FACULTY OF COMPUTING AND INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

WEB APPLICATION DEVELOPMENT REPORT

NAME: TWIJUKYE JAMES

REG NO:2022/BCS/021/PS

LECTURER: TONY ENGWAU AND YONASI SAFARI

DATE:25/MAY/2024

Table of Contents

Chapter1: Introduction

1.1 Background Informations

Tourism plays a vital role in Uganda's economy, contributing significantly to its Gross Domestic Product (GDP) and providing employment opportunities for millions of people. According to the Uganda Ministry of Tourism, Wildlife and Antiquities, Tourism Direct Gross Domestic Product accounts for 3.64 percent of the GDP in terms of the direct contribution. Additionally, tourism directly employs close to 1.6 million people, with 68 percent of them being females, and accounts for 14.7 percent of the total number of jobs in the country. With such substantial contributions, the effective management of tourism systems becomes paramount for sustaining this economic g1rowth.

1.2 Problem statement

This project aims to develop a comprehensive Tourism Management System (TMS) for Uganda using Bootstrap and Django to streamline operations and enhance efficiency. By addressing challenges like fragmented management, inadequate technology integration, and lack of cohesive strategies, the TMS will support sustainable growth and development. Leveraging modern web technologies, this system will foster socio-cultural development and drive economic growth in Uganda's tourism sector.

1.3 Motivation

My research is motivated by the urgent need to address the deficiencies in the management of tourism systems in Uganda. By conducting this project, I aim to develop a comprehensive Tourism Management System (TMS) that integrates modern web technologies to overcome existing challenges and enhance the efficiency and competitiveness of Uganda's tourism industry. This project seeks to contribute to the sustainable growth and development of the tourism sector in Uganda.

1.4 Objectives

The objectives of my project are as follows:

1.41 General Objective:

I aim to develop a comprehensive Tourism Management System (TMS) for Uganda, integrating modern web technologies using Bootstrap and Django.

1.42 Specific Objectives:

- To conduct a detailed analysis of the current state of tourism management in Uganda, identifying key challenges and areas for improvement.
- To design and develop a TMS prototype that addresses the identified challenges and incorporates features for efficient management of tourism activities.
- To integrate modern web technologies, including Bootstrap for front-end development and Django for back-end functionality, into the TMS prototype.
- To test and evaluate the usability, functionality, and performance of the developed TMS prototype. To provide recommendations for the implementation and adoption of the TMS prototype by stakeholders in Uganda's tourism industry.

Chapter2: Literature Review

Introduction

Tourism management systems play a pivotal role in the efficient organization and operation of tourism-related activities, facilitating bookings, payments, and communication between tourists and service providers. In the context of Uganda, where tourism is a significant contributor to the economy, the development of robust tourism management systems is imperative for sustainable growth and enhanced visitor experiences. Modern web technologies such as Bootstrap and Django offer valuable tools to streamline the development of these systems, enabling responsive interfaces and scalable backend architectures.

Overview of Tourism Management Systems

Tourism management systems encompass a range of software applications designed to facilitate various aspects of the tourism industry, including booking accommodations, transportation, tours, and activities. These systems aim to optimize resource allocation, enhance customer satisfaction, and maximize revenue generation. Over time, tourism management systems have evolved from basic reservation systems to sophisticated platforms incorporating features like dynamic pricing, personalized recommendations, and integrated communication channels.

Web Technologies in Tourism Management

Bootstrap:

Bootstrap, a front-end framework, plays a crucial role in crafting intuitive and visually appealing user interfaces for tourism management systems. Its grid system and pre-styled components simplify the development process, ensuring consistency across different devices and browsers.

Django:

As a backend framework, Django offers numerous benefits for building robust and secure tourism management systems. Its scalability, security features, and adherence to the Model-Template-View (MTV) architecture make it an ideal choice for handling complex business logic and managing data.

Integration of Bootstrap and Django

The integration of Bootstrap and Django brings synergistic advantages to tourism management system development. combining these technologies streamlines development processes and improves system performance. By leveraging Bootstrap's front-end capabilities and Django's backend functionalities, developers can create cohesive and efficient web applications tailored to the needs of the tourism industry.

Current Trends and Future Directions

Current trends in tourism management systems include the integration of AI, machine learning, and data analytics to personalize recommendations, optimize resource allocation, and predict traveler preferences. In the future, technologies like Bootstrap and Django are expected to evolve further to meet the growing demands of the tourism industry, offering more advanced features for seamless booking experiences and enhanced communication channels.

Conclusion

In conclusion, the literature review underscores the pivotal role of tourism management systems in enhancing the efficiency and competitiveness of the tourism industry in Uganda. The integration of modern web technologies such as Bootstrap and Django offers promising avenues for developing innovative solutions that cater to the evolving needs of travelers and service providers alike.

Chapter 3 Methodology

Design

For this tourism management system, I employed an Agile methodology to steer the research process. Agile methodologies are highly regarded for their iterative development, collaboration, and flexibility, which are particularly beneficial for navigating the dynamic and evolving landscape of research, especially in the context of Uganda's tourism industry. The Agile approach was chosen to promptly adapt to emerging insights and stakeholder feedback, ensuring the relevance and effectiveness of the research outcomes. This approach enabled me to effectively explore the integration of modern web technologies such as Bootstrap and Django within Uganda's tourism management systems, fostering innovation and efficiency in the research process.

Sampling

The population for this sampling included academic articles, industry reports, case studies, and government publications related to tourism management systems in Uganda. A purposive sampling technique was employed to select relevant literature based on criteria such as relevance to the research topic, publication date (within the last 10 years), and methodological rigor. The sample size was determined based on the saturation of themes and concepts in the literature, aiming for comprehensive coverage of relevant sources.

Data Collection Methods

I collected data through systematic searches of electronic databases such as PubMed, Google Scholar, and Scopus. Keywords including "tourism management systems," "Uganda tourism," "hospitality industry," "Bootstrap," and "Django" were used to retrieve relevant literature. Additionally, references cited in identified articles were examined to ensure a comprehensive search. Data were collected by the researcher and recorded systematically using reference management software.

Data Analysis

I employed thematic analysis to analyze the collected data. Themes and patterns related to tourism management systems in Uganda, the role of Bootstrap and Django, current trends, and future directions were identified and synthesized. Data analysis involved coding, categorizing, and interpreting information from the selected literature. The analysis was conducted iteratively, with regular reflection and discussion to ensure rigor and validity.

Ethical Considerations

I maintained ethical considerations by ensuring proper citation and acknowledgment of sources. All data I used in the literature review were obtained from publicly available sources, and no direct interaction with human subjects was involved. Proper ethical guidelines for academic research were followed, including maintaining confidentiality and integrity in data presentation.

Validity and Reliability

To ensure the validity and reliability of the findings, I carried out the data collection and analysis process. I held Consensus meetings to resolve any discrepancies or disagreements in data interpretation. Additionally, the use of established databases and peer-reviewed sources enhanced the credibility of the findings.

Methodology Timeline

Week 1-2: Literature Search

- I conducted systematic searches of electronic databases using keywords related to tourism management systems, Bootstrap, Django, and Uganda tourism.
- I gathered relevant academic articles, industry reports, government publications, and case studies collected literature was organized using reference management software.

Week 3-4: Data Collection

- I gathered additional literature through manual searches and exploration of references cited in identified articles.
- I ensured comprehensive coverage of relevant sources based on predetermined inclusion criteria.
- I recorded all collected systematically, including bibliographic information and key findings.

Week 5-6: Initial Data Analysis

- I initiated thematic analysis of collected literature, focusing on identifying recurring themes and patterns.
- I coded Information and categorized it according to predefined themes related to tourism management systems, Bootstrap, Django, current trends, and future directions.
- I held Preliminary discussions with the research team to refine coding schema and ensure consistency in analysis.

Week 7-8: Continued Data Analysis

I refined Thematic analysis through iterative coding and categorization.

I discussed emerging themes and insights with the research team to validate interpretations and identify areas for deeper exploration.

I conducted additional searches or data extraction as needed to address any gaps or inconsistencies in the literature.

Week 9-10: Synthesis of Findings

- I synthesized Key findings from thematic into coherent narratives.
- I identified Connections and relationships between different themes and concepts.
- I drafted initial sections of the literature review report, including introduction and methodology,

Week 11-12: Report Writing and Finalization

- I wrote remaining sections of the literature review report, including discussion, conclusion, and recommendations
- I ensured clarity, coherence, and logical flow of information throughout the report were Thorough proofreading and editing were conducted to address any errors or inconsistencies.
- I finalized the report and prepared for submission or presentation.

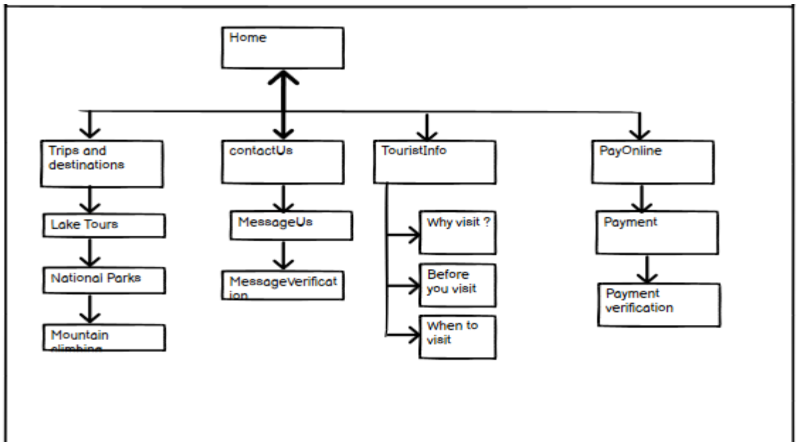
Budget

The project efficiently utilized its budget allocation of 1.675 million from planning to design stages. Existing institutional subscriptions to databases and reference management software minimized additional financial expenses. Maximizing access to resources like PubMed, Google Scholar, and Scopus ensured optimal fund utilization. Primary resource investment in my time and effort aligned with predetermined budget constraints, reflecting prudent financial management for high-quality outcomes.

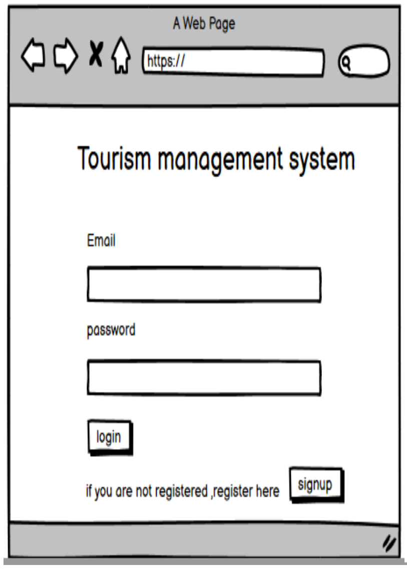
Activity No	Activity	Cost(UGX)
1	**Planning**	150,000
1.0	Project initiation	50000
2.0	Resource allocation	40000
3.0	Project Scope	30000
4.0	Budget cost	30000
2	**Analysis**	100000
2.0	Data analysis	25000
2.1	System Requirements specs	30000
2.2	User surveys and interviews	45000
3	**Design**	250000
3.0	Database design	60000
3.1	System analysis	40000
3.2	Use case, DFD	150000
4	**coding**	1000,000
4.1	Front end	400000
4.2	backend	400000
4.3	Database programming	200000
5	**Testing**	175,000
5.1	System testing	55,000
5.2	User approval and feedback	30,000
5.3	Bug fixing, errorrs and issue	90,000
	Total estimated cost	1,675,000

