Software Design Document

for

Ethio health care assistance

Prepare by:-

* Yohannes Girma
* Zenaw Asfaw
* Leykun Ejigu
* Melaku alehilign
* Tewodros Cheru
* Sultan Lombiso

Date created 1/27/2020

TABLE OF CONTENTS

1. INTRODUCTION ………………………………………………………………………………………………1

1.1 Purpose ……………………………………………………………………………………………………..1

1.2 Scope ………………………………………………………………………………………………………….1

1.3 Overview ……………………………………………………………………………………………………….2

1.4 Reference Material ………………………………………………………………………………………..2

1.5 Definitions and Acronyms…………………………………………………………………………….. 2

2. SYSTEM OVERVIEW ………………………………………………………………………………………….3

3. SYSTEM ARCHITECTURE ………………………………………………………………………………………4

3.1 Architectural Design ……………………………………………………………………………………… 4

3.2 Decomposition Description …………………………………………………………………………….5

3.3 Design Rationale …………………………………………………………………………………………11

4. DATA DESIGN ………………………………………………………………………………………………….14

4.1 Data Description …………………………………………………………………………………………14

4.2 Data Dictionary …………………………………………………………………………………………..14

5. COMPONENT DESIGN ………………………………………………………………………………………18

6. HUMAN INTERFACE DESIGN ……………………………………………………………………………19

6.1 Overview of User Interface ……………………………………………………………………………19

6.2 Screen Images ………………………………………………………………………………………………19

6.3 Screen Objects and Actions ……………………………………………………………………………21

7. REQUIREMENTS MATRIX …………………………………………………………………………………… 23

1.0 Introduction

* 1. Purpose

This document is a system and architectural design description of Ethio health care assistance software . It includes the architectural features of the system down through details of what operations each code module will perform and the precise implementation details required to satisfy the requirements as specified in the Software Requirements Specification (SRS) and also the database layout of the system.

This software design document is intended for the developers of the software ethio health care assistance and also for any software developer interested in the design details of the ethio health care assistance software and eager enough to contribute for the future upgrade of the system.

* 1. Scope

- This document is concerned in illustrating the architectural and system details of the software to be developed.

- Ethio health care assistance is a web application that provides and maintains interaction between a patient and a health care centers around Ethiopia. This web app allows patients or users to choose from a list of health care centers and book an appointment at the selected health care centre. It will also allows patient to delete,update and check their appointment. This web app supports uploading,deleting and accessing a patients complete health records for users. This web app will benefit users to book appointment without going to the hospital. It will notify users their appointment schedule.It will allow users to avoid waiting for their appointment at health care center, it will notify the time they should arrive for their time.

In general the objective and goal of the software to be developed can be listed as:

- Allow users to book appointment at a user preferred health care center using any browser and from all over Ethiopia.

- Store patient health record and grant access to patients.

- Encourage users to be aware of their health status.

- Grants health care centers to advertise their services.

- Allow user to pay for services using e-banking systems.

* 1. Overview

This document contains the system and architecture overview, design considerations and system architecture with components and user interface designs. Contents of this document are organized according to the standard of the software design document template.

chapter 2- illustrates the overview of system including functionality, context and design of the software.

Chapter-3 provides high level overview of how the system is decomposed in to modules with a certain task and how they are integrated together to provide the required functionality. Decomposition description and design rationale are described here. it Also contains diagram that show the major subsystems and their interaction.

Chapter -4 This section describes how the major data or system entities are stored, processed and organized. In general it contains the basic data structure design and data dictionary.

Chapter-5 - illustrates the function of each component in a systematic way.

Chapter-6- contains the details about the user interface, screen object and action, screen shots of the system.

Chapter-7 contain the requirement matrix.

* 1. References [https://docs.djangoproject.com/en/3.0/topics/](https://docs.djangoproject.com/en/3.0/topics/templates/)

<https://en.wikipedia.org/>

1.5 Definitions and Acronym's

|  |  |
| --- | --- |
| 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. Ridiculously fast. |
| Bootstrap | Bootstrap is a free and open-sources CSS framework directed at responsive, mobile-first end web development. |
| HTML | Stands for Hyper Text Markup Language, it is used to to design the front portion of web pages using mark up language, which defines the structure web pages. |
| Database | Database is a structured set of data held in a computer, especially one that is accessible in various ways. |
| CSS | Cascading Style Sheets is a simply designed language intended to simplify the process of making web pages presentable.it allows you to apply styles to web pages. |
| Python | Is a programming language that lets you work quickly and integrate systems more efficiently. |
| SDD | Software design document |
| FnREQn | Feature number requirement number |

2.0 System overview

- Most of Ethiopia's health cares struggle to deliver a quality service . The main reasons being inefficiency to serve and treat patients as quick as possible and also not giving adequate information about the diseases and the treatment it needs. From waiting long hours to get a new appointment to receiving a treatment takes a very long time and its unimaginable to wait this long while being sick. This tedious institutions forces people to look for other solutions with out going to the health cares unless its a serious condition. The adverse effect of peoples reluctance to go to health cares due to the backward and tedious work environment of health cares is a health decline to the society. Most of the people forget to be present for their appointment despite its necessity. The main cause being ignorance and despise to the time consuming and pain exacerbating health care services. Although health cares are the most necessary institutions, peoples assumption of health cares in Ethiopia is not really good.

- Mainly the purpose of this project is to minimize the discomfort caused by

Health cares by introducing a more modernized way of handling patients and related things. Our main objective is to build a software program that enables patients and health cares to interact on the internet. To be articulate, the software handles new appointments,settled appointments,specify the location of Health cares, stores patient information on a database etc.

Making new appointments and checking on for settled appointments on the internet saves the unnecessary discomfort in being ill and waiting for appointments. The software allows patients to make appointments with out leaving their house,after setting the new appointment, the software notifies patients to come at a given time. Since most of the patients information is stored on paper, disappearance of files is a big issue. Our software extends to solve this by storing patient information on a secured cloud service and also giving access to both the patient and the health care. we believe that giving information access to patients for the long term allows them to explore more about the diagnosis and treatments. It helps patients to be more aware and take necessary actions. The software will be a website application which is to be developed by Website design and development tools. The users of this software will be the patients and the health care institutions. The patient and the health cares registers on the software on any computer and smart phone. and from abstract point of view, the software handles interaction between the two users.

