FAQ chatbots website using Flask deployment on AWS EC2

PROJECT REPORT

Submitted in fulfilment for the J Component of ITA6009 – Cloud Computing

in

M.C.A

by

Rajat Singh (22MCA0139)

Hrishikesh S G (22MCA0162)

Sambit Basu (22MCA0240)

Under the guidance of

Prof. Krishnamoorthy N

SITE

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1.	Abstract	**3**
2.	Introduction	**4**
3.	Problem Statement	**5**
4.	Modules Description	**5**
5.	System Design/ER Diagram	**7**
6.	Use case diagram	**8**
7.	Implementation	**9**
8.	Result	**23**
9.	Conclusion	**27**
10.	References	**28**

Abstract

This project report presents the development and deployment of a FAQ chatbots website using Flask and AWS EC2. The objective of the website is to host multiple chatbots that provide accurate and relevant answers to user queries based on predefined topics. The chatbots were developed using ConvAI, a conversational AI framework. Flask, a Python web framework, was utilized to create the website and handle user interactions. The website offers an intuitive interface for users to input their questions and receive prompt responses from the chatbots. Each chatbot is specialized in a specific topic, ensuring accurate and reliable information delivery. AWS EC2 was chosen as the deployment platform to ensure scalability, availability, and high performance. Throughout the project, various stages were completed, including the design and development of the chatbots using ConvAI, the creation of the website using Flask, and the deployment of the website on AWS EC2. Testing and optimization were performed to enhance the performance and user experience. The FAQ chatbots website provides a user-friendly and efficient solution for accessing topic-specific information. The utilization of Flask and AWS EC2 ensures the website's reliability, scalability, and availability, making it suitable for a wide range of users and applications.

Keywords: FAQ chatbots, website, Flask, deployment, AWS EC2, ConvAI, user engagement, instant assistance, common user queries, conversational AI, web framework, intuitive interface, scalability, availability, high performance, testing, optimization, accuracy, responsiveness.

Introduction

In today's digital era, chatbots have emerged as a popular tool for enhancing user engagement and providing instant assistance. Frequently Asked Questions (FAQ) chatbots are particularly useful in addressing common user queries and providing relevant information. This project focuses on the development and deployment of a FAQ chatbots website using Flask and AWS EC2.

The main objective of this project is to create a platform that hosts multiple chatbots, each specialized in a specific topic, to efficiently address user inquiries. ConvAI, a conversational AI framework, is employed to develop the chatbot models. Flask, a lightweight and flexible web framework in Python, is utilized to design and implement the website interface.

The website aims to offer an intuitive and user-friendly experience, allowing users to input their questions and receive accurate and timely responses from the chatbots. The deployment is carried out on AWS EC2, a cloud computing service that ensures scalability, availability, and high performance.

This project encompasses various stages, including chatbot development, website creation, and deployment on AWS EC2. Testing and optimization are performed to ensure the chatbots' accuracy and the website's responsiveness.

Overall, the FAQ chatbots website provides an efficient and interactive solution for addressing user queries on a range of topics. The combination of Flask and AWS EC2 offers a reliable and scalable platform to deliver seamless user experiences.

Problem Statement

The problem addressed in this project is the need for a user-friendly and efficient solution to provide instant assistance and answer common user queries on a website. Traditional FAQ pages often lack interactivity and may not effectively address specific user concerns. Therefore, developing a website with multiple FAQ chatbots using Flask and deploying it on AWS EC2 aims to overcome these limitations and enhance user engagement. The challenge lies in designing an intuitive interface, implementing conversational AI techniques through ConvAI, ensuring scalability, availability, high performance, and optimizing the chatbots for accuracy and responsiveness.

