



MBARARA UNIVERSITY OF SCIENCE AND TECHNOLOGY

FACULTY OF COMPUTING AND INFORMATICS

**End of Semester Two Examination for the Degree of Bachelor
of Science in Computer Science**

Course Code: CSC2207

Course Name: Web Application Development

Course Year: Two

Academic Year: 2021/2022

Date: 28th August, 2023

Room: ICS LABIII

Monday, August 28th, 2023

Student Name: AINEMBABAZI CONFIDENCE

Reg Number: 2021/BCS/002

SYSTEM REPORT

1: Introduction

In this report, I provide a comprehensive overview of my web development project, in which I cloned the Corrigan MineHan Heart Center website using Django as the backend framework. This project allowed me to immerse myself in web development, from design replication to functional implementation, while integrating the power of Django. I selected the Corrigan MineHan Heart Center website due to its modern design, interactive features, and potential to demonstrate Django's capabilities. The original website offers a platform for admin to add all the respective services and to check for all the recent actions that have been taking place in the website. It also provides patients opportunities to navigate through the website and make appointments with their favorite doctors by filling the appointment form, phone calling the doctors and even emailing them through the hospital email.

I cloned the Heart Care Website/ application using the Python Django framework version 4.1, SQLite 3, JavaScript. The Django framework was used to create and manage the application's operations and navigation throughout the site (Backend of the website), SQLite as a database server (store and retrieve records), JavaScript for responsiveness of the web pages and different attributes, html for creating web pages, CSS for styling and designing web pages. Visual Studio Code served as my primary code editor, while Git facilitated version control.

2. Website Cloning Process:

Django Setup: I began by creating a Django project and apps for different sections of the website, such as login and Form.

Database Models: I designed Django models to mimic the structure of the original website's

data, including doctors, services, and items. Django models have been used to create various tables in the database which are used to store information from the web pages. Below is an example of a model that stores items, services, and doctors' details in a database.

```
class Service(models.Model):
    title = models.CharField(max_length=120)
    description = models.TextField()
    items = models.ManyToManyField(to='Item',)
    thumbnail = models.ImageField(upload_to='services/')
    cover = models.ImageField(upload_to='services/')
    image1 = models.ImageField(upload_to='services/', blank=True, null=True)
    image2 = models.ImageField(upload_to='services/', blank=True, null=True)

    def __str__(self):
        return self.title

class Item(models.Model):
    title = models.CharField(max_length=120)

    def __str__(self):
        return self.title

class Doctor(models.Model):
    name = models.CharField(max_length=120)
    speciality = models.CharField(max_length=120)
    picture = models.ImageField(upload_to="doctors/")
    details = models.TextField()
    experience = models.TextField()
    expertize = models.ManyToManyField(to='Expertize', related_name='doctors')
    twitter = models.CharField(max_length=120, blank=True, null=True)
    facebook = models.CharField(max_length=120, blank=True, null=True)
    instagram = models.CharField(max_length=120, blank=True, null=True)
```

Figure1: Models' snippet

HTML Templates: Using Django's template engine, I created reusable HTML templates that dynamically generate content from the database.

3. Functionality:

I successfully cloned and implemented the following functionalities from the original Corrigan MineHan Heart Center website:

Admin Authentication and Registration

Utilizing Django's built-in authentication system, I created a login page where users can securely log in with their credentials that is **username(Admin)** and **password(confie123@gmail.com)**. Additionally, I designed an appointment form that allows all patients to make appointments. This ensures easy accessibility of doctors. Below are the forms for admin's login and Appointment respectively.

LOGIN_PAGE

On the admin's login page, the admin types in his or her email address (used as username) and password. Once the credentials submitted don't match, an error message is displayed on top of the page indicating that please enter the correct username and password for a staff account. Note that both fields may be case-sensitive.