3.System architecture

3.1 Architecture design

Ethio health care assistance website is client server software developed using Django framework which is based on MVC(model view controller) architecture pattern. As a consequence our software system follows a model view controller software architecture pattern. In general the software contains three high level components which are model,controller and view.

**Model** is the component that interacts with the database that resides in the software system and handles data presentation. It acts as a mediator between the website interface and the database. In technical terms, it is the object which implements the logic for the application’s data domain or handles the data logic.

**View (Template in Django)** is the component that contain user interface logic, to be articulate template is actually the User Interface of the web-application and contains the parts like HTML, CSS and other Front-end technologies like bootstrap.

**Controller(View in Django)** is the component  provides the logic to either handle presentation flow in the view or update the model’s data.The controller handles the user interaction and selects a view according to the model. To summarize, controller takes users Inputs or interactions and makes a request to the model.

To generalize, when a user interacts or request for something on the website the controller(view in Django) component accepts the users request or users input and request a command to the model component which handle the request and returns a data from the database. The controller part receives the data and then transfers the data into the view(template in Django) for rendering and updating the user interface which is handled by the view component and at last the controller(view in Django) receives the rendered user interface and returns it to the user. It must be noted that the view(template in Django) can only interact with the model component directly and models and view are connected through the controller. The term view is used instead of controller in django and the term template is used instead of view.

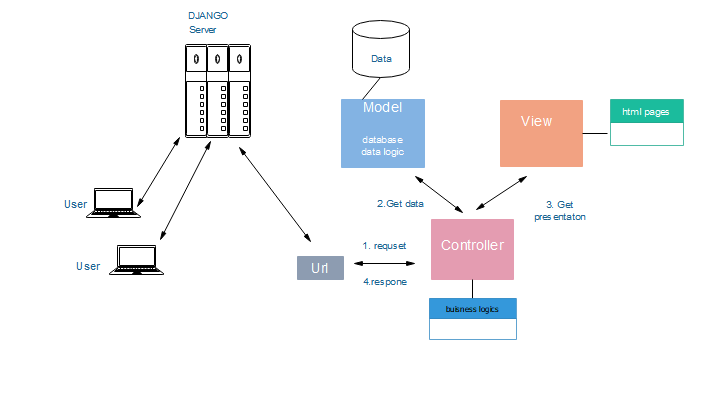


Figure 1.1 architecture model

3.2 Decomposition description

The subsystem described in the architecture design are decomposed in to different modules on the basis of their responsibilities and the work they had to perform.

1.View/template subsystem-  This subsystem contains the static parts of the desired HTML outputs as well as some special syntax describing how dynamic content will be inserted. This subsystem is composed of d/t templates specified for some task.

Home page

- it contains the HTML, css and embedded python code the creates the layout of the homepage

Notification page

- it contains the HTML, css and embedded python code the creates the layout of the login page

Login page

- it contains the HTML, css and embedded python code the creates the layout of the login page.

Sign up page

- it contains the HTML, css and embedded python code the creates the layout of the

Sign up page

Profile page

- it contains the HTML, css and embedded python code the creates the layout of the profile page

Health care page

- it contains the HTML, css and embedded python code the creates the layout of the health care page

Home page for health cares

- it contains the HTML, css and embedded python code the creates the layout of the homepage for health care as a user.

Notification page for health care

- it contains the HTML, css and embedded python code the creates the layout of the notification for health care as a user.

1. Controller/view subsystem

This subsystem is composed of modules that are responsible to responding to the user input and performs interactions on the data model objects. The controller receives the input, optionally validates it and then passes the input to the model.