Module Description

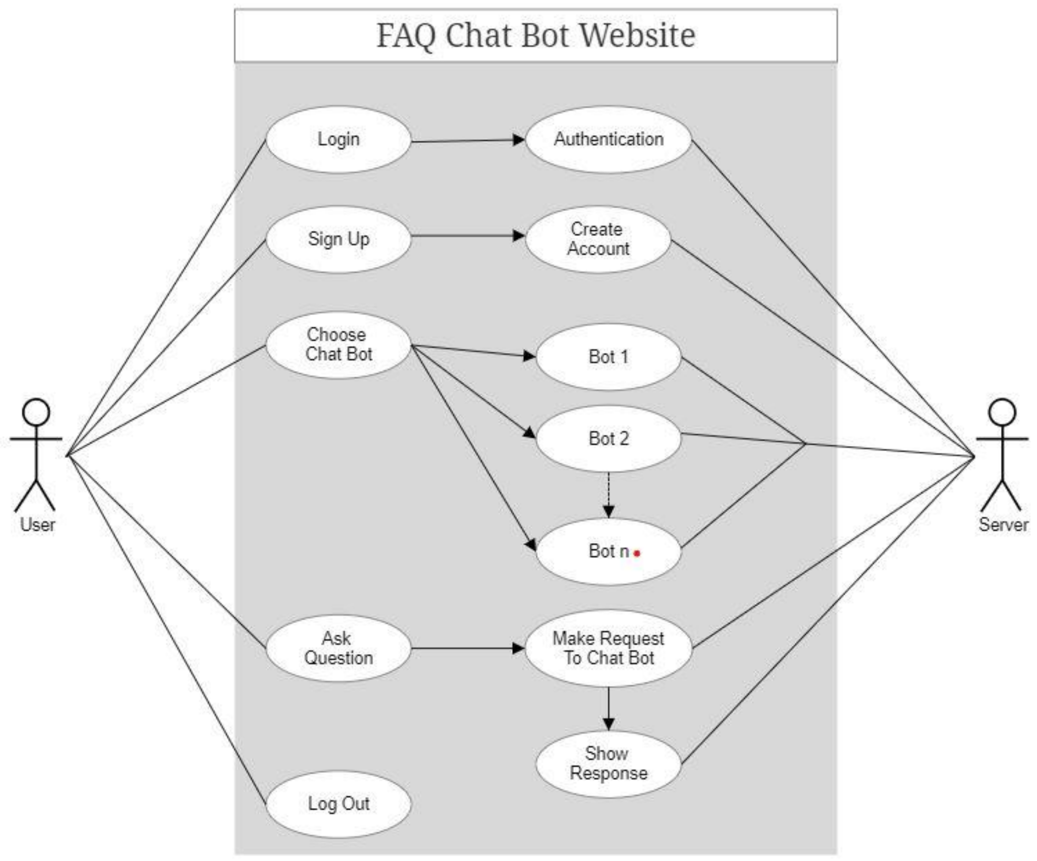
1. **Flask**: Flask is a lightweight web framework in Python used for building web applications. It provides a simple and flexible architecture for developing server-side components of the project. Flask offers features like URL routing, request handling, template rendering, and session management, making it suitable for developing the website's backend. Its modular design allows easy integration with other libraries and extensions.
2. **AWS EC2**: Amazon Elastic Compute Cloud (EC2) is a scalable and secure cloud computing service provided by Amazon Web Services (AWS). EC2 enables the deployment of virtual servers in the cloud, allowing easy scalability and high availability of the website. By hosting the project on EC2, it ensures reliable and efficient performance, automatic scaling capabilities, and seamless integration with other AWS services.
3. **ConvAI**: ConvAI is a conversational AI platform that enables the development of chatbots using natural language processing (NLP) techniques. It provides pre-trained models, language understanding capabilities, and response generation mechanisms, making it suitable for building intelligent chatbots. ConvAI allows the chatbots to understand user queries and provide relevant answers based on the given topics, enhancing the interactive and conversational experience for users.

4. **SQLite**: SQLite is a lightweight and embedded relational database management system. It is a popular choice for small to medium-sized applications due to its simplicity, portability, and minimal configuration requirements. In the project, SQLite can be used as the database module for storing and managing data related to the FAQ chatbots and user interactions. SQLite offers a self-contained, serverless architecture, allowing the database to be directly integrated into the application. It supports SQL queries, transactions, and data indexing, providing efficient data storage and retrieval. With its small footprint and compatibility with Flask, SQLite is well-suited for managing the chatbot-related data in a streamlined and efficient manner.

System Design

1. **User Interface**: The user interface is developed using Flask, which allows users to interact with the chatbot system through a web browser. Users can input their questions and receive relevant answers from the chatbots.
2. **Flask Application**: The Flask application acts as the central component of the system, handling user requests, processing input data, and generating appropriate responses. It integrates the chatbot functionality with the user interface and communicates with the chatbot modules.
3. **Chatbot Modules**: Multiple chatbot modules are developed using ConvAI, each specialized in answering questions related to specific topics. These modules utilize natural language processing (NLP) techniques and predefined knowledge bases to provide accurate and relevant responses.
4. **Database (SQLite)**: The SQLite database is used to store and manage data related to chatbot interactions, user preferences, and topic-specific information. It provides efficient data storage and retrieval capabilities, enabling seamless integration with the Flask application.
5. **AWS EC2**: The system is deployed on AWS EC2, a scalable and reliable cloud computing service. EC2 provides the necessary infrastructure to host the Flask application and make it accessible to users over the internet.

