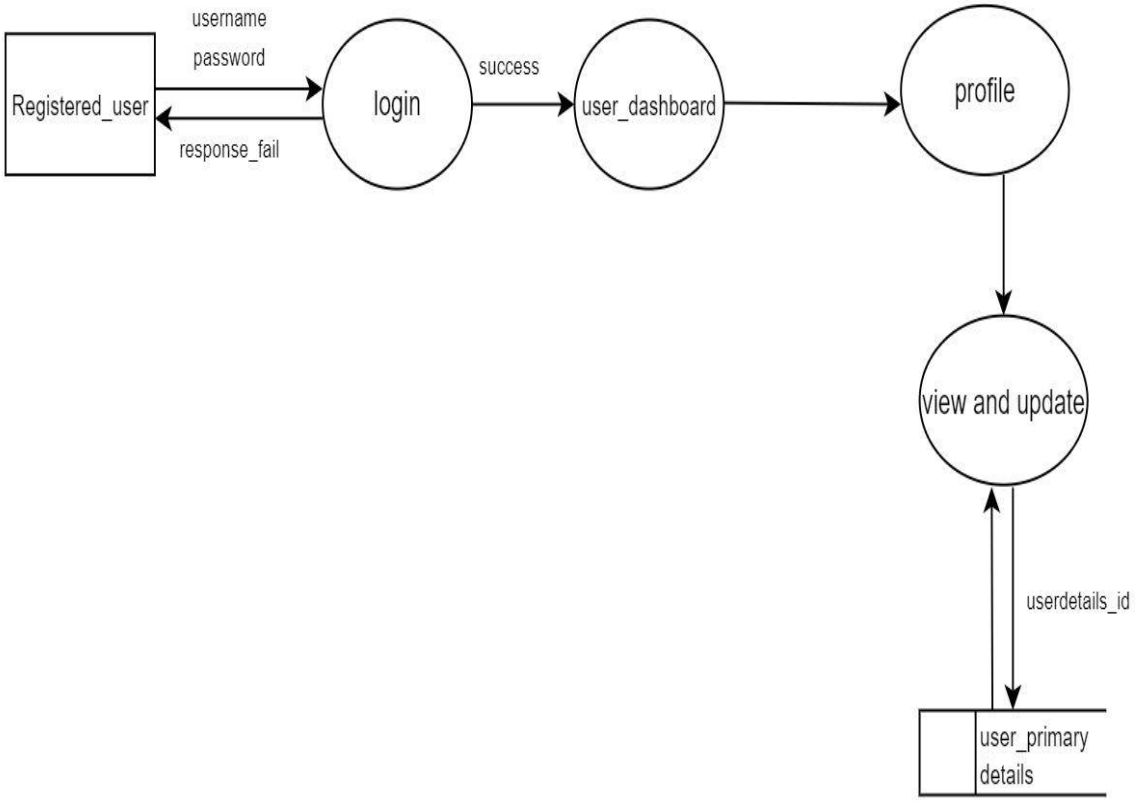

username
password
Guest_user
login
page
success
User
Dash board
view
profile
response
fail
Forgot
Password
Update
profile
enter email
Password reset link to
registered email
Verify
e-mail
send HTTP
mail response
Send
verification
link
email server
email API