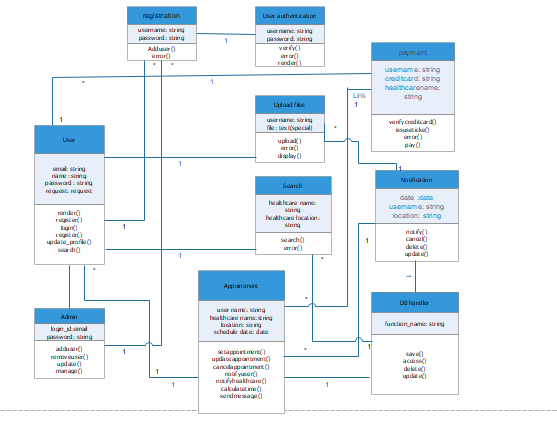


Figure class diagram

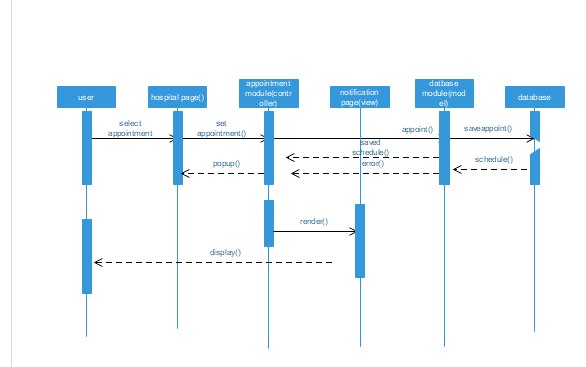


Figure 1. Sequence diagram of appointment

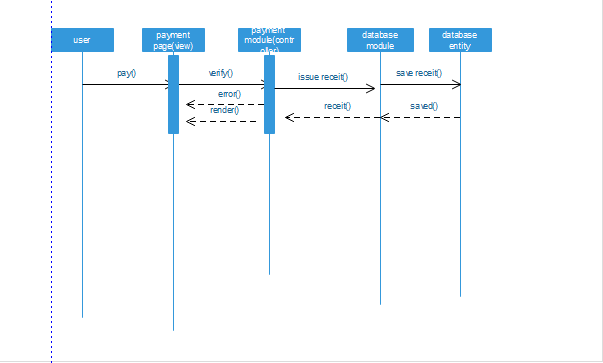


Figure 1. Sequence diagram of appointment

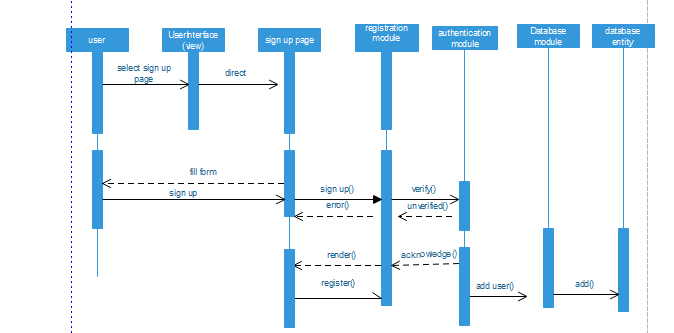


Figure 1. Sequence diagram of appointment

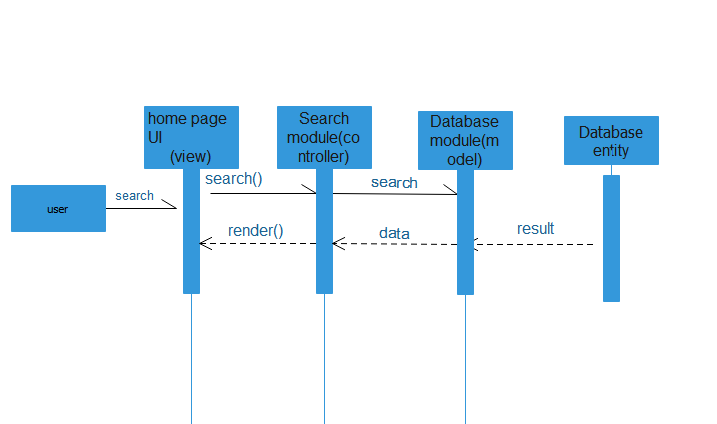


Figure 1. Sequence diagram of appointment

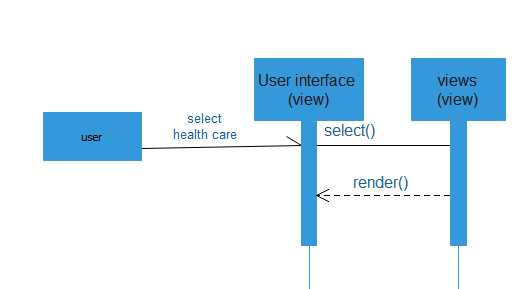


Figure 1. Sequence diagram of appointment

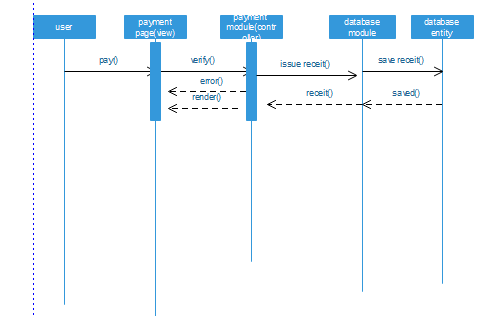
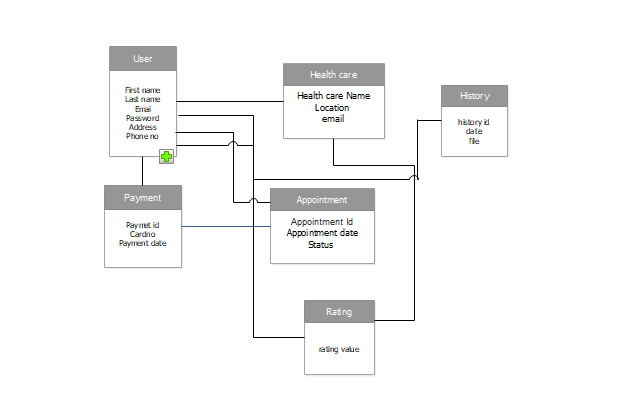


Figure 1. Sequence diagram of appointment

3.0 Model subsystem



3.3 Design rationale

Model view controller architecture style is adopted for this software system. Django framework is used to develop the software system, it follows MVC pattern and

It makes it easy to implement the software designed using MVC. Few others reason for choosing the MVC architecture can be listed as:

* MVC allows multiple developers to work simultaneously on the model, controller and views.
* The very nature of the MVC framework is such that there is low coupling among models, views or controllers
* Due to separation of responsibilities in MVC, future development or modification is easier

4.0 Data Design

4.1 data design

The data in this software system is structured using the built in data structured in python. The framework Django used for this software system provides a range of tools and libraries to transform the information into data-structures. Data definition and processing are handled by the model subsystem of the software. The model is the single, definitive source of information about the data. It contains the essential fields and behaviors of the data to be stored. Generally, each model maps to a single database table. Each model is a Python class

And each attribute of the model represents a database field.With all of this, Django gives you an automatically-generated database-access API from which data can be processed and stored back.

|  |  |
| --- | --- |
| Tables | Description |
| User | Stores data about the user |
| Health care | Stores data about the health care |
| Payment | Stores data about the the transaction and payment |
| rating | Stores values of rating for each health cares |
| Appointment | Stores data about appointment process |
| History | Stores history files of a particular user |

4.2 data dictionary

Appointment table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Appointment date | Date/time | models.DateTimeField() | The date of an appointment |
| Appointment ID | Text | models.CharField(max\_length=N) | Primary key which identify an appointment |
| Health care email | Text | models.EmailField() | References health care table |
| User email | Text | models.EmailField() | Reference user table |
| Payment id | Text | models.CharField(max\_length=N) | References payment table |
| History ID | Text | models.CharField(max\_length=N) | References history table |

Health care table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Health care name | Text | models.CharField(max\_length=N) | The name of the health care |
| Email | Text | models.EmailField() | Primary key which identify a health care. Its the email address of a particular health care |
| Image | Text | models.ImageField() | Image of health care |
| Location | Text | models.CharField(max\_length=N) | Location of the health care |

History table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Appointment ID | Text | models.CharField(max\_length=N) | references appointment table. |
| File | Text | models.ImageField() | History file |
| Health care email | Text | models.EmailField() | References health care table |
| History ID | Text | models.EmailField() | Primary key that identifies history |
| Payment ID | Text | models.CharField(max\_length=N) | References payment table |
| User email | Text | models.CharField(max\_length=N) | References user table |