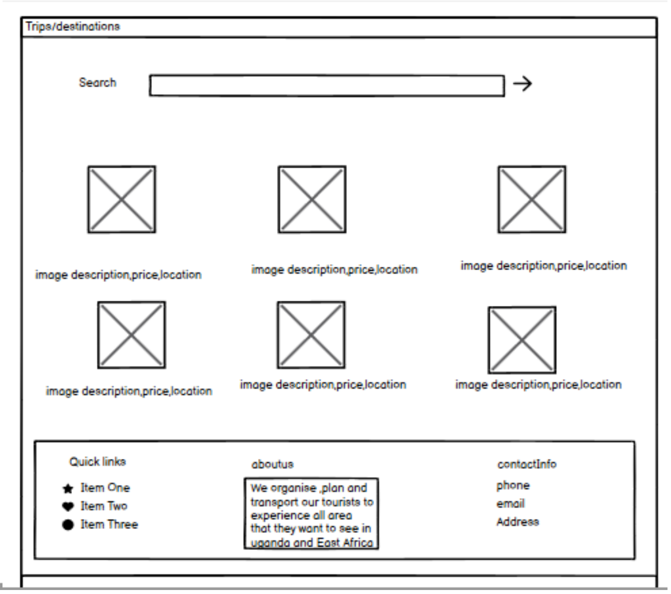
Chapter 4: Implementation

Site map



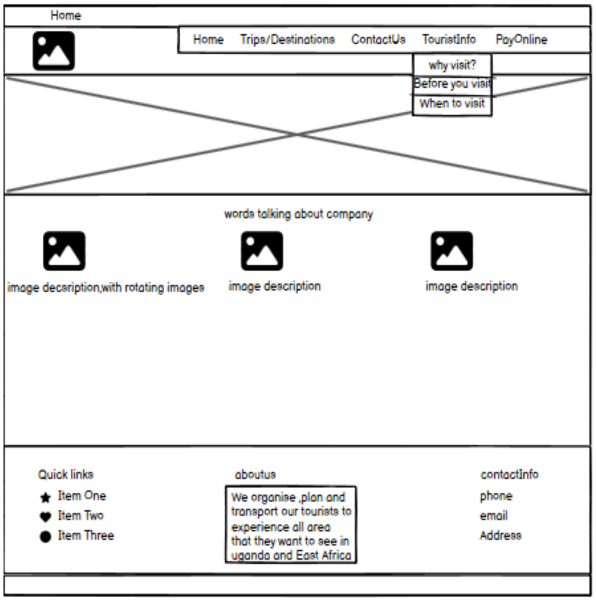
User interfaces drawn with balsamic wireframes


A Web Page
https://
Tourism management system
Email
password
login
if you are not registered ,register here
signup

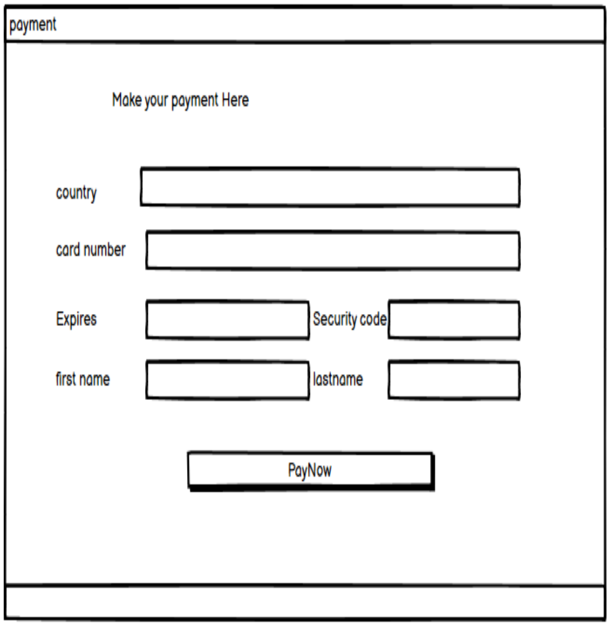

Trips/destinations
Search
image description,price,location
image description,price,location
image description,price,location
image description,price,location
image description,price,location
image description,price,location
Quick links
Item One
Item Two
Item Three
aboutus
contactInfo
phone
email
Address

We organise ,plan and transport our tourists to experience all area that they want to see in uganda and East Africa

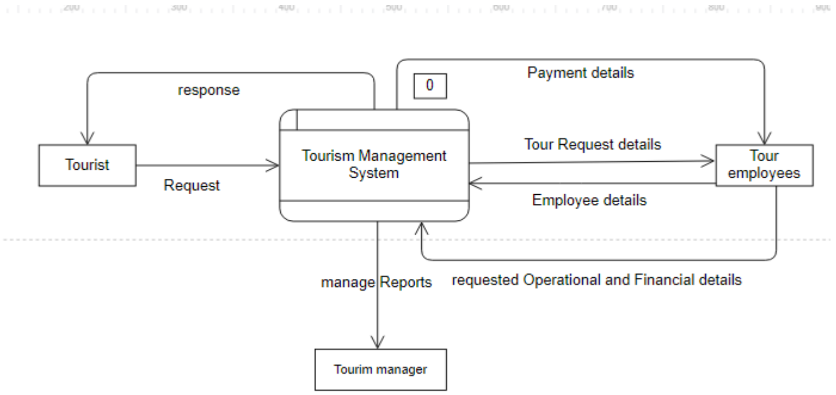
Twijukye James 2022/BCS/021/PS

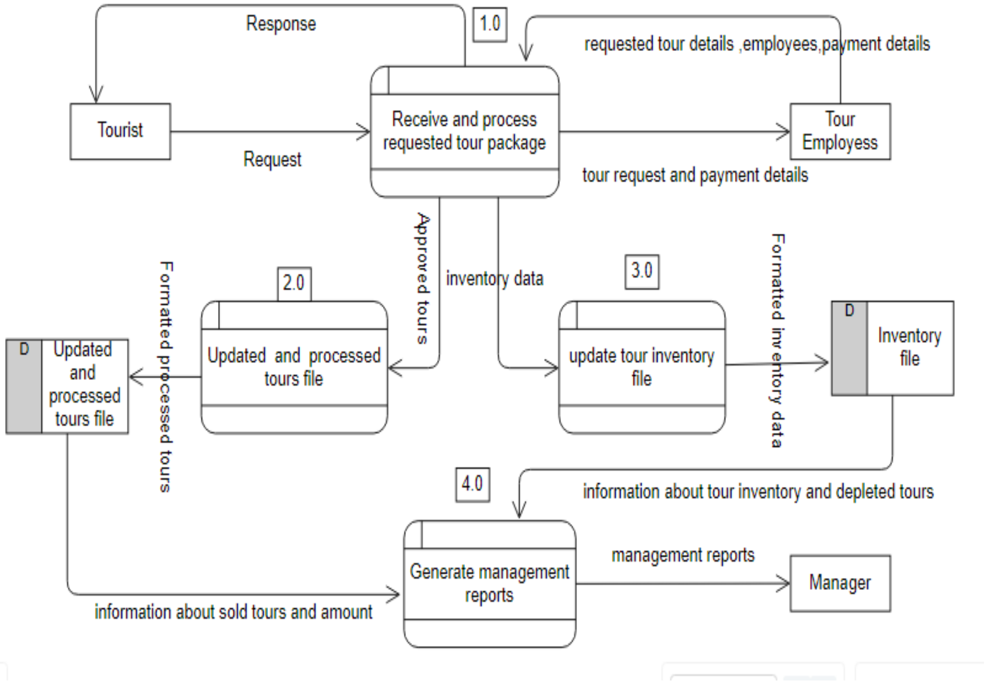


We organise ,plan and transport our tourists to experience all area that they want to see in uganda and East Africa

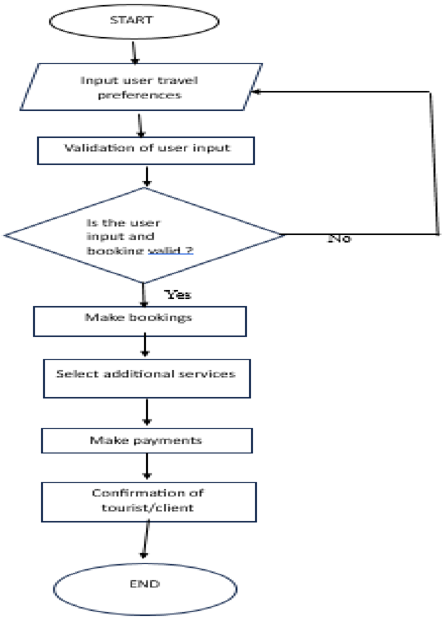


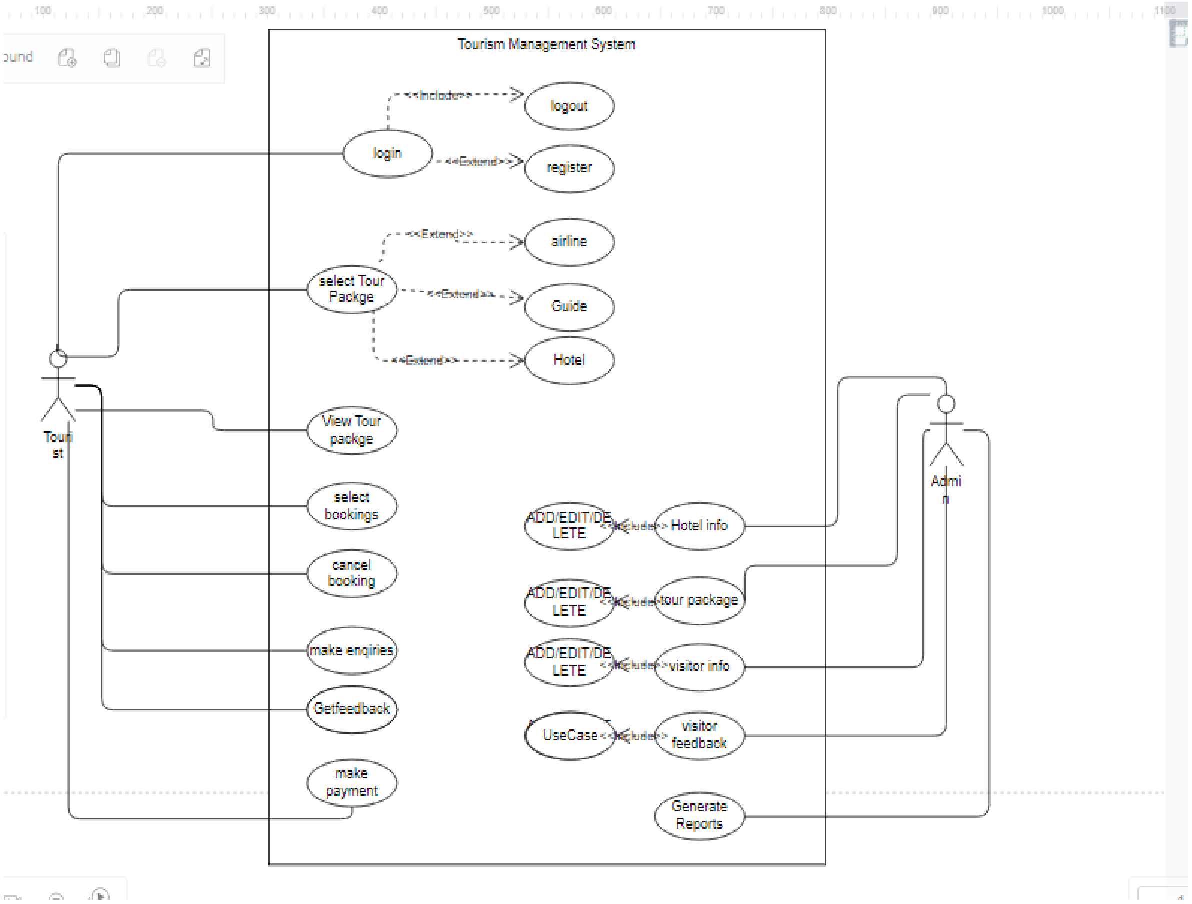
Data flow diagrams




1.0
Receive and process requested tour package
Tourist
Tour Employess
Response
Request
requested tour details ,employees,payment details
tour request and payment details
Approved tours
inventory data
2.0
Updated and processed tours file
3.0
update tour inventory file
D
Updated and processed tours file
Formatted processed tours
Formatted inventory data
D
Inventory file
4.0
Generate management reports
information about tour inventory and depleted tours
management reports
Manager
information about sold tours and amount