Registered user

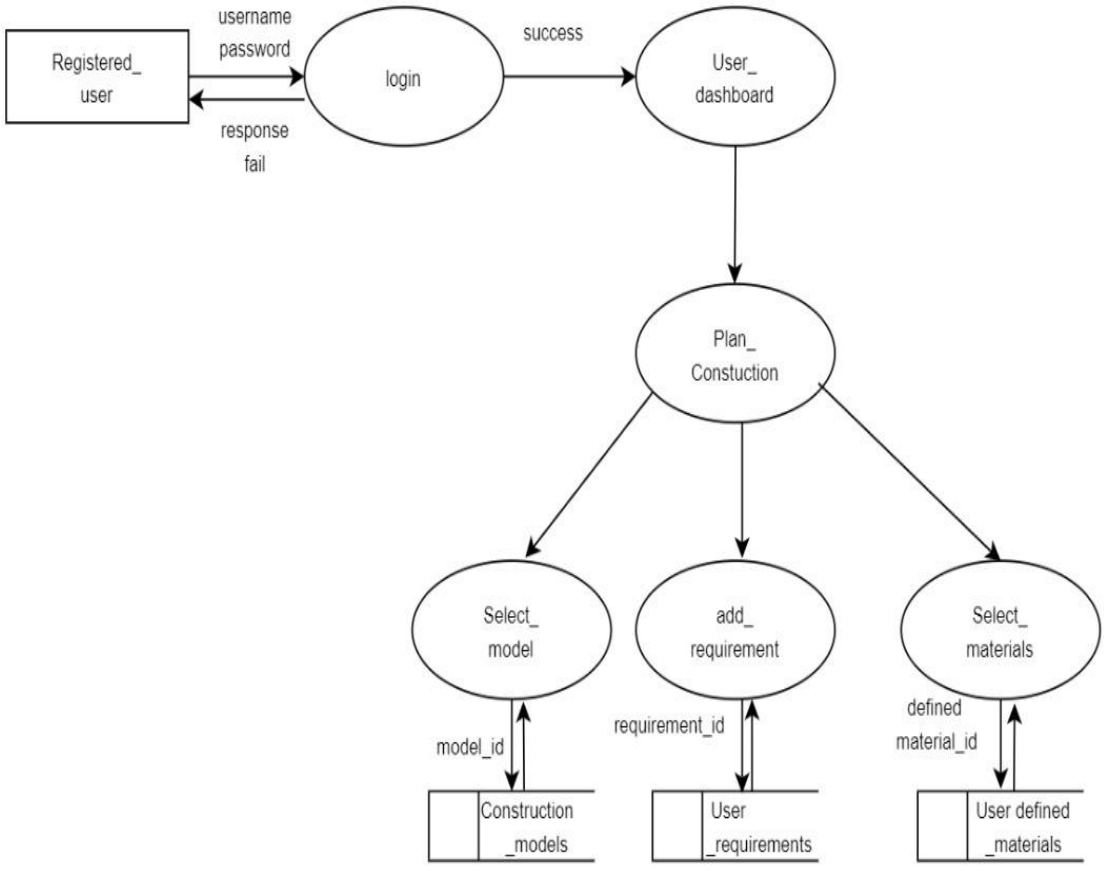
Level 2 of Registered_user


Registered user
username
password
reponse
fail
login
success
user dashboard
Profile
user_id
userdetails_id
User_quick register
User_primary details
Plan
requirement_id
User requirements
View cost & estimates
estimation_id
Estimation
AI Cost prediction
fail
OTP login
Send OTP code to registered mobile.
SMS_API
Verify OTP
otp_id
otp_details