Payment table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Appointment ID | Text | models.CharField(max\_length=N) | References appointment table |
| Health care email | Text | models.EmailField() | References health care table |
| payment date | Date/time | models.DateTimeField() | The date of payment |
| Payment id | Text | models.CharField(max\_length=N) | Primary key of payment table |
| User email | Text | models.EmailField() | Reference user table |

Rating table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Appointment Id | Text | models.CharField(max\_length=N) | References appointment table and identify rating |
| Hospital email | Text | models.EmailField() | References hospital table and identify rating |
| Value | Number | models.FloatField() | Value given by a user |
| User email | Text | models.EmailField() | References user table and identify rating |

User table

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Data type | Django model type | Description |
| Address | Text | models.CharField(max\_length=N) | Address of a user |
| Email | Text | models.EmailField() | Primary key which identify a user. Its the email address of a particular user |
| First name | Text | models.CharField(max\_length=N) | First name of a user |
| Last name | Text | models.CharField(max\_length=N) | Last name of a user |
| Image | Text | models.ImageField() | Image of user |

5.0 component description

1. User module- A module for a user object, contains the framework for the user object to be created.

Create{

Username

Password

Name

Register }

1. Authentication module- A module that handles both authentication and authorization of users .

Pseudo-code:

Get Username Get Password

IF FILE EXIST THEN

READ Password FROM FILE

IF FILE. Password == Entered Password

Login successful

ELSE

PRINT “incorrect Username or password “

END IF

ELSE

PRINT “incorrect Username or password “

END IF

1. Admin module- Admin module allows system administrator to set up back-end of the system and perform basic system configuration.

Pseudo-code:

IF required privileges

{

IF exiting account

{

IF select remove user

{

Confirm

}

ELSE select Manage User

}

IF Suspend account

{

Enter relevant details

Confirm

}

ELSE

{

Create account

}

}

ELSE

Print “Required Privileges not given “

EXIT

1. Upload module- A module used to update and delete file.

While Upload file

{

Add details

Remove details

}

Save changes

EXIT

1. Appointment module- A module to set appointment at a particular health care.

If set appointment{

Request for appointment If{

User==verified

Calculate schedule

print a schedule

}

Else{

Print error

}

}

Exit

1. Search module- A module that takes user input and search for the required data and returns a result.

If value{

Value exist

Print value

}

Else

{

Print no search history

}

Exit

1. Notification module- Generate a notification when an appointment is set.

if notification{

Display

Delete

close

}

Exit

1. Payment module- A module that take care of transaction between the user and a particular health care

While payment{

Insert credit card

If credit card is valid{

Issue ticket

}

Else{

}

}

6.0 Human Interface design

6.1.1 Overview of user interface

The user interface is composed of web pages, each page is responsible for some task. The web pages are home page, profile page, notification page, contact, sign up, log in,health care page, about page.

A user first enters a home page which is the main page the user enters from a web search engine, it contains navigation bar which contains links, from which a user navigates to other pages of the website. It also contains the health care list each health care have their own form(Section) on the body of the page,the form contains images,label(name of the health care) and a rating button which a user use to rate the health care, when a user select a certain form or health care it is directed to the homepage of the selected health cares. A search button on the homepage is used to search health cares. Most searched health cares are displayed on body of the home page. Services offered by the website are displayed on the bottom part of the home page. A footer section of the home page contains quick links and about information.

A user must login in to the system to use the features of the system. A user navigates to the login page through the navigation links. A login page contains a form which contains password and email text fields, links to a sign up pages and a login button, a user fills the text field with the appropriate information and hit the login button. if the information is verified the user will be navigated to the home page. Else it would be requested to try again.

when a new user visits the website, there will be a request to sign up on the sign up page in-order to use the features of the website. A sign up page is a from which contains text fields which a user uses to insert personal information like name, email etc. After filling the information a user hit the sign up page. If the information inserted is verified the user will be navigated to the homepage.

When a user selects a health care from the homepage, user will be navigated to the health care page. Health care page contains a navigation bar which allows to user to navigate into different section of the page. The body of the homepage contains a layout that display services offered by the health care and it contains a form that allow users to make an appointment. A user enters some information on the text field of this form and hit the appointment button to book an appointment.

When a user book an appointment , the user will be notified with the schedule of the appointment. The notification page contains schedule of the appointment and allows canceling and update of appointment schedule.

A user have a profile page which displays information about the user. Upload page contains form with upload button, it allows users to upload their history file.