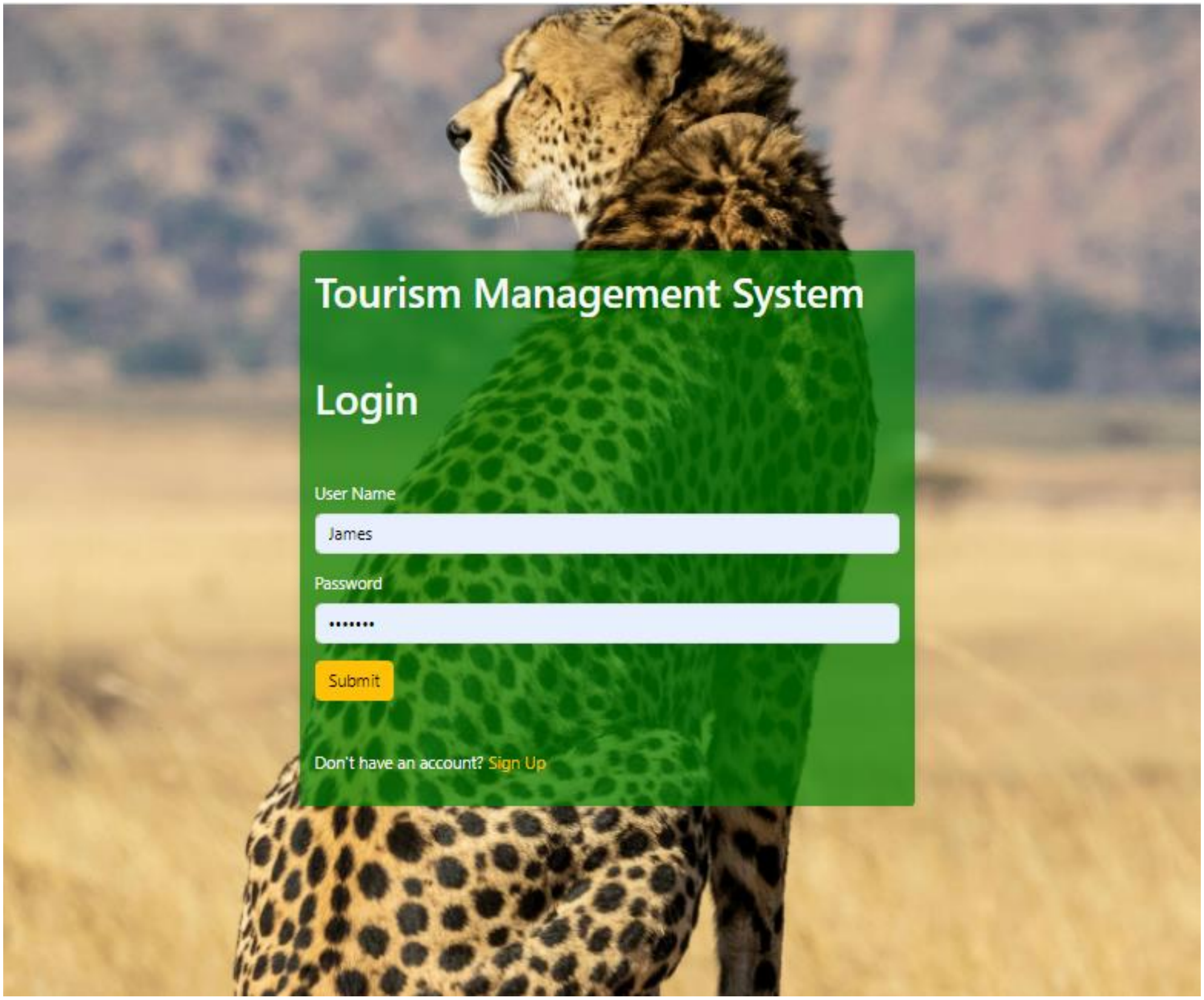
Flow chart and use case diagrams


START
Input user travel preferences
Validation of user input
Is the user input and booking valid ?
No
Yes
Make bookings
Select additional services
Make payments
Confirmation of tourist/client
END



Login page

The login.html template provides a user-friendly interface for logging in. The corresponding views handle user authentication, directing successful logins to the homepage while displaying error messages for invalid credentials. The models in the login app interface with Django's authentication system