Use Case Diagram



Implementation

Github Code Link: https://github.com/rajat-singh1999/faq-chatbot-flask

Codes:

```python
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from os import path
from flask_login import LoginManager

db = SQLAlchemy()
DB_NAME = "database.db"

def create_app():
    app = Flask(__name__)
    app.config['SECRET_KEY'] = "12345678"
    app.config['SQLALCHEMY_DATABASE_URI'] = f'sqlite:///{DB_NAME}'

    db.init_app(app)

    from .views import views
    from .auth import auth

    app.register_blueprint(views, url_prefix='/')
    app.register_blueprint(auth, url_prefix='/')

    from .models import User, Prompt

    with app.app_context():
        db.create_all()

    login_manager = LoginManager()
    login_manager.login_view = 'auth.login'
    login_manager.init_app(app)

    @login_manager.user_loader
    def load_user(id):
        return User.query.get(int(id))

    return app
```

```python
from flask import Blueprint, render_template, request, flash, redirect, url_for
from .models import User
from werkzeug.security import generate_password_hash, check_password_hash
from . import db
from flask_login import login_user, login_required, logout_user, current_user

auth = Blueprint('auth', __name__)

@auth.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')

        user = User.query.filter_by(email=email).first()
        if user:
            if check_password_hash(user.password, password):
                flash('Logged in successfully!', category='success')
                login_user(user, remember=True)
                return redirect(url_for('views.index'))
            else:
                flash('Incorrect password, try again.', category='error')
        else:
            flash('User does not exist.', category='error')

    return render_template("login.html", user=current_user)

@auth.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('auth.login'))


@auth.route('/refresh')
@login_required
def refresh():
    return redirect(url_for('views.index'))

@auth.route('/sign-up', methods=['GET', 'POST'])
def sign_up():
    if request.method == 'POST':
        email = request.form.get('email')
        name = request.form.get('name')
        password1 = request.form.get('password1')
        password2 = request.form.get('password2')

        user = User.query.filter_by(email=email).first()
        if user:
            flash('Email already exists.', category='error')
        elif len(email) <4:
            flash('Email must be greater than 4 characters.', category='error')
        elif len(name) <2:
            flash('First name must be atleast 2 characters long.',
category='error')
        elif password1 != password2:
            flash('Passwords dont match!', category='error')
        elif len(password1) <7:
            flash('Password must be atleast 7 characters long.',
category='error')
        else:
            #add to database
            new_user = User(email=email, name=name,
password=generate_password_hash(password1, method='sha256'))
            db.session.add(new_user)
            db.session.commit()
            login_user(new_user, remember=True)
            flash('Account created!', category='success')
            return redirect(url_for('views.index'))

    return render_template("sign_up.html", user=current_user)
```

```
from . import db
from flask_login import UserMixin
from sqlalchemy.sql import func

class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    name = db.Column(db.String(100))
    password = db.Column(db.String(150))

class Prompt(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    char_name=db.Column(db.String(100))
    content = db.Column(db.String(10000))
```

```python
from concurrent.futures import process
from flask import Blueprint, render_template, request, flash, redirect, url_for,
jsonify
import requests
import json
from flask_login import login_required, current_user
from .models import User, Prompt
from . import db
from dotenv.main import load_dotenv
import os
load_dotenv()

views = Blueprint('views', __name__)

@views.route('/', methods=['GET', 'POST'])
@login_required
def index():
    if request.method == 'GET':
        pass

    if request.method == 'POST':
        pass

    return render_template('index.html')

@views.route('/chat/<name>', methods=['GET', 'POST'])
@login_required