6.2 Screen Images

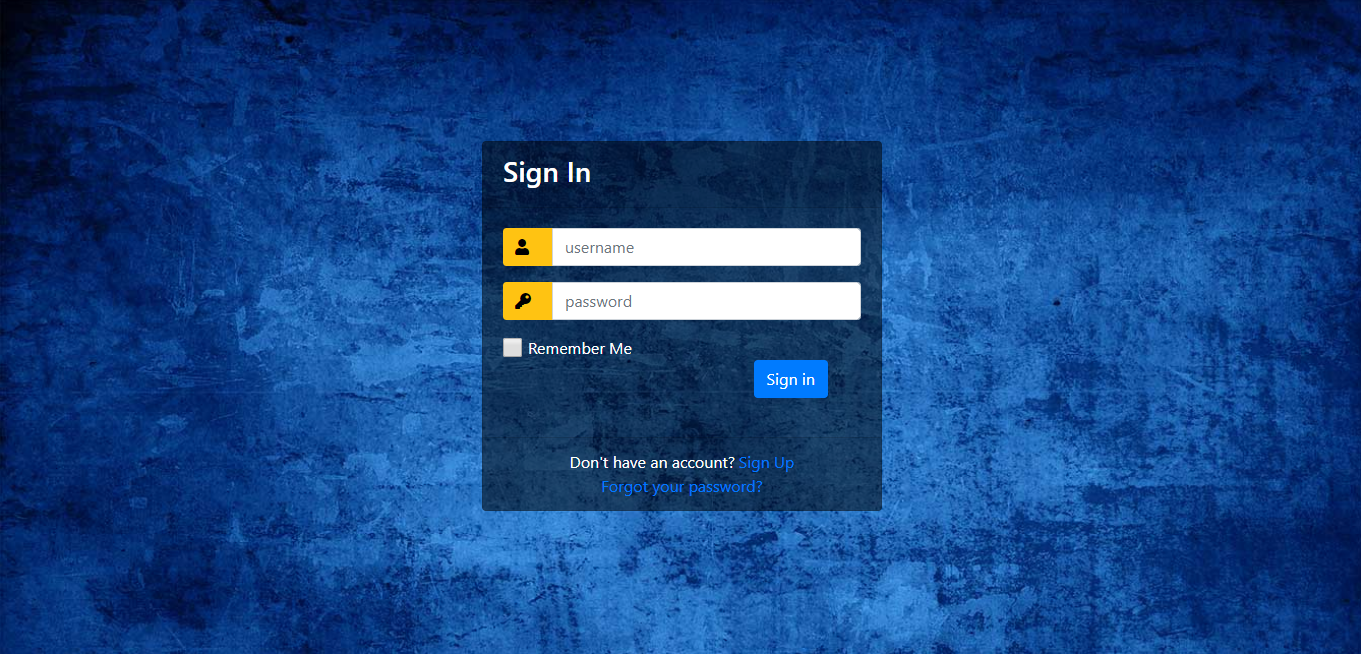


Figure 1.1 sign in page

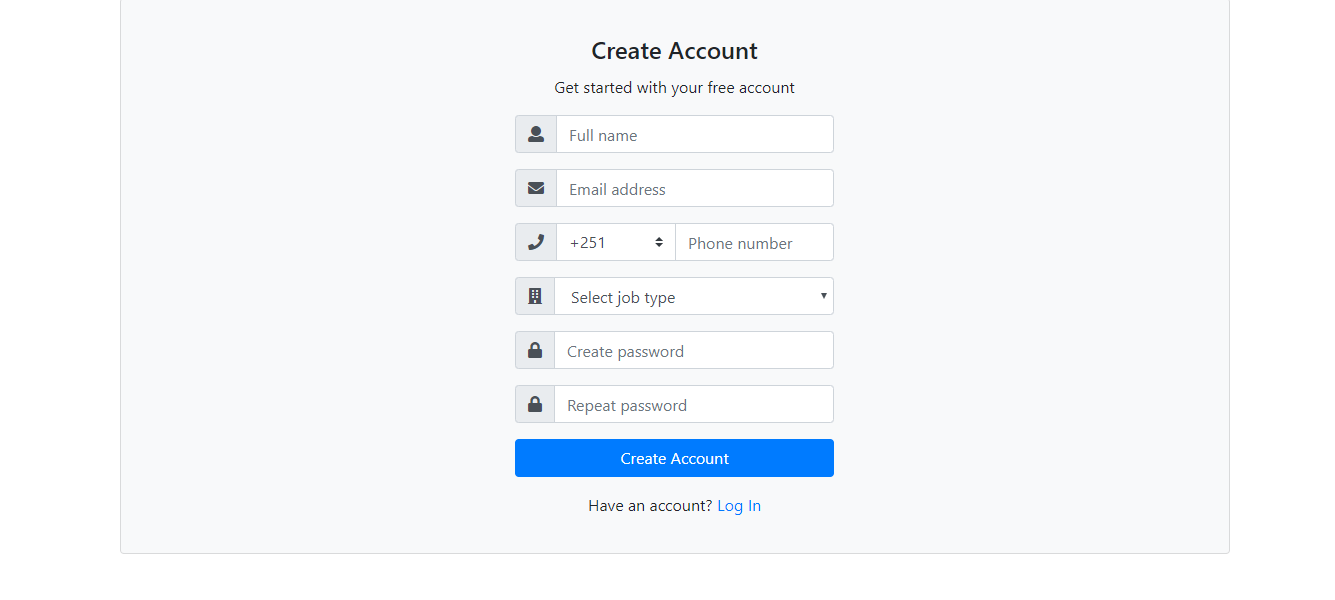
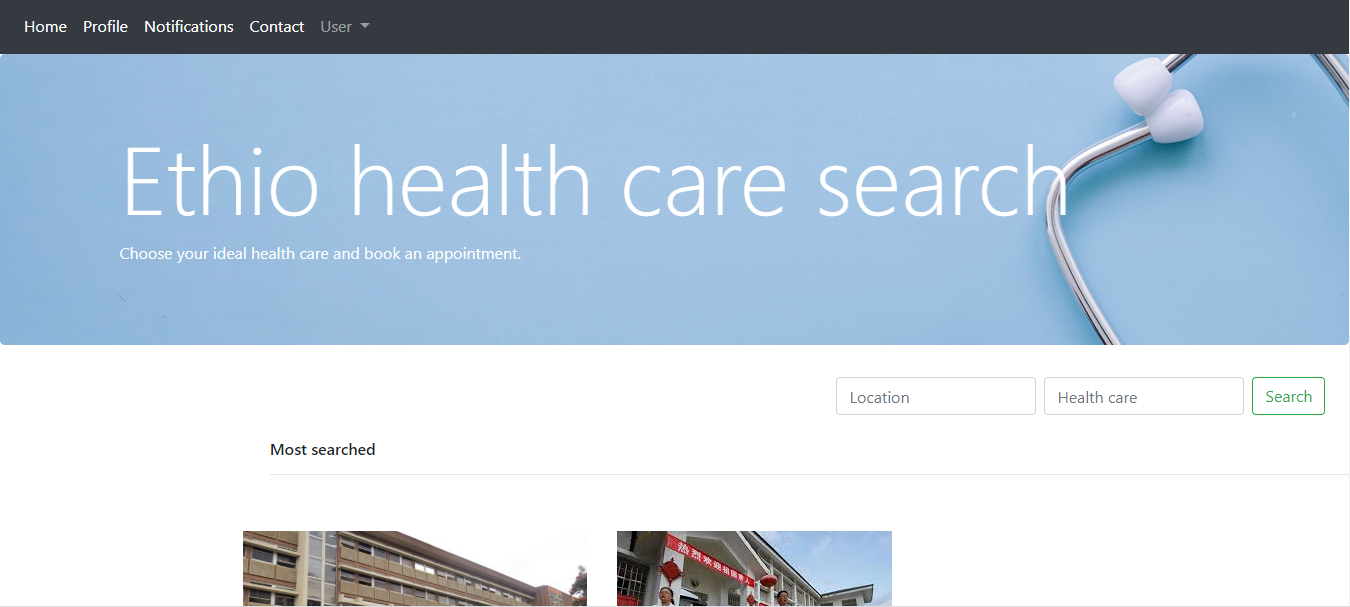