Tourism Management System

Login

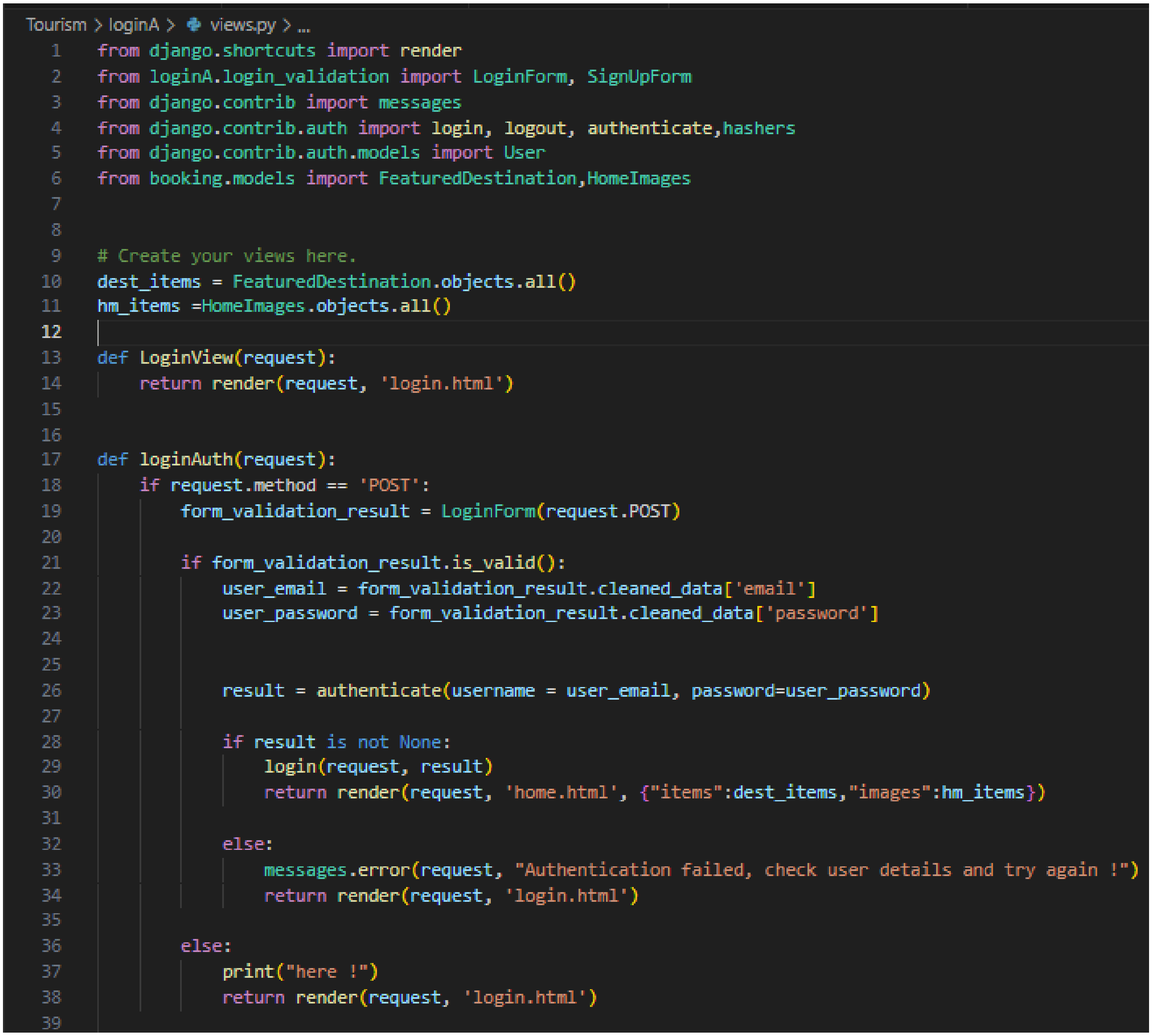
User Name

James

Password

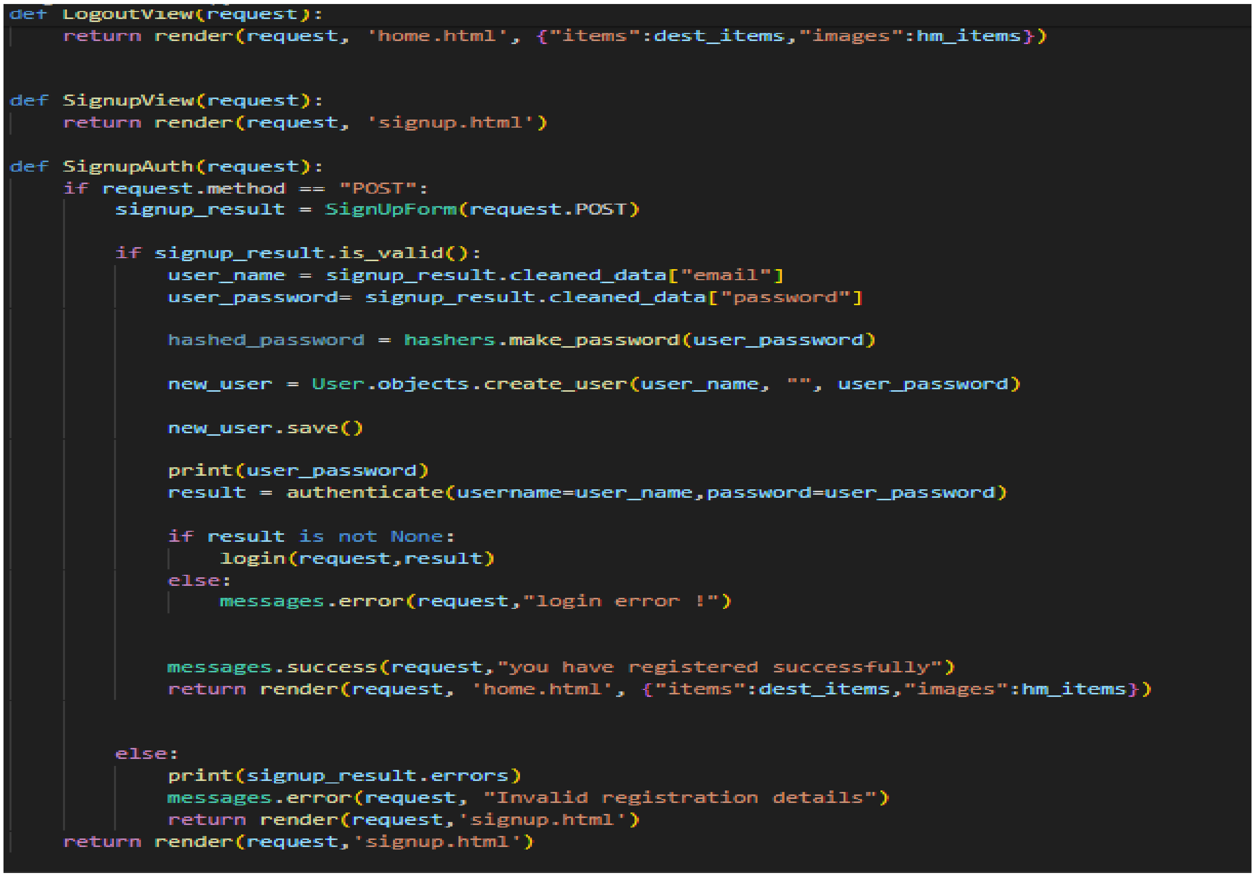
•••••••

Don't have an account? Sign Up

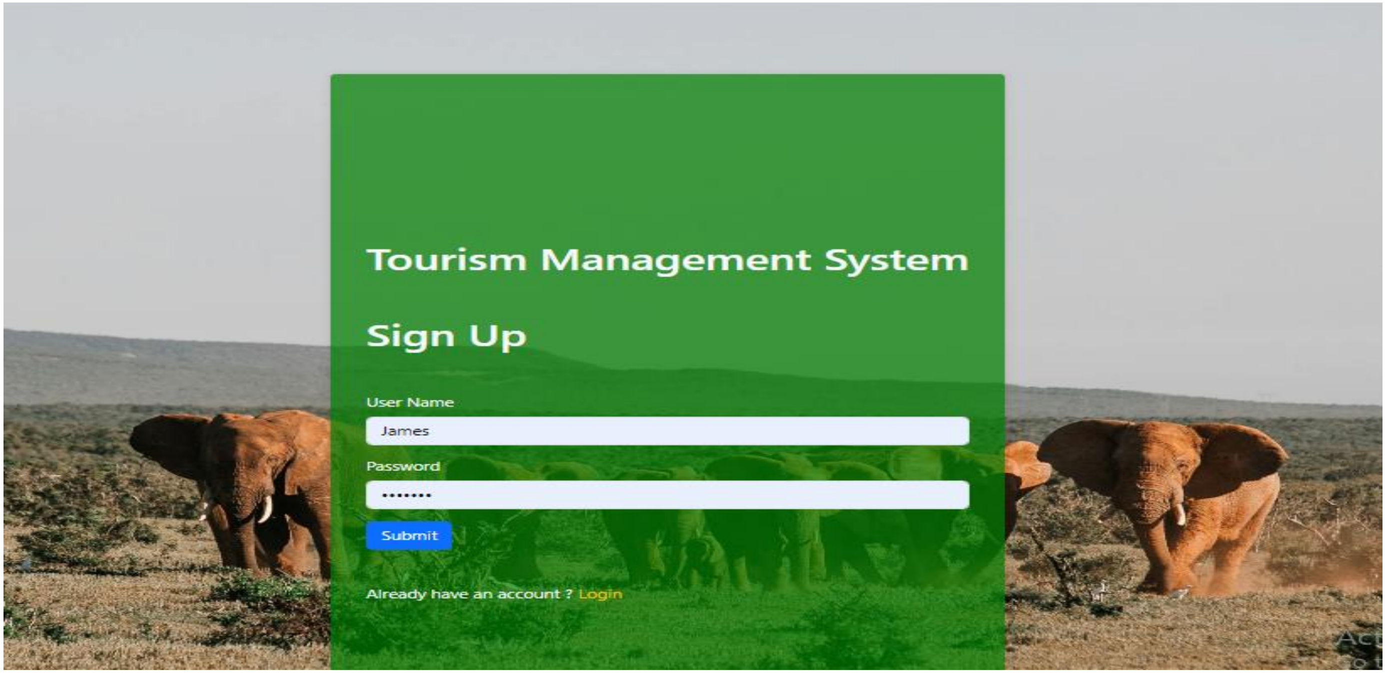


```python
from django.shortcuts import render
from loginA.login_validation import LoginForm, SignUpForm
from django.contrib import messages
from django.contrib.auth import login, logout, authenticate,hashers
from django.contrib.auth.models import User
from booking.models import FeaturedDestination,HomeImages


# Create your views here.
dest_items = FeaturedDestination.objects.all()
hm_items =HomeImages.objects.all()

def LoginView(request):
    return render(request, 'login.html')


def loginAuth(request):
    if request.method == 'POST':
        form_validation_result = LoginForm(request.POST)

        if form_validation_result.is_valid():
            user_email = form_validation_result.cleaned_data['email']
            user_password = form_validation_result.cleaned_data['password']


            result = authenticate(username = user_email, password=user_password)

            if result is not None:
                login(request, result)
                return render(request, 'home.html', {"items":dest_items,"images":hm_items})

            else:
                messages.error(request, "Authentication failed, check user details and try again !")
                return render(request, 'login.html')

        else:
            print("here !")
            return render(request, 'login.html')

```

Sign up page and views



```
def LogoutView(request):
    return render(request, 'home.html', {"items":dest_items,"images":hm_items})


def SignupView(request):
    return render(request, 'signup.html')

def SignupAuth(request):
    if request.method == "POST":
        signup_result = SignUpForm(request.POST)

        if signup_result.is_valid():
            user_name = signup_result.cleaned_data["email"]
            user_password= signup_result.cleaned_data["password"]

            hashed_password = hashers.make_password(user_password)

            new_user = User.objects.create_user(user_name, "", user_password)

            new_user.save()

            print(user_password)
            result = authenticate(username=user_name,password=user_password)

            if result is not None:
                login(request,result)
            else:
                messages.error(request,"login error !")


            messages.success(request,"you have registered successfully")
            return render(request, 'home.html', {"items":dest_items,"images":hm_items})


        else:
            print(signup_result.errors)
            messages.error(request, "Invalid registration details")
            return render(request,'signup.html')
    return render(request,'signup.html')
```



Tourism Management System

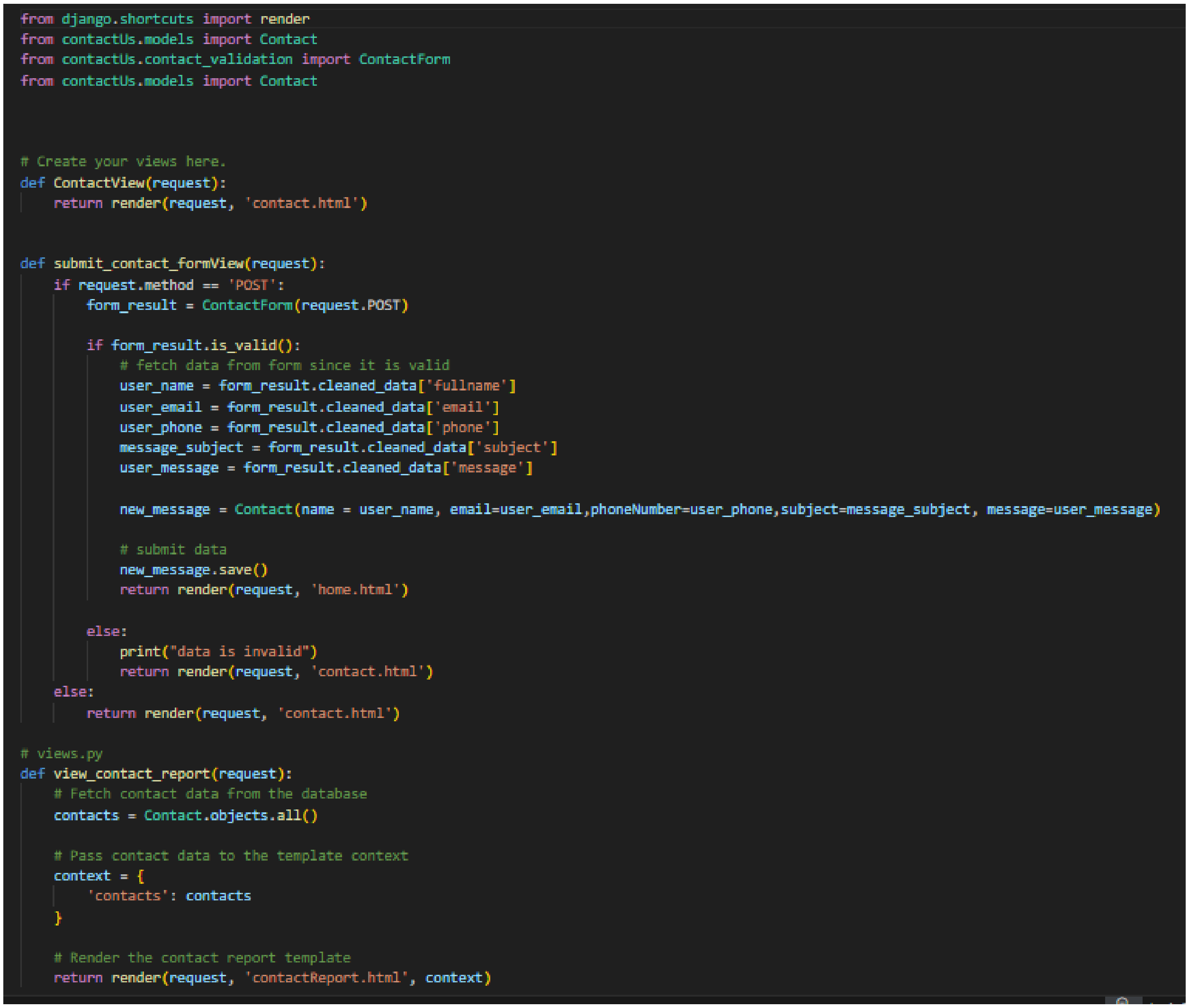
Sign Up

User Name: James

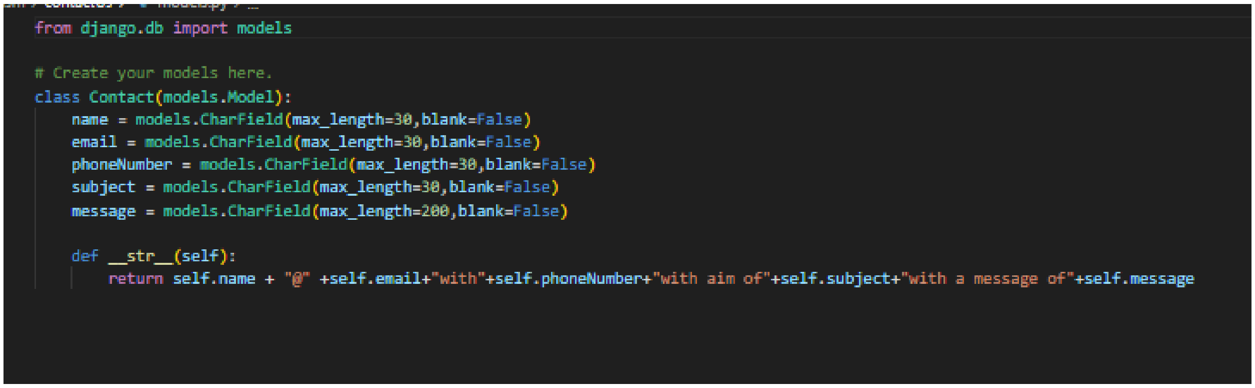
Password: •••••••

Already have an account ? Login

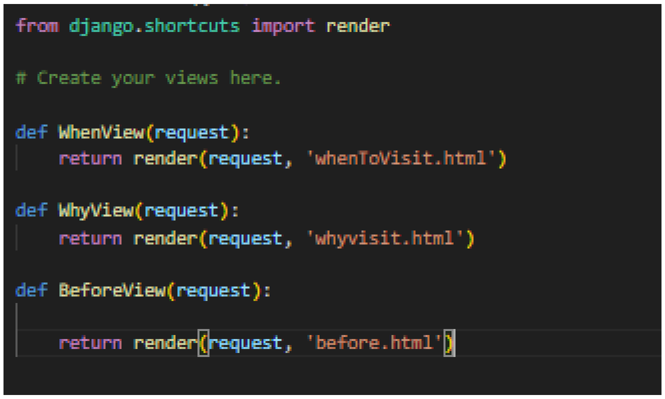
Contact-views, models code and contact us form