def solve(name):
    chars = {
        'Chandu':'a3290b94-f6da-11ed-92a9-42010a400002',
        'Golu': '61dadc76-f6f3-11ed-aedd-42010a400002',
        'Ravi': 'c9cbcd4c-f6ec-11ed-a8e4-42010a400002',
        'Bhola': 'babad10e-f6f1-11ed-b4c2-42010a400002',
        'Gopal': '5fdecad0-f6ea-11ed-bcdf-42010a400002',
        'Ramu': '044c1734-f6f6-11ed-9e28-42010a400002'
    }

    m_names = {
            "Chandu":['Chandu: Hi there! My name is Chandu. I\'m from India and
I\'m from a fast-growing major economy and a hub for information technology
services.'],
            "Golu":['Golu: Hi! I\'m Golu. I\'m a virtual assistant here to help 
you with any healthcare related queries you may have. How may I help you?'],
            "Ravi":['Ravi: I am Ravi, I can answer anything on IPL and cricket in 
general.'],
            "Bhola":['Bhola: I can answer your queries regarding The Taj.'],
            "Gopal":['Gopal: Hi, my name is Gopal. Ask me anything about 
Bollywood!'],
            'Ramu':['Ramu: My name is Ramu, I am a banking expert. Come after 
lunch time!']
            }

    #favorite_language = os.environ['API_KEY']

    for i in m_names.keys():
        t = Prompt.query.filter_by(user_id=current_user.id, char_name=i)
        for j in t:
            if j.content not in m_names[i]:
                print(f"{i}----{j.content}")
                m_names[i].append(j.content)

    if request.method == 'POST':
        url = "https://api.convai.com/character/getResponse"
        prompt = request.form.get('message')
        if prompt is not "":
            payload={'userText': prompt,
            'charID': chars[name],
            'sessionID': '-1',
            'voiceResponse': 'False'}
            headers = {
            'CONVAI-API-KEY': '382837905acc0a7153694e508ba43307'
            }

            response = requests.request("POST", url, headers=headers, 
data=payload)
            data = response.json()
            character_response = data["text"]

            user_prompt = current_user.name+": "+prompt
            character_response = name+": "+character_response

            new_prompt1 = Prompt(char_name=name,user_id=current_user.id, 
content=user_prompt)
            new_prompt2 = Prompt(char_name=name,user_id=current_user.id, 
content=character_response)
            db.session.add(new_prompt1)
            db.session.add(new_prompt2)
            db.session.commit()

            m_names[name].append(user_prompt)
            m_names[name].append(character_response)

    l = m_names[name][::-1]
    return render_template('chat.html', messages=l, name=name)
```

```
from website import create_app

app = create_app()


if __name__ == '__main__':
    app.run()
```

```
{% extends "index.html" %}
{% block title %}Chat{% endblock %}

{% block content %}

<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbar">
        <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbar">
        <div class="navbar-nav">
            <a class="nav-item nav-link" id="home" href="/">Home</a>
            <a class="nav-item nav-link" id="home" href="/chat/Chandu">Chandu</a>
            <a class="nav-item nav-link" id="home" href="/chat/Bhola">Bhola</a>
            <a class="nav-item nav-link" id="home" href="/chat/Gopal">Gopal</a>
            <a class="nav-item nav-link" id="home" href="/chat/Golu">Golu</a>
            <a class="nav-item nav-link" id="home" href="/chat/Ravi">Ravi</a>
            <a class="nav-item nav-link" id="home" href="/chat/Ramu">Ramu</a>
            <a class="nav-item nav-link" id="logout" href="/logout">Logout</a>

        </div>
    </div>
</nav>
```

```
<p class="d-none">{ c = "/chat/"+name }</p>
<br />
<br />
<div class="container">
    <h2>{{ name }}</h2>
</div>
<br />
<div class="container" action="{{ c }}">
    <form method="POST">
        <textarea name="message" id="message" class="form-control"></textarea>
        <br />
        <div align="center">
        <button type="submit" class="btn btn-dark">Go!</button>
        </div>
    </form>
</div>

  <div class="container">
    <ul class="list-group list-group-flush" id="notes">
    {% for m in messages %}
    <li class="list-group-item">
        <div class="container">
            <div class="container"><h5>{{ m }}</h5></div>

        </div>
    </li>
    <br />
    {% endfor %}
    </ul>
</div>



{% endblock %}
```