Figure 1.2 sign up page



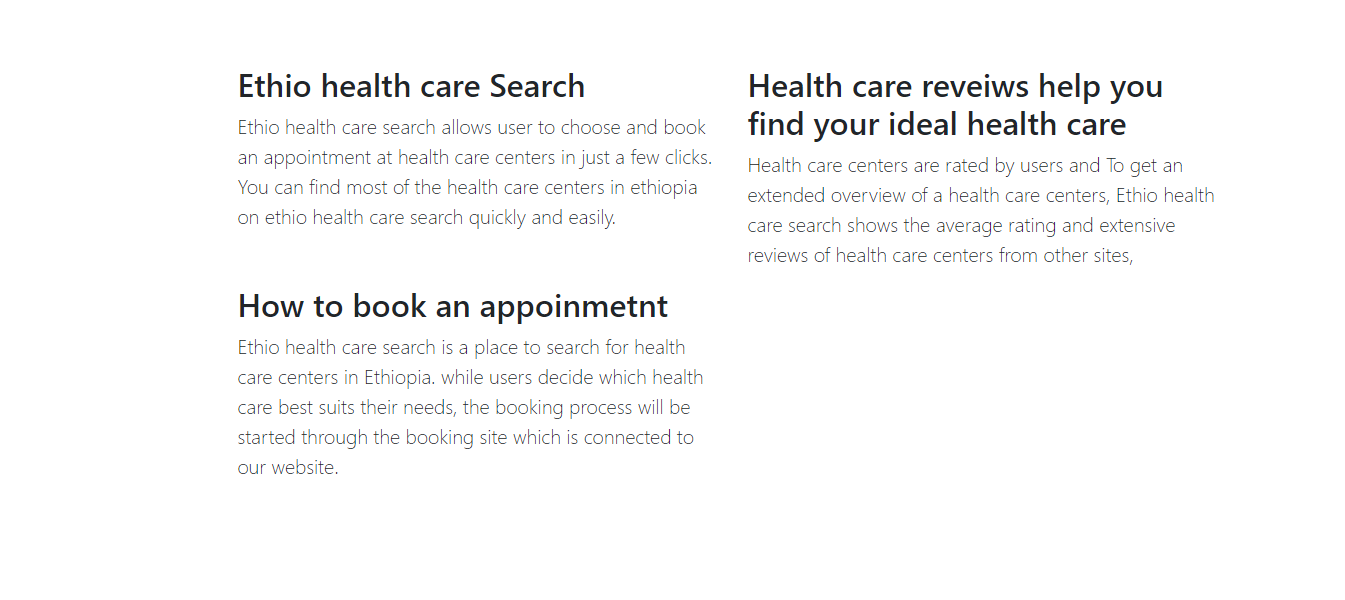
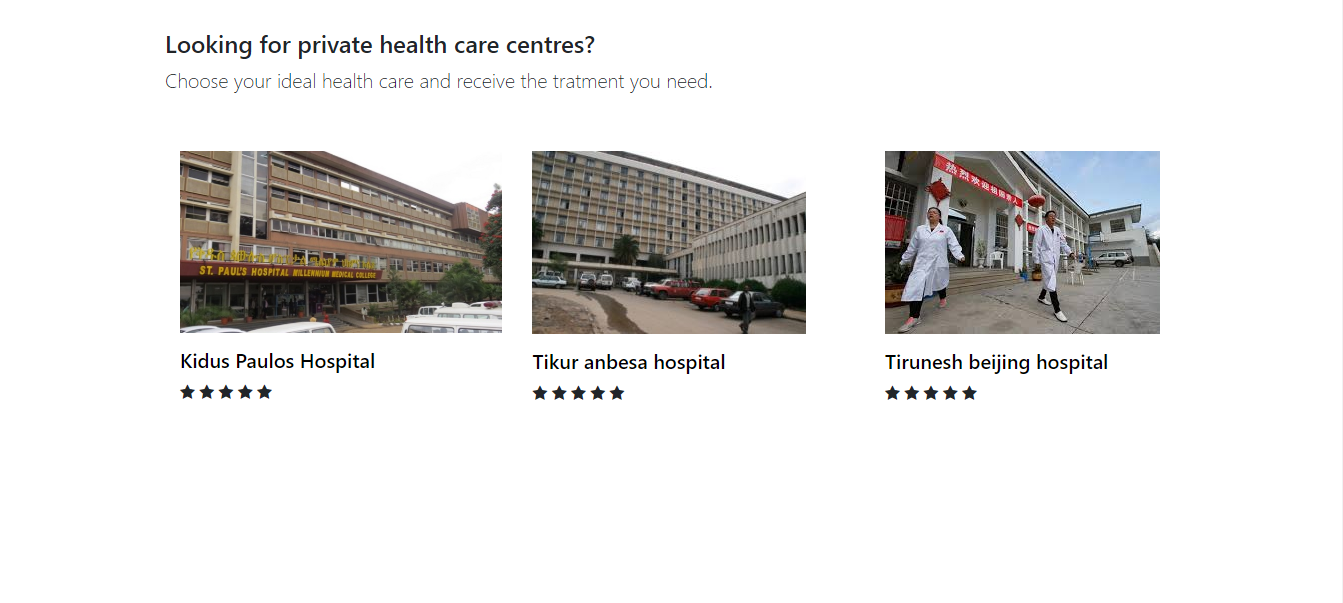