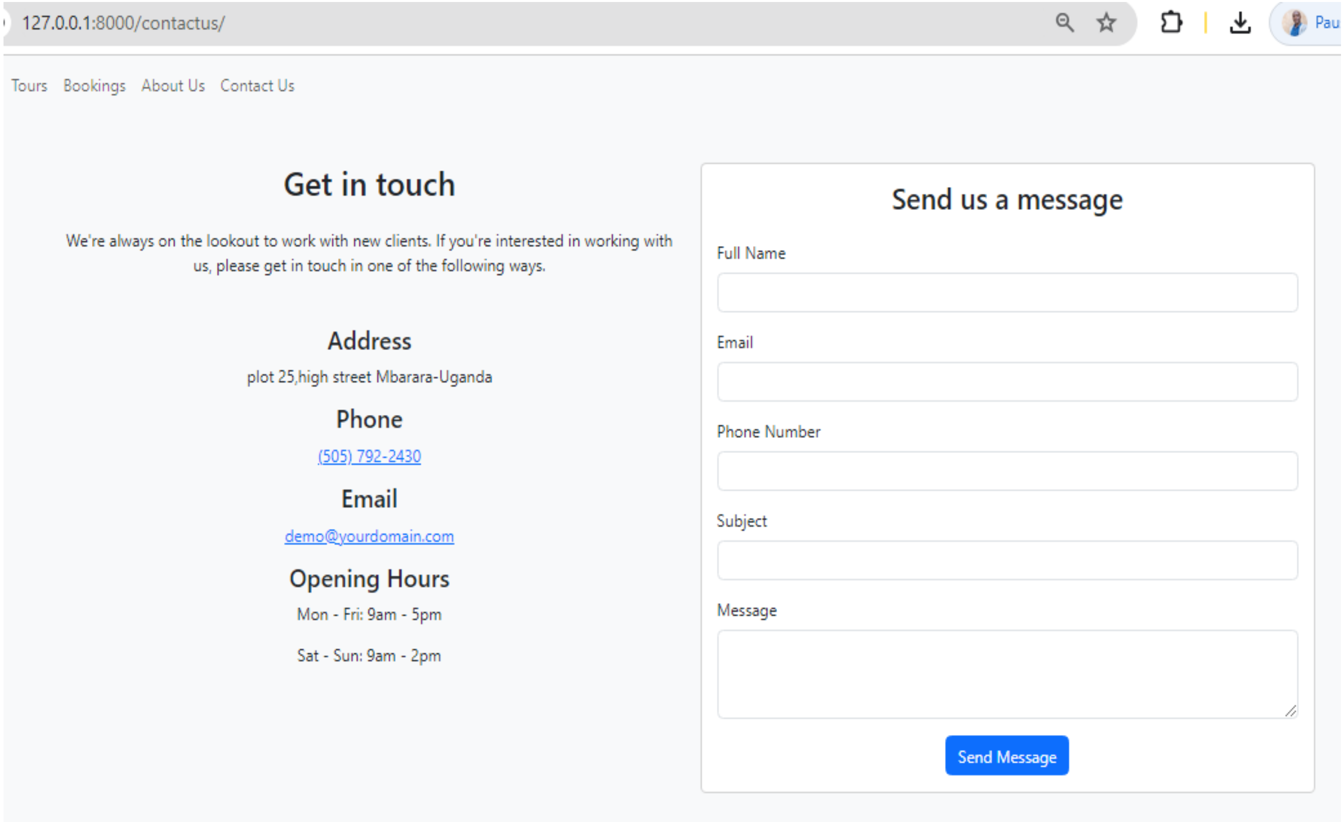
```python
from django.shortcuts import render
from contactUs.models import Contact
from contactUs.contact_validation import ContactForm
from contactUs.models import Contact



# Create your views here.
def ContactView(request):
    return render(request, 'contact.html')


def submit_contact_formView(request):
    if request.method == 'POST':
        form_result = ContactForm(request.POST)

        if form_result.is_valid():
            # fetch data from form since it is valid
            user_name = form_result.cleaned_data['fullname']
            user_email = form_result.cleaned_data['email']
            user_phone = form_result.cleaned_data['phone']
            message_subject = form_result.cleaned_data['subject']
            user_message = form_result.cleaned_data['message']

            new_message = Contact(name = user_name, email=user_email,phoneNumber=user_phone,subject=message_subject, message=user_message)

            # submit data
            new_message.save()
            return render(request, 'home.html')

        else:
            print("data is invalid")
            return render(request, 'contact.html')
    else:
        return render(request, 'contact.html')

# views.py
def view_contact_report(request):
    # Fetch contact data from the database
    contacts = Contact.objects.all()

    # Pass contact data to the template context
    context = {
        'contacts': contacts
    }

    # Render the contact report template
    return render(request, 'contactReport.html', context)
```



```
from django.db import models

# Create your models here.
class Contact(models.Model):
    name = models.CharField(max_length=30,blank=False)
    email = models.CharField(max_length=30,blank=False)
    phoneNumber = models.CharField(max_length=30,blank=False)
    subject = models.CharField(max_length=30,blank=False)
    message = models.CharField(max_length=200,blank=False)

    def __str__(self):
        return self.name + "@" +self.email+"with"+self.phoneNumber+"with aim of"+self.subject+"with a message of"+self.message
```



```
from django.shortcuts import render

# Create your views here.

def WhenView(request):
    return render(request, 'whenToVisit.html')

def WhyView(request):
    return render(request, 'whyvisit.html')

def BeforeView(request):

    return render(request, 'before.html')
```



Get in touch

We're always on the lookout to work with new clients. If you're interested in working with us, please get in touch in one of the following ways.

Address

plot 25,high street Mbarara-Uganda

Phone

(505) 792-2430

Email

demo@yourdomain.com

Opening Hours

Mon - Fri: 9am - 5pm

Sat - Sun: 9am - 2pm

Send us a message

Full Name

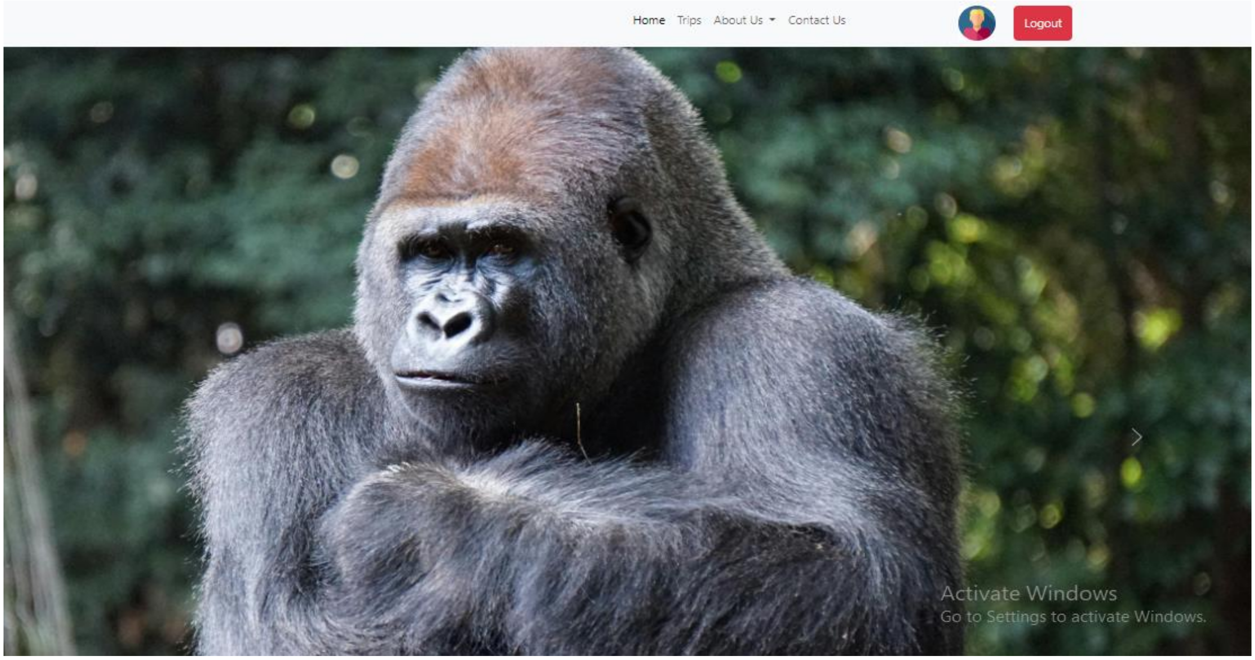
Email

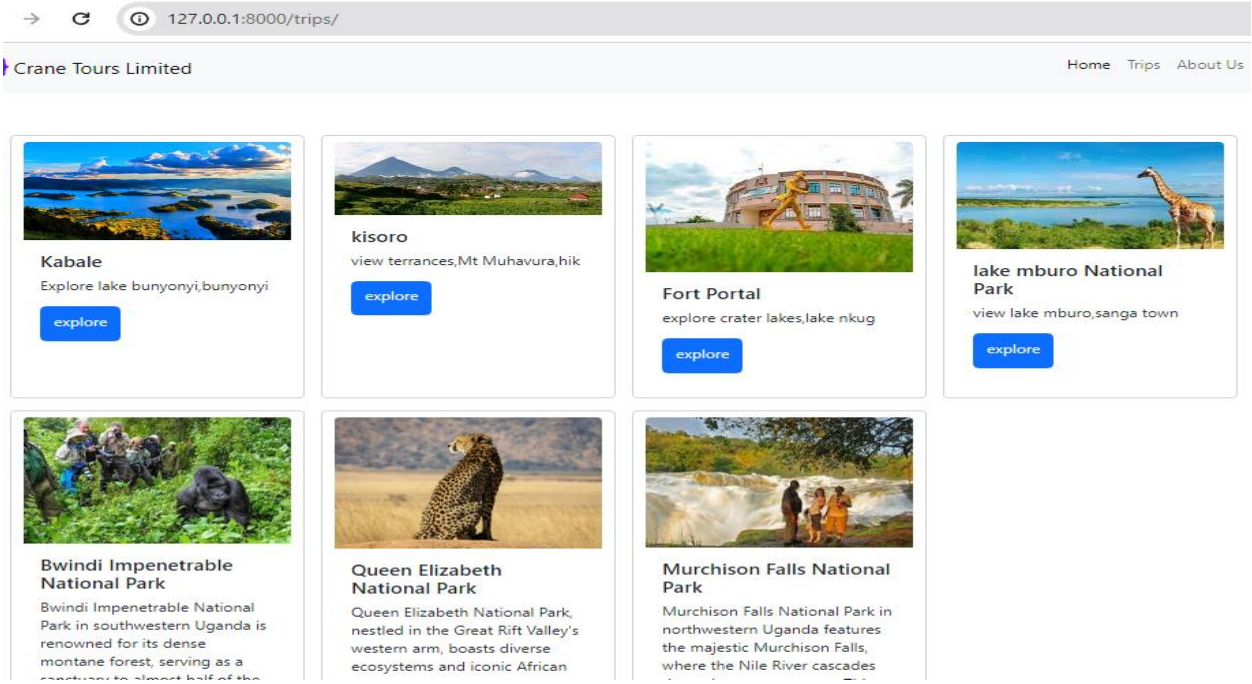
Phone Number

Subject

Message

Home page, views and models





Kabale

Explore lake bunyonyi,bunyonyi

kisoro

view terrances,Mt Muhavura,hik

Fort Portal

explore crater lakes,lake nkug

lake mburo National Park

view lake mburo,sanga town

Bwindi Impenetrable National Park

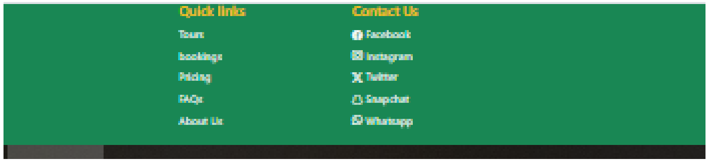
Bwindi Impenetrable National Park in southwestern Uganda is renowned for its dense montane forest, serving as a

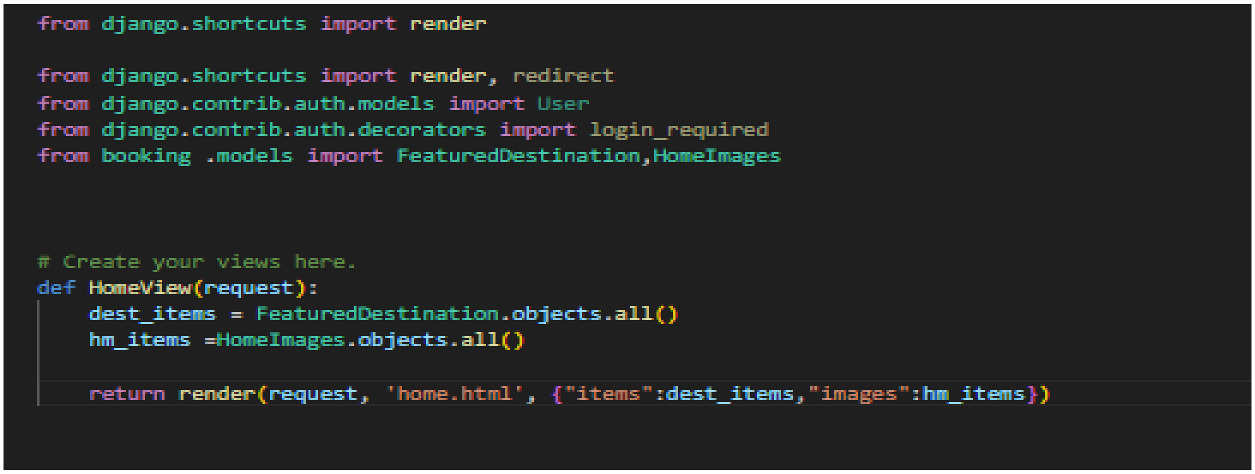
Queen Elizabeth National Park

Queen Elizabeth National Park, nestled in the Great Rift Valley's western arm, boasts diverse ecosystems and iconic African