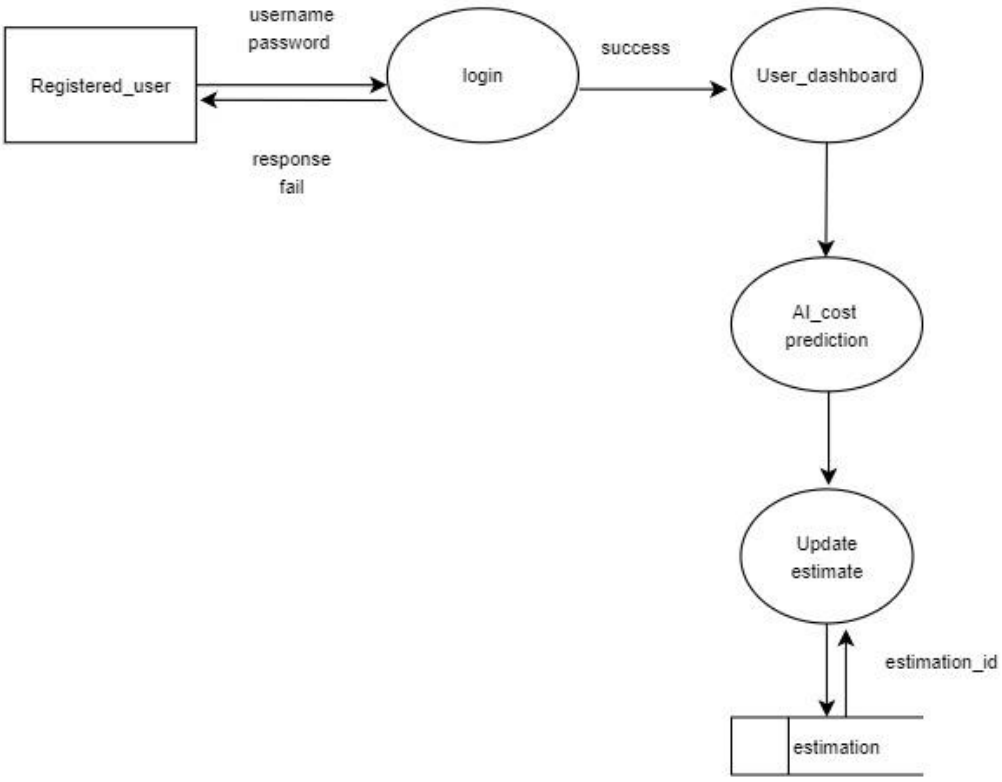
2.1


username
password
Registered_user
login
success
user_dashboard
profile
response_fail
view and update
userdetails_id
user_primary
details

2.2


Registered_
user
username
password
response
fail
login
success
User_
dashboard
Plan_
Constuction
Select_
model
add_
requirement
Select_
materials
model_id
requirement_id
defined
material_id
Construction
_models
User
_requirements
User defined
_materials

2.3


Registered_user
username
password
login
success
User_dashboard
response
fail
AI_cost
prediction
Update
estimate
estimation_id
estimation

5.2 Database Design

The overall objective in the development of database technology has been to treat data as an organizational resource and as an integrated whole. DBMS allow data to be protected and organized separately from other resources. Database is an integrated collection of data. The most significant form of data as seen by the programmers is data as stored on the direct access storage devices. This is the difference between logical and physical data.

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 independence.

The proposed system stores the information relevant for processing in the SQL Lite database. This database contains tables, where each table corresponds to one particular type of information. Each piece of information in table is called a field or column. A table also contains records, which is a set of fields. All records in a table have the same set of fields with different information. There are primary key fields that uniquely identify a record in a table. There are also fields that contain primary key from another table called foreign keys.

A good database design starts with a list of the data that you want to include in your database and what you want to be able to do with the database later on. This can all be written in your own language, without any SQL.

Database name:TEBDB

Table no: 5.3.1

Table Name:**userquick_register**			Primary Key:user_id
Description:Registration of user.			Number of fields:6
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
user_id	Int	Primary key	Id of the user.
user_email	Varchar(50)		E-mail of the user.
user_mobile	Bigint		Mobile number of the user.
user_postalcode	Int		Postal code of the user.
user_name	Varchar(50)		Name of the user.
user_fname	Varchar(50)		First name of the user.

Table no:5.3.2

Table Name:**user_login**			Primary Key:userlogin_id
Description: Login of user.			Number of fields:6
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
userlogin_id	Int	Primary key.	Login id of the user.
userlogin_uname	varchar(50)		username

userlogin_upass	varchar(50)		Password of the user
userlogin_device	varchar(50)		Device of the user
userlogin_login key	varchar(50)		Login key of the user.
userlogin_status	varchar(50)		Login status of the user.