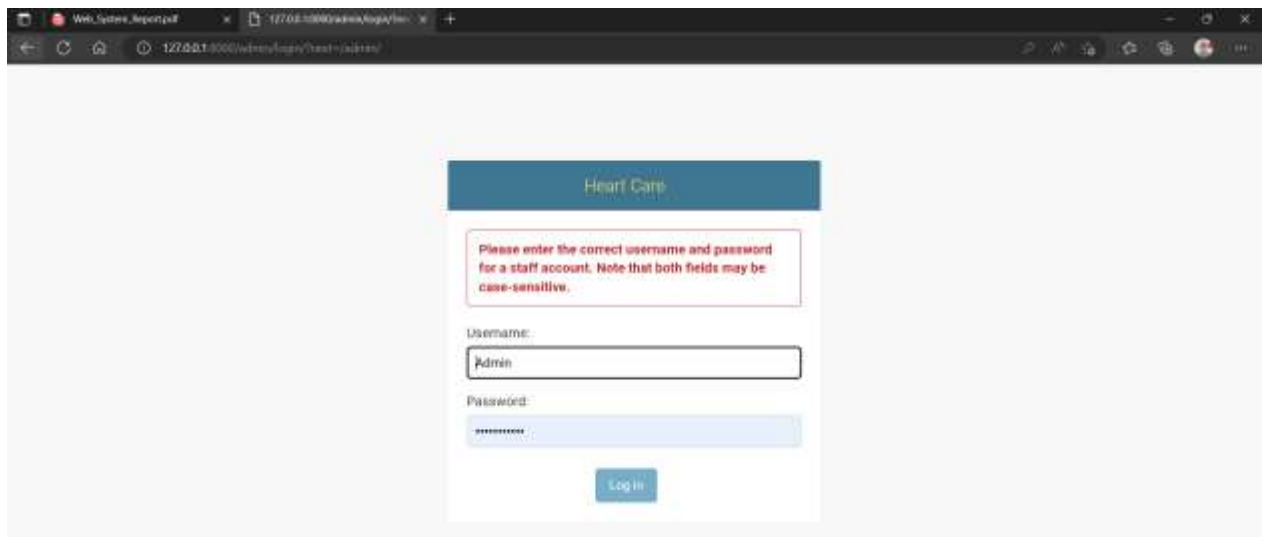


Figure2: Login form and Login error Message

If the username (email) and password match with the record in the database, then the admin is taken to his page where he can do his respective work from. From the admin page, login page was designed to enable the admin log into his corresponding dashboard. The administrator from his dashboard can add doctor, add services, add gallery pictures. An admin can also view the above information where by he can also delete, update and view the

recent actions that have been taking place in the system. The administrator can also register a new administrator. The administrator can also view the site and change the password as well. Below is the dashboard for the admin