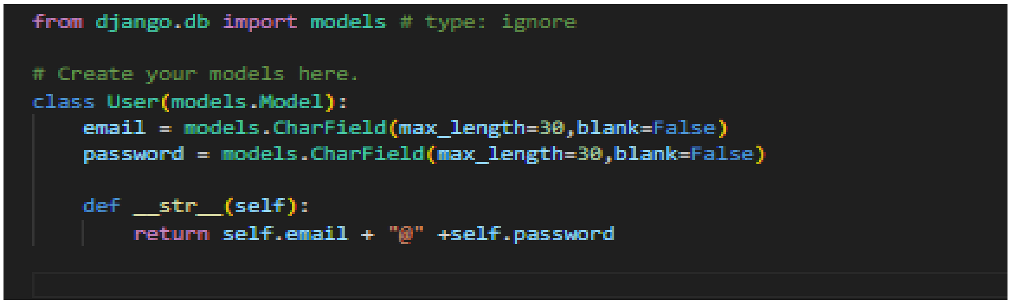
Murchison Falls National Park

Murchison Falls National Park in northwestern Uganda features the majestic Murchison Falls, where the Nile River cascades



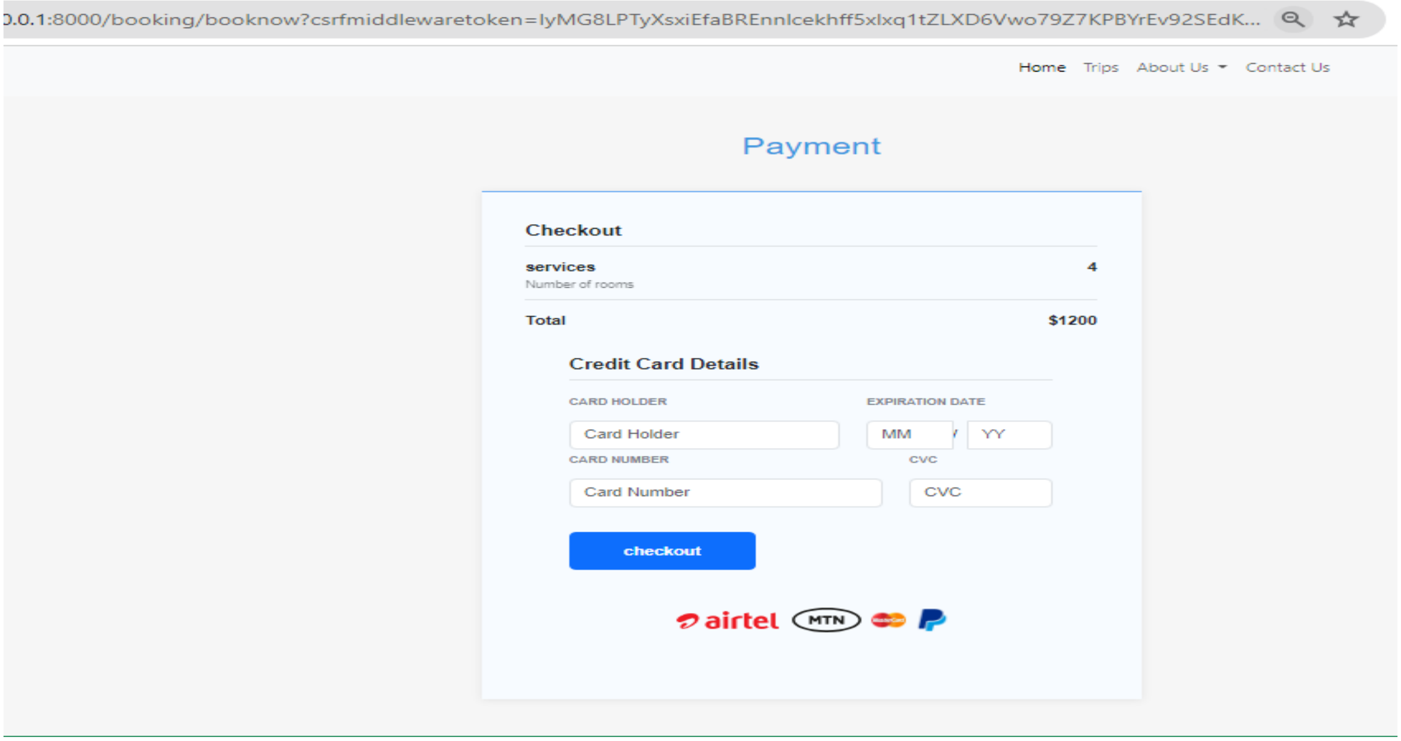


```
from django.shortcuts import render

from django.shortcuts import render, redirect
from django.contrib.auth.models import User
from django.contrib.auth.decorators import login_required
from booking .models import FeaturedDestination,HomeImages



# Create your views here.
def HomeView(request):
    dest_items = FeaturedDestination.objects.all()
    hm_items =HomeImages.objects.all()

    return render(request, 'home.html', {"items":dest_items,"images":hm_items})
```



```
from django.db import models # type: ignore

# Create your models here.
class User(models.Model):
    email = models.CharField(max_length=30,blank=False)
    password = models.CharField(max_length=30,blank=False)

    def __str__(self):
        return self.email + "@" +self.password
```

Pay form, model and views



.0.1:8000/booking/booknow?csrfmiddlewaretoken=IyMG8LPTyXsxiEfaBREnnlcekhff5xlxq1tZLXD6Vwo79Z7KPBYrEv92SEdK...