Table no: 5.3.3

Table Name:**Project Portfolio**			PrimaryKey:portfolio_id
Description:Details of project portfolio			Number of fields:8
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
portfolio_id	Int	PrimaryKey	Id of the portfolio
portfolio_name	varchar(50)		Name of the portfolio
portfolio_description	varchar(50)		Description of the portfolio
portfolio_location	varchar(50)		Location of the portfolio
portfolio_date	varchar(50)		Date of the portfolio
portfolio_client	varchar(50)		Portfolio of the client
portfolio_category	varchar(50)		Category of the portfolio
portfolio_sqft	varchar(50)		Square Feet description

Table no: 5.3.4

Table Name:**Portfolio_album**			PrimaryKey:album_id
Description: Details of portfolio album			Number of fields:6
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
album_id	Int	PrimaryKey	Id of the album
portfolio_id	Int	Foreign Key	Id of the portfolio
album_name	varchar(50)		Name of the album
album_desc	Text		Description of the album
album_date	varchar(50)		Date of the album
album_status	varchar(50)		Status of the album

Table no: 5.3.5

Table Name:**portfolio_gallery**			PrimaryKey:galleryimage_id
Description:Details of gallery			Number of fields:6
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
galleryimage_id	Int	PrimaryKey	Image id of portfolio gallery
galleryimage_name	varchar(50)		Image name
album_id	Int	Foreign key	Album id of portfolio album
galleryimage_path	varchar(50)		Image path of image gallery
galleryimage_type	varchar(50)		Type of image gallery

galleryimage_status	varchar(50)		Status of the image gallery

Tableno:5.3.6

Table Name:**Construction_materials**		PrimaryKey:Construction material_id	
Description:Details of of construction materials.			Number of fields: 9
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
Construction material_id	Int	PrimaryKey	Id of the construction material
Construction material_type	varchar(50)		Type of the construction material
Construction material_name	varchar(50)		Name of the construction_material
Construction material_dose	varchar(50)		Dose of the construction_material
Construction material_category	varchar(50)		Category of the construction material
Construction material_model name	varchar(50)		Model name of the construction material
Construction material_more info	varchar(50)		More information on construction material
Construction material_status	varchar(50)		Status of the construction material

Tableno:5.3.7

Table Name:**constructionmaterial_ image**		PrimaryKey:image_id	
Description:Details of construction material image			Number of fields:6
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
image_id	Int	Primary key	Id of the construction material image
image_name	varchar(50)		Name of the construction material image
image_category	varchar(50)		Category of the construction material image
image_type	varchar(50)		Type of the construction material image
image_path	varchar(50)		Path of the construction material image
image_extension	varchar(50)		Extension of the construction material image
image_status	varchar(50)		Status of the construction material image
Construction material_id	Int	Foreign key	Id of construction material

Tableno:5.3.8

Table Name:**Construction_model**		PrimaryKey:model_id	
Description:Details of construction model		Number of fields:12	
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
model_id	Int	Primary Key	Id of the construction model
model_name	varchar(50)		Name of the construction model
model_category	varchar(50)		Category of the construction model
model_cost	Int		of the construction model
model_description	varchar(50)		Description of the construction model
model_costunit	varchar(50)		Costunit of the construction model
model_features	varchar(50)		Features of the construction model
model_offers	varchar(50)		Offers of the construction model
model_highlights	varchar(50)		Highlights of the construction model
model_more desc	varchar(50)		
model_status	varchar(50)		Status of of the construction model
model_contractormessage	varchar(50)		Message from contractor

Tableno:5.3.9

Table Name**:userprimary_details**			PrimaryKey:userdetails_id
Description:Details of user primary details			Number of fields:12
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
userdetails_id	Int	PrimaryKey	Id of userdetails
userdetails_location	varchar(50)		Location of userdetails
userdetails_address	varchar(50)		Address of the user
userdetails_profession	varchar(50)		Profession of the user
userdetails_city	varchar(50)		City of the user
userdetails_whatsapp number	varchar(50)		Whatsapp number of the user
userdetails_gender	varchar(50)		Gender of the user
userdetails_dob	varchar(50)		Dob of the user
userdetails_requirement	varchar(50)		Requirements of the user
userdetails_message	varchar(50)		Message of the user