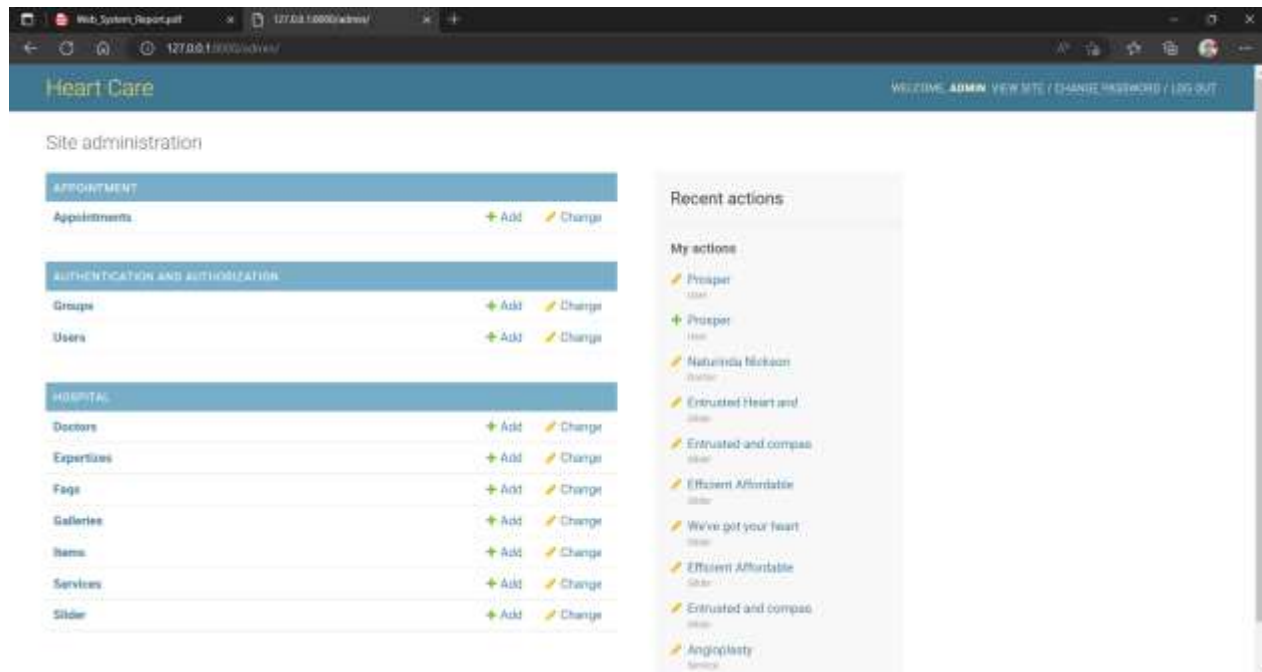


Figure3: Admin's Dashboard


APPOINTMENT'S DASHBOARD

From the appointment dashboard, the users can make appointments with their favorable doctors by filling the form for the appointment which includes full name, phone, doctor, your email, date, daytime and extra note.

Web System Report.pdf Heart Care Hospital 127.0.0.1:3000/appointment/

Get Appointment

Home // Appointment



Full Name *

Your Phone *

Your Email *

mm/dd/yyyy

Dr. Ankurda Trida

Morning

Extra Note

Book Appointment

Figure4:Appointment's dashboard

This web system is made up of a different categories of web pages which include home page, Services page, administration page, doctor's page, appointment page, FAQs page, Gallery page, and contact page. The home page contains different hyperlinks in the navigation bar which when clicked directs the user to the corresponding log in page. The home page is shown in the figure below.



Figure5: Home Page

VIEWS

I used the Django views to render and redirect to different web pages on request by the user. More so, they have been used with the help of “**POST**” method to transfer data from a webpage (in put fields) to the models for storage and retrieving using query sets. Below is a sample of a view used to post data on the patient’s dashboard.

```

1  from django.shortcuts import render, redirect
2  from django.core.mail import send_mail
3  from django.contrib import messages
4  from .models import Slider, Service, Doctor, Faq, Gallery
5  from django.views.generic import ListView, DetailView, TemplateView
6
7
8  class HomeView(ListView):
9      template_name = 'hospital/index.html'
10     queryset = Service.objects.all()
11     context_object_name = 'services'
12
13     def get_context_data(self, **kwargs):
14         context = super().get_context_data()
15         context['sliders'] = Slider.objects.all()
16         context['experts'] = Doctor.objects.all()
17         return context
18
19
20 class ServiceListView(ListView):
21     queryset = Service.objects.all()
22     template_name = "hospital/services.html"
23
24
25 class ServiceDetailView(DetailView):
26     queryset = Service.objects.all()
27     template_name = "hospital/service_details.html"
28
29     def get_context_data(self, **kwargs):
30         context = super().get_context_data(**kwargs)
31         context["services"] = Service.objects.all()
32         return context
33

```

Figure6: Views' Snippet

4. Design and Layout:

I aimed to retain the original design's aesthetics, focusing on responsive layout, consistent typography, and the same color palette. The responsive design adapts to different screen sizes.

5. Code Structure and Organization:

Django's project and app structure naturally organizes code. I maintained separation between HTML templates, CSS, and JavaScript files within each app.

6. Testing:

I extensively tested the cloned website

(Heart care website) on various devices and browsers to identify and address any layout issues, broken links, or JavaScript errors. Cross-browser compatibility was achieved through targeted testing and adjustments.

7. Conclusion:

Cloning the Corrigan MineHan Heart website with Django has enriched my web development skills, deepening my understanding of HTML, CSS, JavaScript, and Django. This project underscored the value of attention to detail and responsive design in building user-friendly website.

8. Future Improvements:

In the future, I plan to refine the animations and enhance the search functionality to provide more intuitive results.

9. References:

www.massgeneral.org/

<https://docs.djangoproject.com/>

<https://www.w3schools.com/django/>