Payment

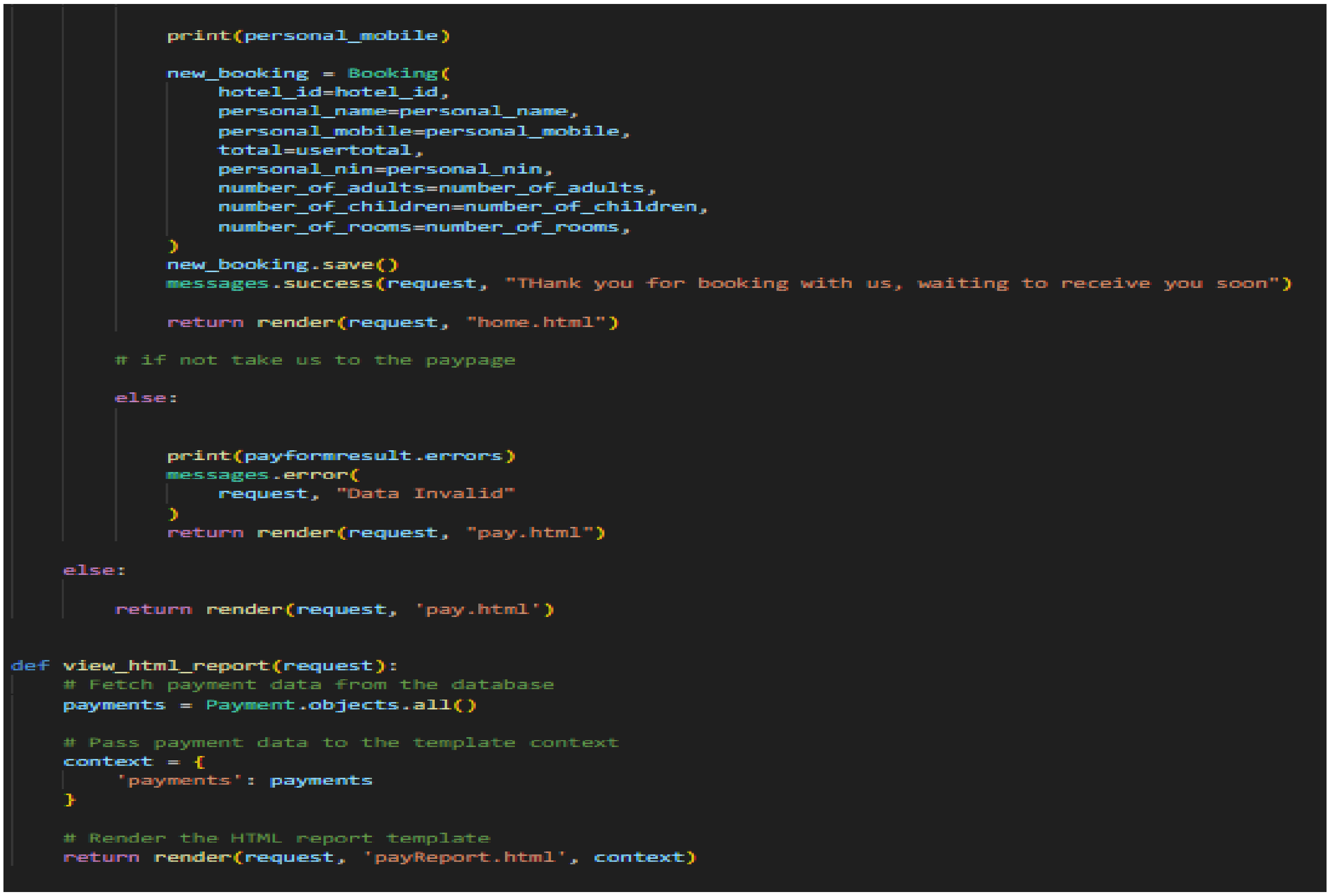
Checkout

services 4
Number of rooms

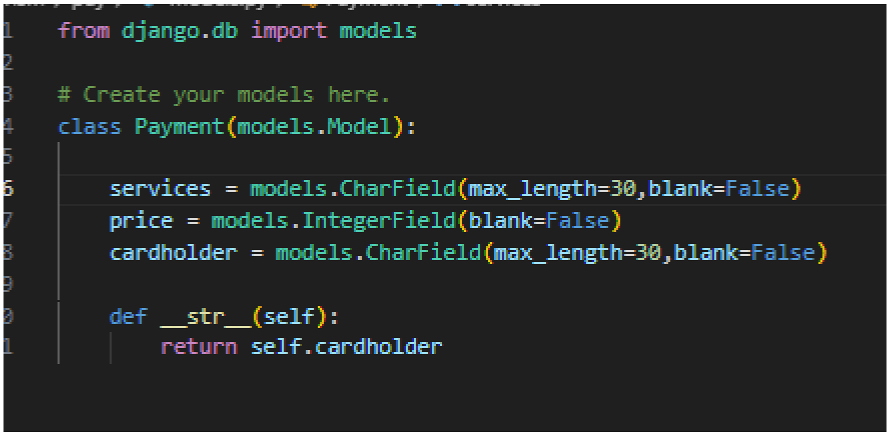
Total $1200

Credit Card Details

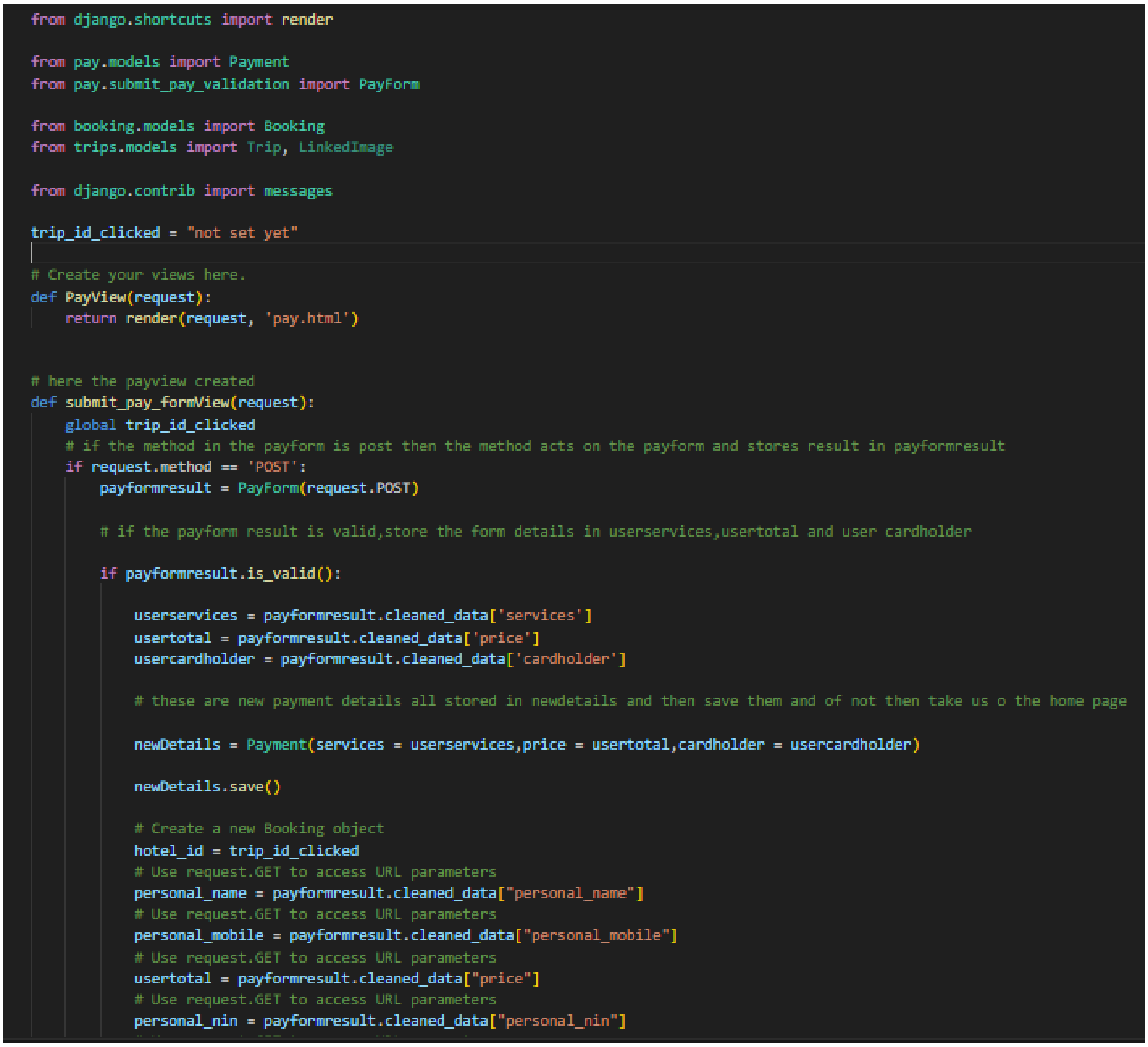
CARD HOLDER: Card Holder
EXPIRATION DATE: MM / YY
CARD NUMBER: Card Number
CVC: CVC



```
            print(personal_mobile)

            new_booking = Booking(
                hotel_id=hotel_id,
                personal_name=personal_name,
                personal_mobile=personal_mobile,
                total=usertotal,
                personal_nin=personal_nin,
                number_of_adults=number_of_adults,
                number_of_children=number_of_children,
                number_of_rooms=number_of_rooms,
            )
            new_booking.save()
            messages.success(request, "THank you for booking with us, waiting to receive you soon")

            return render(request, "home.html")

        # if not take us to the paypage

        else:

            print(payformresult.errors)
            messages.error(
                request, "Data Invalid"
            )
            return render(request, "pay.html")

    else:

        return render(request, 'pay.html')

def view_html_report(request):
    # Fetch payment data from the database
    payments = Payment.objects.all()

    # Pass payment data to the template context
    context = {
        'payments': payments
    }

    # Render the HTML report template
    return render(request, 'payReport.html', context)
```

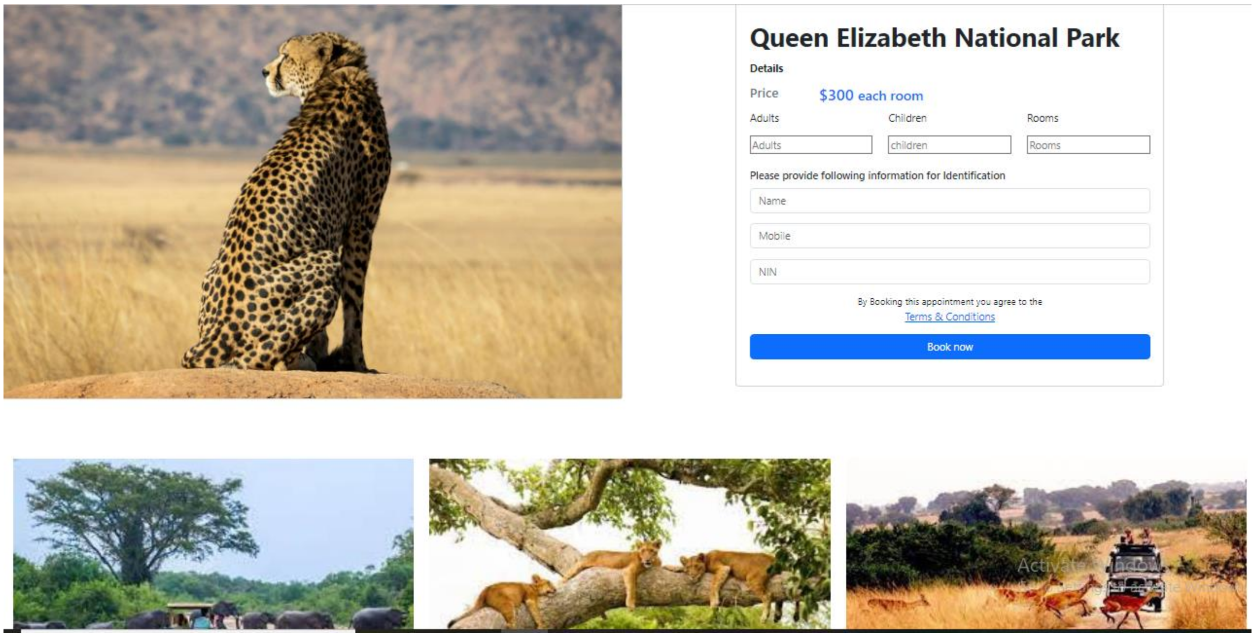


```python
from django.db import models

# Create your models here.
class Payment(models.Model):

    services = models.CharField(max_length=30,blank=False)
    price = models.IntegerField(blank=False)
    cardholder = models.CharField(max_length=30,blank=False)

    def __str__(self):
        return self.cardholder
```



```python
from django.shortcuts import render

from pay.models import Payment
from pay.submit_pay_validation import PayForm

from booking.models import Booking
from trips.models import Trip, LinkedImage

from django.contrib import messages

trip_id_clicked = "not set yet"

# Create your views here.
def PayView(request):
    return render(request, 'pay.html')


# here the payview created
def submit_pay_formView(request):
    global trip_id_clicked
    # if the method in the payform is post then the method acts on the payform and stores result in payformresult
    if request.method == 'POST':
        payformresult = PayForm(request.POST)

        # if the payform result is valid,store the form details in userservices,usertotal and user cardholder

        if payformresult.is_valid():

            userservices = payformresult.cleaned_data['services']
            usertotal = payformresult.cleaned_data['price']
            usercardholder = payformresult.cleaned_data['cardholder']

            # these are new payment details all stored in newdetails and then save them and of not then take us o the home page

            newDetails = Payment(services = userservices,price = usertotal,cardholder = usercardholder)

            newDetails.save()

            # Create a new Booking object
            hotel_id = trip_id_clicked
            # Use request.GET to access URL parameters
            personal_name = payformresult.cleaned_data["personal_name"]
            # Use request.GET to access URL parameters
            personal_mobile = payformresult.cleaned_data["personal_mobile"]
            # Use request.GET to access URL parameters
            usertotal = payformresult.cleaned_data["price"]
            # Use request.GET to access URL parameters
            personal_nin = payformresult.cleaned_data["personal_nin"]
```