Tableno:5.3.10

Table Name:**user_requirements.**			PrimaryKey:requirement_id
Description:Details of user requirements			Number of fields:14
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
requirement_id	Int	PrimaryKey	Id of the user requirements
mode_id	Int		Id of user requirements
requirement_desc	Text		Description of user requirements

requirement_square feet	varchar(50)		Square feet of user requirements
requirement_location	varchar(50)		Location of the user requirements
requirement_latitude	varchar(50)		Latitude of the user requirements
requirement_longitude	varchar(50)		Longitude of the user requirements
requirement_construction type	varchar(50)		Type of Construction
requirement_construction category	varchar(50)		Category of construction
requirement_plottype	varchar(50)		Plottype of construction requirement
requirement_plotarea	varchar(50)		Plot area of the requirement
requirement_transportation type	varchar(50)		Transportation type of the requirement
requirement_roadaccess	varchar(50)		Roadaccess of the requirement
requirement_plotlocation	varchar(50)		Plot location of the requirement

Table no:5.3.11

Table Name:**userdefined_materials**			PrimaryKey:definedmaterial_id
Description:Details of userdefined materials			Number of fields:12
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
definedmaterial_id	Int	PrimaryKey	Id of defined material
construction material_id	Int		Id of the construction material
definedmaterial_note	varchar(50)		The material defined
definedmaterial_message	varchar(50)		Message of the designed material
requirement_id	Int	Foreign key	Id of the requirement

Tableno:5.3.12

Table Name:**news**			PrimaryKey:news_id
Description:Details of news			Number of fields:9
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
news_id	Int	PrimaryKey	Id of the news
news_categor y	varchar(50)		Category of the news
news_datetim	varchar(50)		Date and time of the news

e			
news_heading	varchar(50)		Heading of the news
news_desc	Text		Description of the news
news_image	varchar(50)		Images of news
news_status	varchar(50)		Status of the news
news_priority	varchar(50)		Priority of the news
news_index	varchar(50)		index of news

Tableno:5.3.13

Table Name:**Otp_details**			PrimaryKey:otp_id
Description:Details of otp			Number of fields:8
FIELD NAME	**DATATYPE**	**CONSTRAINT**	**DESCRIPTION**
otp_id	Int	PrimaryKey	Id of the otp
otp_date	varchar(50)		Date of the otp sent
otp_time	varchar(20)		Time of the otp sent
otp_status	varchar(20)		Status of the otp
otp_validfor	varchar(20)		Otp valid for
otp_value	varchar(50)		Value of the otp
otp_category	varchar(50)		Category of the otp
user_id	Int	Foreign key	Id of the user

5.3 Input Design

Data design creates a model of data and or information that is represented at a high level of abstraction. The structure of data has always been an important part of software design. The data design activity translates these elements of requirement model into data structure at the software component level. In actuality, the design of data begins during the creation of the analysis model.

In this project all the fields are validated. If any field then error message will be displayed, so as to help the user while giving inputs. The drop down lists are used to reduce the user inputs and to select a preferred item from the list easily. Check boxes are used for user's category selection. User just needs to click the preferred category from the checkbox list. User will select one of the items in list boxes. The following design guidelines will result in a friendly and deficient interface. Minimize the number of input actions required from user. This can be accomplished by using the mouse to select from the predefined set of inputs. In application user can select the options.

The input design is done in such a way that the users of the system will never get confused or enter wrong data. The simplicity and ease of use lies in the act that the desired objectives can be accomplished with a few mouse clicks.

5.4 Output Design

A quality output is one, which meets the requirement of the end user and presents the information clearly. In any system results of processing are communicated to the user and to other system through output. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output designs improve the system's relationship to help user decision-making.