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <link
      rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
```

```
      integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
      crossorigin="anonymous"
    />
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-
awesome.min.css"
      crossorigin="anonymous"
      />
    </head>

    <title>{% block title %}HomePage{% endblock %}</title>
    <body align="center">
        {% block content %}

        <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
          <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbar">
              <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbar">
              <div class="navbar-nav">
                  <a class="nav-item nav-link" id="home" href="/">Home</a>
                  <a class="nav-item nav-link" id="logout"
href="/logout">Logout</a>
              </div>
          </div>
      </nav>
        <br />
        <br />
        <div class="container">
          <h1>These are the currently available FAQ bots.</h1>
          <p>You may ask them questions pertaing to the topics assigned to
them.</p>
          <p>Please do get creative and ask away anything! Our bots will
definetely have answers!!!</p>
        </div>

        <div class="container">

        <div class=" col g-3">
          <div class="row" >
          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
```

```
            <img src="{{ url_for('static', filename='india.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Chandu</h5>
              <p class="card-text">ask me anything about India!</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Chandu">Chat</a>
            </div>
          </div>

          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
            <img src="{{ url_for('static', filename='appo.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Golu</h5>
              <p class="card-text">ask me about Appolo hospitals and its
services.</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Golu">Chat</a>
            </div>
          </div>

          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
            <img src="{{ url_for('static', filename='ipl.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Ravi</h5>
              <p class="card-text">ask me anything about IPL and cricket in
general.</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Ravi">Chat</a>
            </div>
          </div>
          </div>
          <div class="row">
          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
            <img src="{{ url_for('static', filename='taj.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Bhola</h5>
              <p class="card-text">i am the FAQ bot for the Taj</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Bhola">Chat</a>
            </div>
          </div>
          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
            <img src="{{ url_for('static', filename='bolly.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Gopal</h5>
              <p class="card-text">ask me anything about bollywood!</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Gopal">Chat</a>
            </div>
          </div>

          <div class="card" style="width: 18rem; margin: 15px 15px 15px 15px">
            <img src="{{ url_for('static', filename='Bank.jpg') }}" class="card-
img-top" alt="...">
            <div class="card-body">
              <h5 class="card-title">Ramu</h5>
              <p class="card-text">The Banking expert.</p>
              <a type="button"  class="btn btn-dark btn-lg btn-block"
href="/chat/Ramu">Chat</a>
            </div>
          </div>
        </div>
        </div>
      </div>
        {% endblock %}

        <script
        src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
        integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
        crossorigin="anonymous"
      ></script>
      <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
        integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
        crossorigin="anonymous"
      ></script>
      <script
        src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
        integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
        crossorigin="anonymous"
      ></script>
      <script
          type="text/javascript"
          src="{{ url_for('static', filename='index.js') }}"
          ></script>
    </body>
</html>
```

```
{% extends "index.html" %}
{% block title %}Login{% endblock %}

{% block content %}

<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbar">
      <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbar">
      <div class="navbar-nav">
          {% if user.is_authenticated %}
          <a class="nav-item nav-link" id="home" href="/">Home</a>
          <a class="nav-item nav-link" id="logout" href="/logout">Logout</a>
          {% else %}
          <a class="nav-item nav-link" id="login" href="/login">Login</a>
          <a class="nav-item nav-link" id="signUp" href="/sign-up">Sign-Up</a>
          {% endif %}
      </div>
  </div>
</nav>
{% with messages = get_flashed_messages(with_categories=true) %}

{% if messages %}
  {% for category, message in messages %}
      {% if category == 'error' %}
      <div class = 'alert-danger alert-dismissable fade show' role="'alert">
          {{ message }}
          <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
          </button>
      </div>
      {% else %}
      <div class = 'alert-success alert-dismissable fade show' role="'alert">
          {{ message }}
          <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
          </button>
      </div>
      {% endif %}
  {% endfor %}
{% endif %}
{% endwith %}