Booking page, models and views



Queen Elizabeth National Park

Details

Price $300 each room

Adults	Children	Rooms
Adults	children	Rooms

Please provide following information for Identification

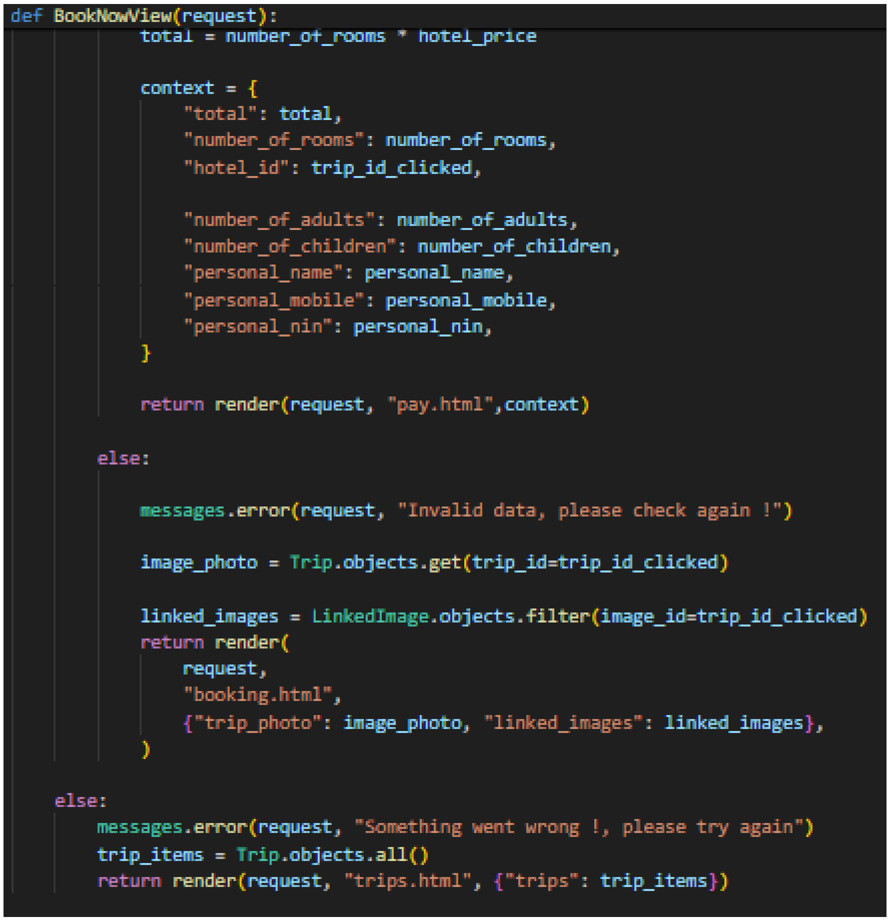
Name

Mobile

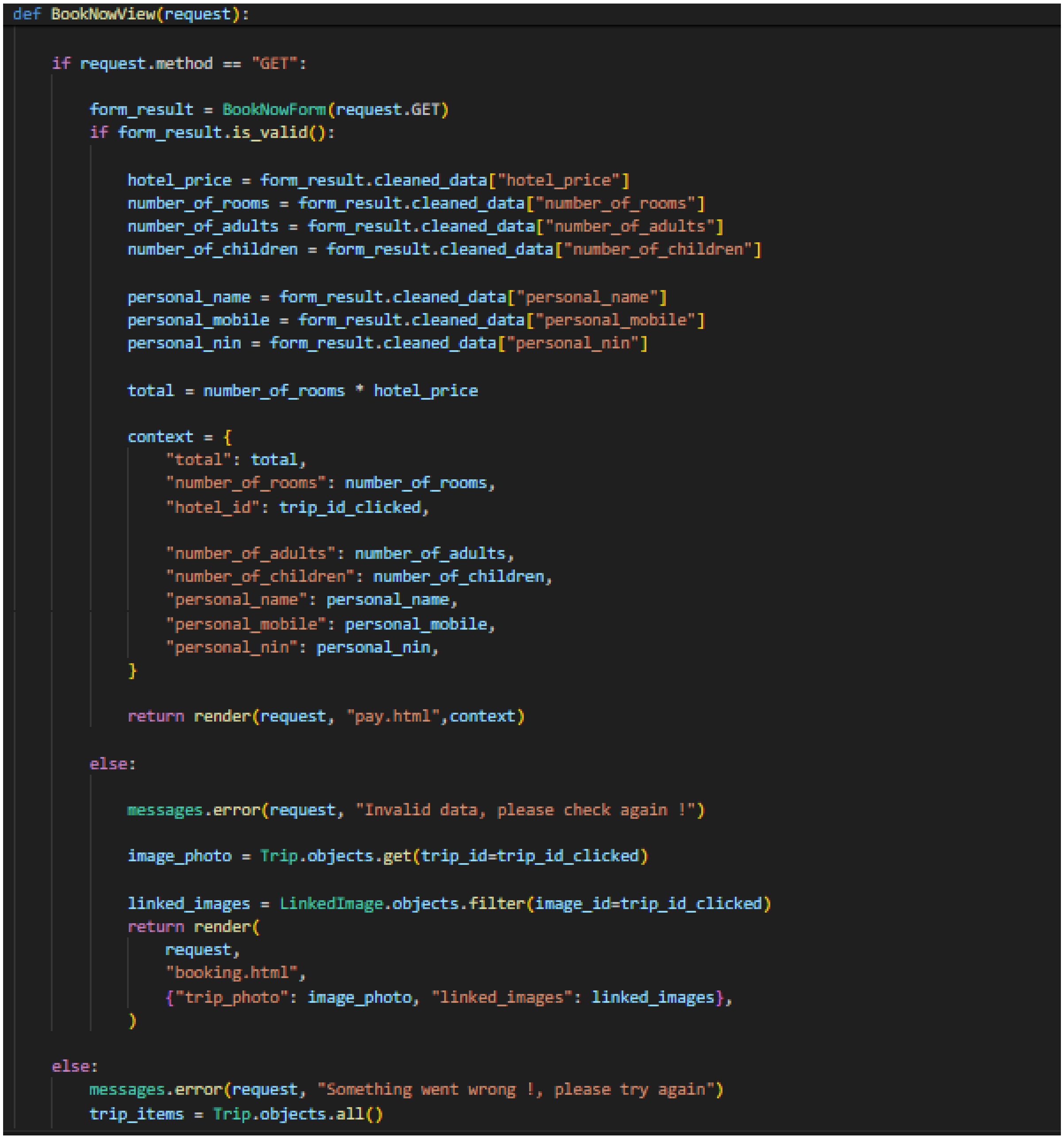
NIN

By Booking this appointment you agree to the
Terms & Conditions

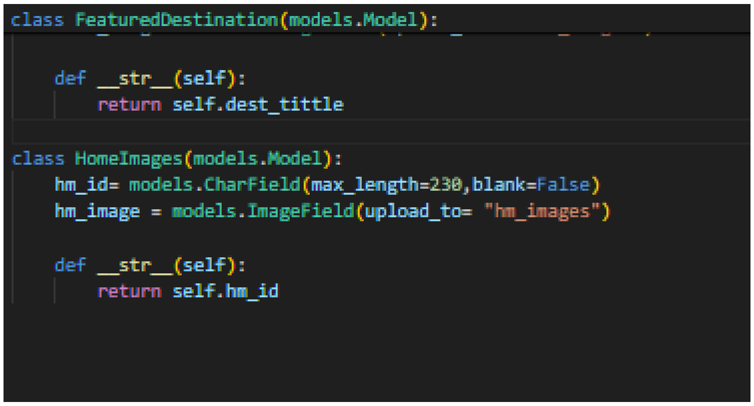
Book now



```
def BookNowView(request):
            total = number_of_rooms * hotel_price

            context = {
                "total": total,
                "number_of_rooms": number_of_rooms,
                "hotel_id": trip_id_clicked,

                "number_of_adults": number_of_adults,
                "number_of_children": number_of_children,
                "personal_name": personal_name,
                "personal_mobile": personal_mobile,
                "personal_nin": personal_nin,
            }

            return render(request, "pay.html",context)

        else:

            messages.error(request, "Invalid data, please check again !")

            image_photo = Trip.objects.get(trip_id=trip_id_clicked)

            linked_images = LinkedImage.objects.filter(image_id=trip_id_clicked)
            return render(
                request,
                "booking.html",
                {"trip_photo": image_photo, "linked_images": linked_images},
            )

    else:
        messages.error(request, "Something went wrong !, please try again")
        trip_items = Trip.objects.all()
        return render(request, "trips.html", {"trips": trip_items})
```

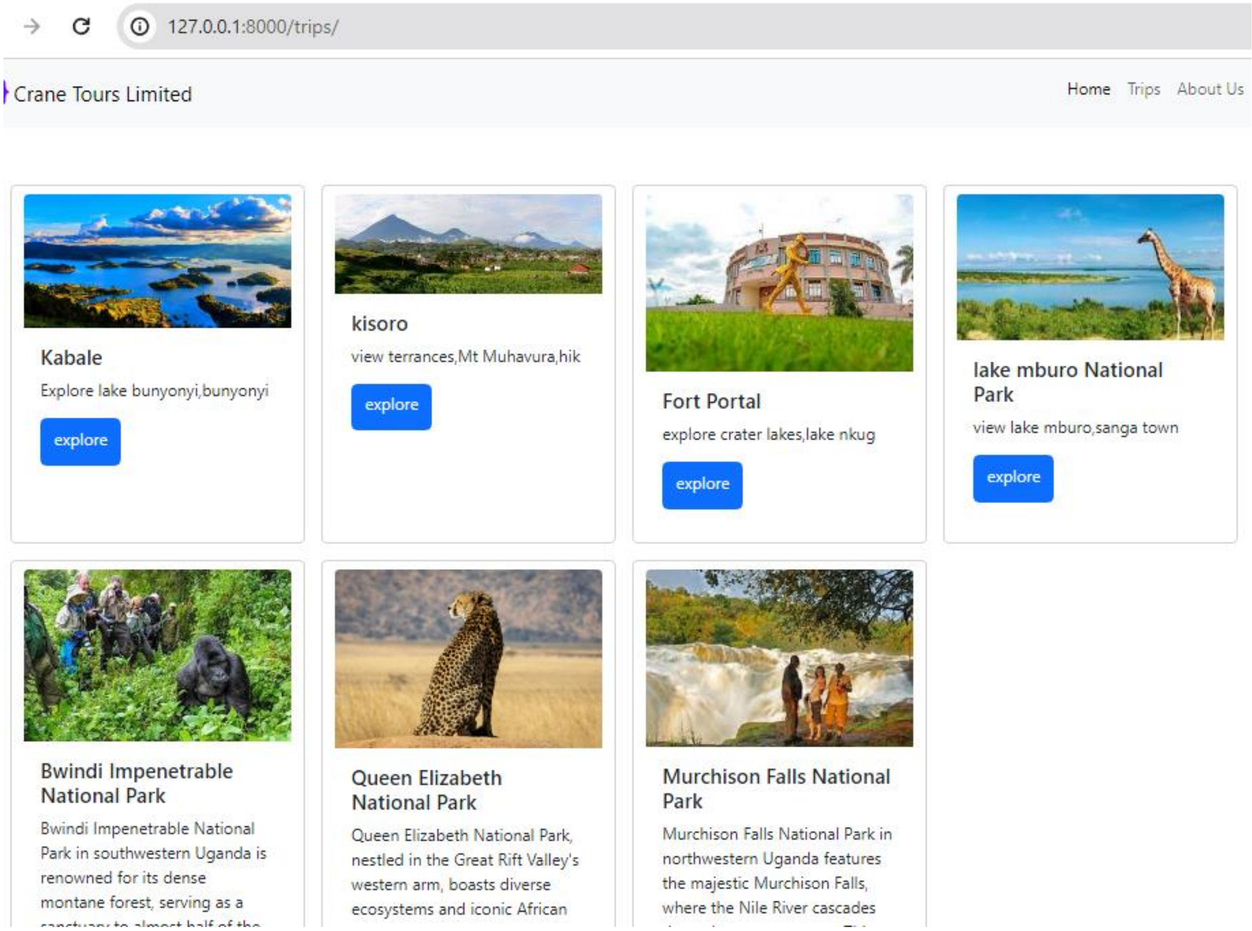


```
def BookNowView(request):

    if request.method == "GET":

        form_result = BookNowForm(request.GET)
        if form_result.is_valid():

            hotel_price = form_result.cleaned_data["hotel_price"]
            number_of_rooms = form_result.cleaned_data["number_of_rooms"]
            number_of_adults = form_result.cleaned_data["number_of_adults"]
            number_of_children = form_result.cleaned_data["number_of_children"]

            personal_name = form_result.cleaned_data["personal_name"]
            personal_mobile = form_result.cleaned_data["personal_mobile"]
            personal_nin = form_result.cleaned_data["personal_nin"]

            total = number_of_rooms * hotel_price

            context = {
                "total": total,
                "number_of_rooms": number_of_rooms,
                "hotel_id": trip_id_clicked,

                "number_of_adults": number_of_adults,
                "number_of_children": number_of_children,
                "personal_name": personal_name,
                "personal_mobile": personal_mobile,
                "personal_nin": personal_nin,
            }

            return render(request, "pay.html",context)

        else:

            messages.error(request, "Invalid data, please check again !")

            image_photo = Trip.objects.get(trip_id=trip_id_clicked)

            linked_images = LinkedImage.objects.filter(image_id=trip_id_clicked)
            return render(
                request,
                "booking.html",
                {"trip_photo": image_photo, "linked_images": linked_images},
            )

    else:
        messages.error(request, "Something went wrong !, please try again")
        trip_items = Trip.objects.all()
```



```
class FeaturedDestination(models.Model):

    def __str__(self):
        return self.dest_tittle

class HomeImages(models.Model):
    hm_id= models.CharField(max_length=230,blank=False)
    hm_image = models.ImageField(upload_to= "hm_images")

    def __str__(self):
        return self.hm_id
```

Trips page



Kabale

Explore lake bunyonyi,bunyonyi

kisoro

view terrances,Mt Muhavura,hik

Fort Portal

explore crater lakes,lake nkug

lake mburo National Park

view lake mburo,sanga town

Bwindi Impenetrable National Park

Bwindi Impenetrable National Park in southwestern Uganda is renowned for its dense montane forest, serving as a

Queen Elizabeth National Park

Queen Elizabeth National Park, nestled in the Great Rift Valley's western arm, boasts diverse ecosystems and iconic African

Murchison Falls National Park

Murchison Falls National Park in northwestern Uganda features the majestic Murchison Falls, where the Nile River cascades

chapter 5: Bibliography

Adventure Tourism Uganda. (n.d.). Home. [online] Available at: https://adventuretourismug.com/.

GeeksforGeeks. (2023). Top 45+ Django Projects with Source Code (Beginners to Advanced). [online] Available at: https://www.geeksforgeeks.org/django-projects/.

Gou.go.ug. (2023). Ministry of Tourism, Wildlife and Antiquities | Uganda National Web Portal. [online] Available at: https://www.gou.go.ug/ministry/ministry-tourism-wildlife-and-antiquities.

tourism Uganda. (n.d.). Statistics. [online] Available at: https://www.tourism.go.ug/statistics [Accessed 25 May 2024].

utb.go.ug. (n.d.). Uganda Tourism Board | Official Corporate Website. [online] Available at: https://utb.go.ug/.