Output forms are the forms where the respective module. We may also see the outputs in reports. The reports often refresh for every change in the database. We can create reports by using SQL queries. Outputs form the system are requires to communicate the results of processing to users. Formats of outputs are defined during output design. The success of the system depends on how well the output reports are generated.

PROGRAM DESIGN

i. GUEST_USER

Step 1: Start

Step 2: The guest user has the privilege to view company portfolio, trends and news, construction materials and services.

Step 3: Once logged in the guest user can maintain a dashboard

Step 4:Guest user can view profile

Step 5: Guest user can update the profile

Step 6: Stop

ii.REGISTERED_USER

Step 1: Start

Step 2: Registered user can view and update user primary details

Step 3: Registered user can maintain a dashboard and can plan construction by selecting a model

Step 4:Registered user can add requirements

Step 5: Registered user can select requirements

Step 6:Registered user can update estimation using AI cost prediction

Step 6: Stop

CHAPTER 6

FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

6.1 Functional Requirements

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Generally, functional requirements are expressed in the form "system must do requirement ".

Functional requirements for each of the uses cases described below:

i. The system shall have options from which the users can view all experts and read articles without login.
ii. The system shall provide functionality for the trainer to train the students by using virtual learning assistants.
iii. The system provides entire details of the registered user and experts at user homepage.
iv. The system shall provide the admin to control all other users.

6.2 Non-Functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. Non-functional requirements are “system shall be requirement ". Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes”, “quality goals", "quality of service requirements" and "non-behavioural requirements.

Some of the non-functional requirements are mentioned below:

i. **Usability**: The system shall have a clean interface with only needed features, clear terminology and tool tips wherever necessary. Warnings or alerts shall be specified in clear way.

ii. **Efficiency**: The system shall respond to different searches being conducted like searching particular product, search quantity, etc. in a very fast way.

iii. **Interoperability**: The system shall be able to interact with other systems. The system should able to be supported at least one software which has a relationship with payment process

iv. **Portability**: The system shall be independent of the specific technological platform used to implement it.

v. **Reliability**: Reliability defined as a measure of the time between failures occurring in a system (measure show frequently the system fails), so that the system shall operate without any failure for a particular period of time

vi. **Availability**: Availability measures the percentage of time the system is in its operational state so that the system shall be available for use 24 hours per day and 365days per year.

CHAPTER 7

TESTING

7.1 Testing Strategies

An engineered product can be tested in one of these two ways. These testing strategies include:

- Black box testing
- White box testing

White box testing

White-box testing is a method of testing the application at the level of the source code. White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are chooses inputs to exercise paths through the code and determine the expected outputs.

Black box testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It is sometimes referred to as specification-based testing.

7.2 Unit Testing

In computer programming, unit testing is a software method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures ,are tested to determine whether they are fit for use intuitively, one can view a unit as the smallest testable part of an application. In procedural programming a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. If forms the basis for component testing.

In the project each module is tested individually and is found to be an error free one.

7.3 Integration Testing

This is the final step in testing. In this case all the modules were combined and given the test data. The combined module works successfully without any side effect on other programs. Everything was found to be working correctly.

In this the entire system was tested as a whole with all modules. This form of testing is popularly known as Black Box testing or system testing.

Black Box testing methods focus on the functional requirement of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external database access, performance errors and initialization errors and termination errors.

In the project each module is tested individually and all the modules are integrated together and the integration testing is carried out for the whole system. The whole system is working accurately without any errors.

7.4 System Testing

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it is vital success of the system.

Testing Objectives:

There are several rules that can serve as testing objectives, they are

- Testing is a process of executing a program with the intent of finding an error
- A good test case is one that has high probability of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered errors.

A test case is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a particular program path or to verify compliance with a specific requirement. Test cases underlie testing that is methodical rather than haphazard. A battery of test cases can be built to produce the desired coverage of the software being tested. Formally defined test cases allow the same tests to be run repeatedly against successive versions of the software, allowing for used to design test cases.

7.5 Testing Results