Figure 1.3 home page

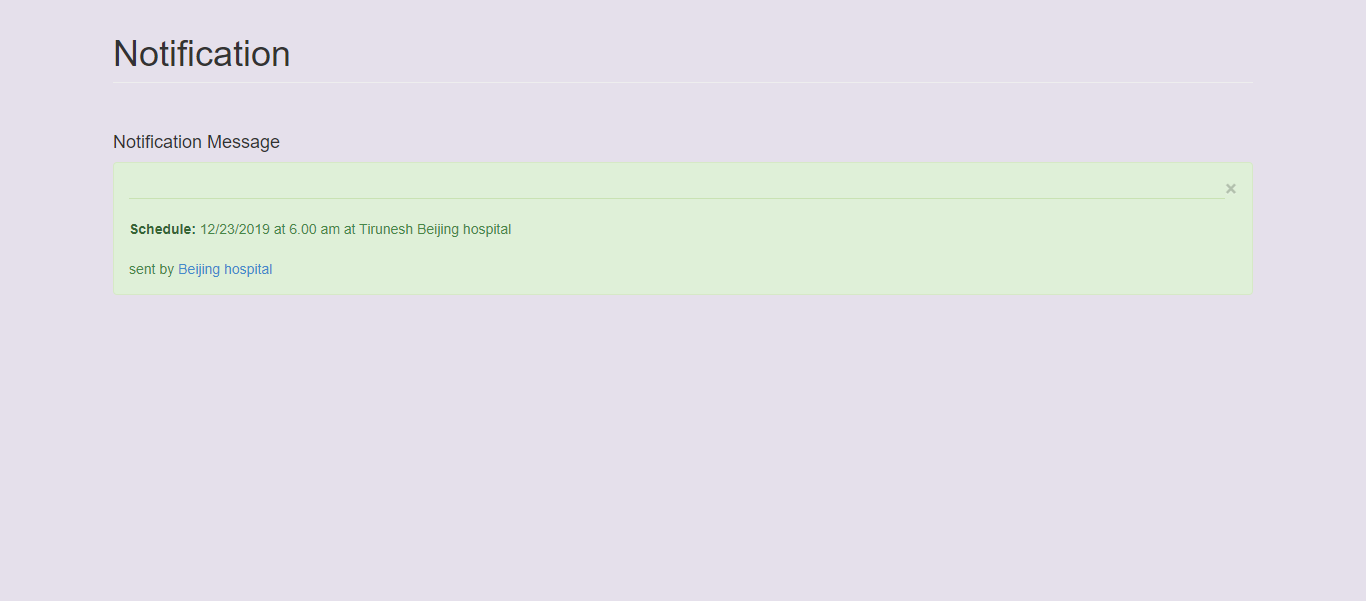


Figure 1.5 notification page

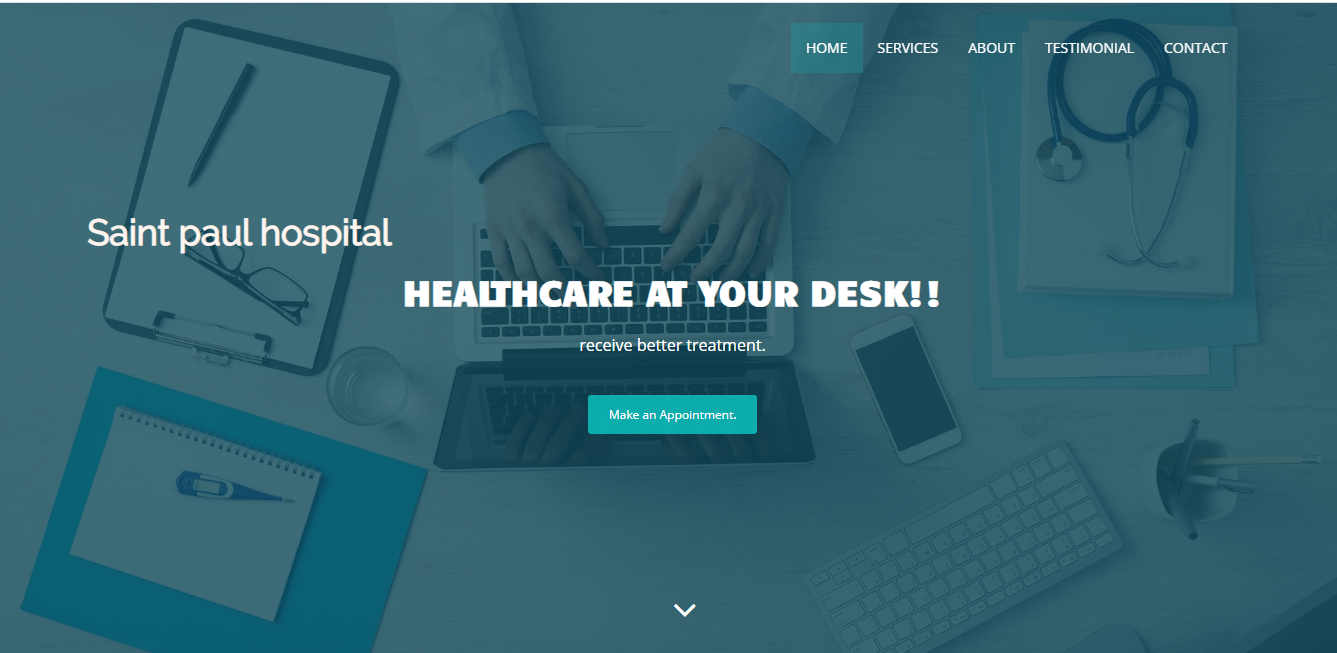


Figure 1.7 health care pages

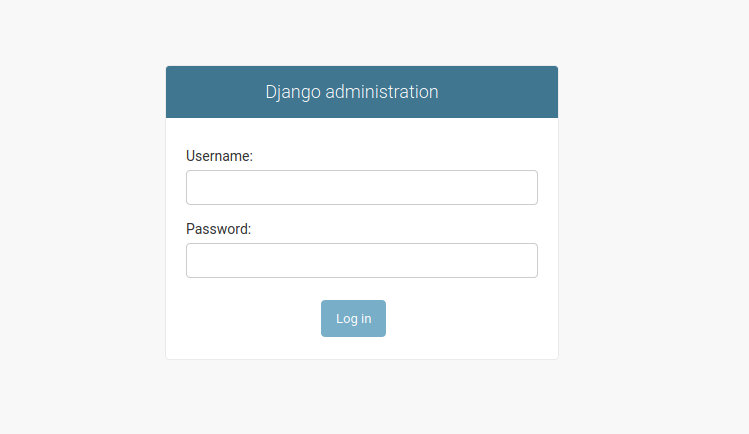
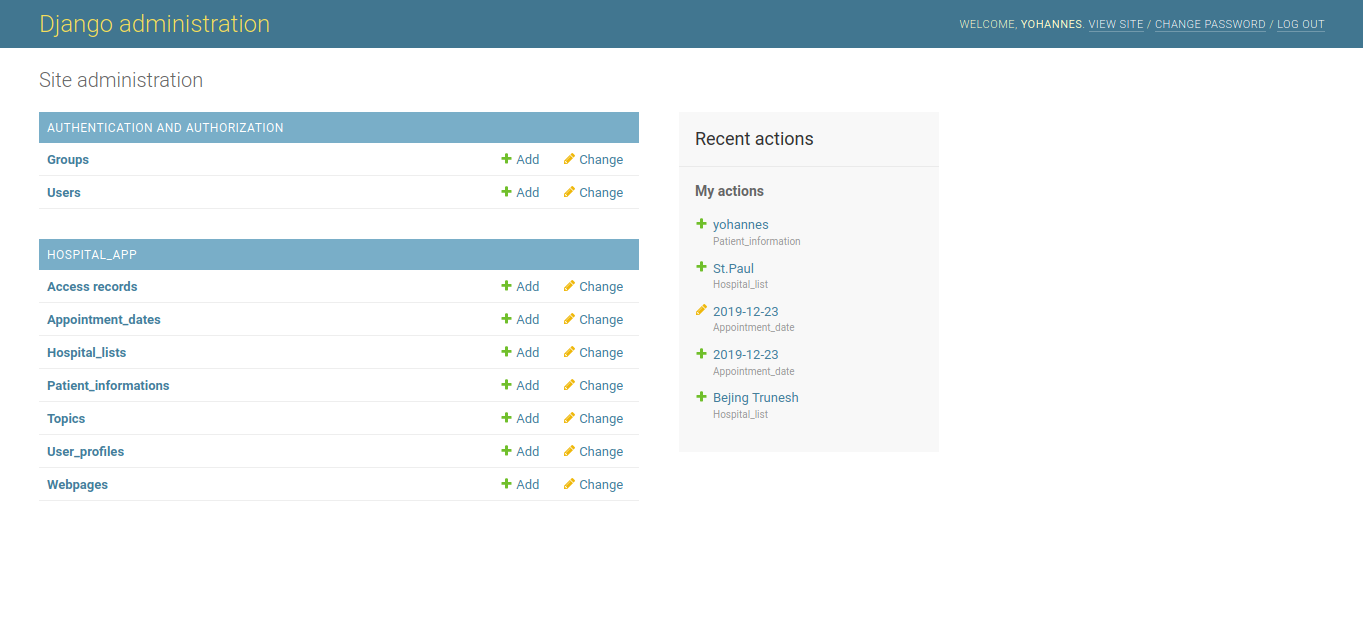
Figure 1.8 admin log in page

Figure 1.9 admin page

6.3 Screen Objects and Action

1. Login page

* Username label and text filed- allow user insert their user name
* Password label and text filled- allow user insert their password
* Sign in button- direct a user into a home page if information is verified.
* Check list- allow users device to remember the users information.
* Links- allow user to navigate to other pages

1. Sign up page

* Text- display text
* Full name label and text field- allow users insert full name
* Email address- allow users insert email address
* Create password- allow users insert new password
* Repeat password -allow user repeat the new password
* Links-allow users navigate to sign in page

1. Home page

* Navigation bar- displays the links that allow user to navigate to other pages.
* Jumbotron - displays logo of the website and other texts
* Search text field and button- allow user to search their desired health search
* Cards- contain the image, name of the health care center and buttons. They allow act as links to navigate to the homepage of the health care.
* Rating button- display the rating of a health care and allow user to rate a health care.
* Paragraph section- displays texts about the service and how to use the service
* Footer section- displays quick links and contact information

1. Home page of health cares

* Navigation bar- displays the links to different section of the page
* Jumbotron- displays the logo and texts of the health care
* Appointment button- links to the appointment section
* Cards- display the image of doctors and their name
* Paragraph section- displays texts about service offered by the health cares.
* Query form

- name label and text field- allow users to fill their name

- email label and text field- allow user insert their emails

-message label and text field- allow users to send message to health care

-send message button- allow user to book an appointment and send message to health care

* Footer section- displays contact info

1. Notification page

* Card- displays the schedule of the appointment
* Cancel button- allows user to cancel an appointment
* Update button- allows user to update the appointment

1. profile page

Name label- displays the users name

Card - displays the image of a user.

Upload button-directs the user to upload page to upload file

Show history button - direct the user to upload page and displays the files.

1. Upload page

Upload file button- allows user to upload a file

List of file section- allow user to select a file and display it on the screen.

7.0 Requirement matrix

|  |  |
| --- | --- |
| Requirements | Component |
| F1REQ-1 | User module |
| F1REQ-2 | Registration and authentication |
| F2REQ-1 | Appointment module |
| F2REQ-2 | Appointment module |
| F3REQ-1 | Search module |
| F3REQ-2 | Search module |
| F4REQ-1 | Notification module |
| F4REQ-2 | Notification module |
| F5REQ-1 | Upload module |
| F5REQ-2 | Upload module |
| F5REQ-3 | Upload module |
| F5REQ-4 | Upload module |
| F6REQ-1 | Payment module |
| F6REQ-2 | Payment module |
| F6REQ-3 | Payment module |
| F7REQ-1 | Authentication module |
| F7REQ-2 | Authentication module |
| F7REQ-3 | Authentication module |
| F8REQ-1 | Admin module |
| F8REQ-2 | Admin module |

Table requirement matrix