<br />

<div class="container" style="margin: 0% 10% 10% 10%">

  <form method="POST">
    <h3 align="center">Login</h3>
    <div class="form-group">
      <label for="email">Email Address</label>
      <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
      />
    </div>
    <div class="form-group">
      <label for="password">Password</label>
      <input
      type="password"
      class="form-control"
      id="password"
      name="password"
      placeholder="Enter password"
      />
    </div>
    <br />
    <button type="submit" class="btn btn-primary">Login</button>
  </form>
</div>
  {% endblock %}
```

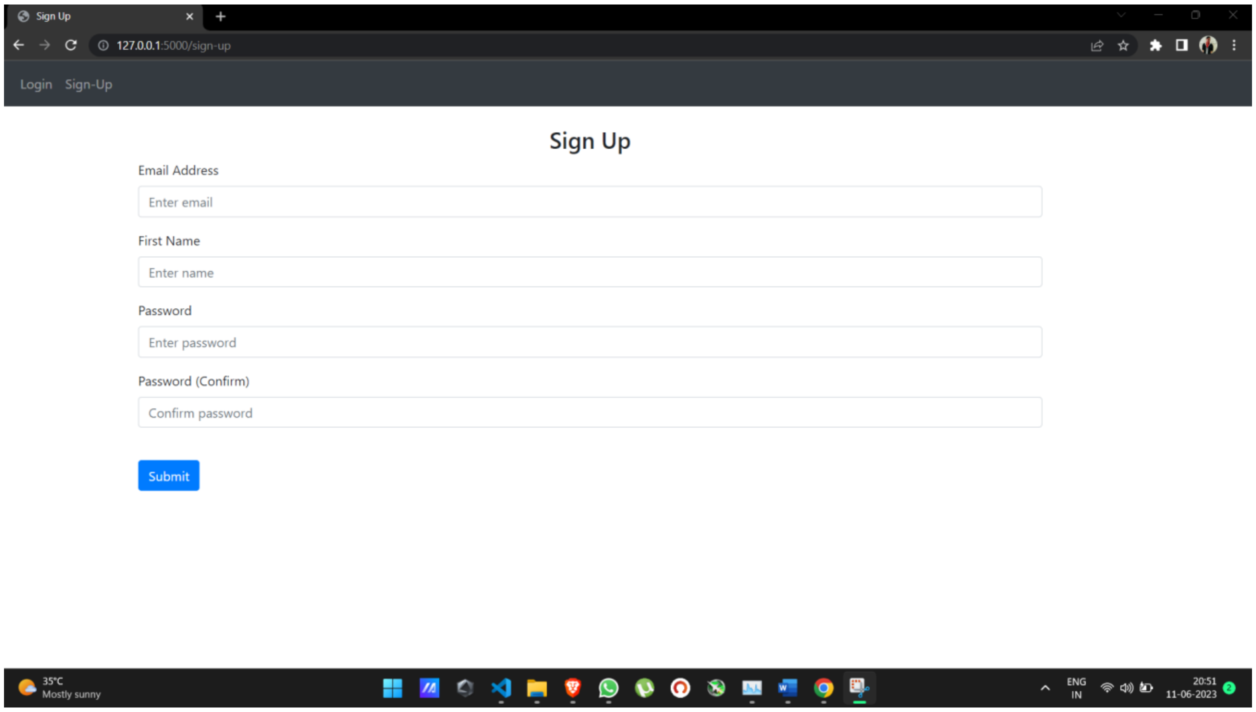
```
{% extends "index.html" %}
{% block title %}Sign Up{% endblock %}
{% block content %}
```

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbar">
      <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbar">
      <div class="navbar-nav">
          {% if user.is_authenticated %}
          <a class="nav-item nav-link" id="home" href="/">Home</a>
          <a class="nav-item nav-link" id="logout" href="/logout">Logout</a>
          {% else %}
          <a class="nav-item nav-link" id="login" href="/login">Login</a>
          <a class="nav-item nav-link" id="signUp" href="/sign-up">Sign-Up</a>
          {% endif %}
      </div>
  </div>
</nav>
{% with messages = get_flashed_messages(with_categories=true) %}

{% if messages %}
  {% for category, message in messages %}
      {% if category == 'error' %}
      <div class = 'alert-danger alert-dismissable fade show' role="'alert">
          {{ message }}
          <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
          </button>
      </div>
      {% else %}
      <div class = 'alert-success alert-dismissable fade show' role="'alert">
          {{ message }}
          <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
          </button>
      </div>
      {% endif %}
  {% endfor %}
{% endif %}
{% endwith %}
<br />
<div class="container" style="margin: 0% 10% 10% 10%">
<form method="POST">
    <h3 align="center">Sign Up</h3>
    <div class="form-group">
      <label for="email">Email Address</label>
      <input
        type="email"
        class="form-control"
        id="email"
        name="email"
        placeholder="Enter email"
      />
    </div>
    <div class="form-group">
      <label for="name">First Name</label>
      <input
        type="text"
        class="form-control"
        id="name"
        name="name"
        placeholder="Enter name"
      />
    </div>
    <div class="form-group">
      <label for="password1">Password</label>
      <input
        type="password"
        class="form-control"
        id="password1"
        name="password1"
        placeholder="Enter password"
      />
    </div>
    <div class="form-group">
      <label for="password2">Password (Confirm)</label>
      <input
        type="password"
        class="form-control"
        id="password2"
        name="password2"
        placeholder="Confirm password"
      />
    </div>
    <br />
    <button type="submit" class="btn btn-primary">Submit</button>
  </form>
</div>
{% endblock %}
```

Results

The results are in the form of output:



Sign Up

127.0.0.1:5000/sign-up

Login Sign-Up

Sign Up

Email Address

Enter email

First Name

Enter name

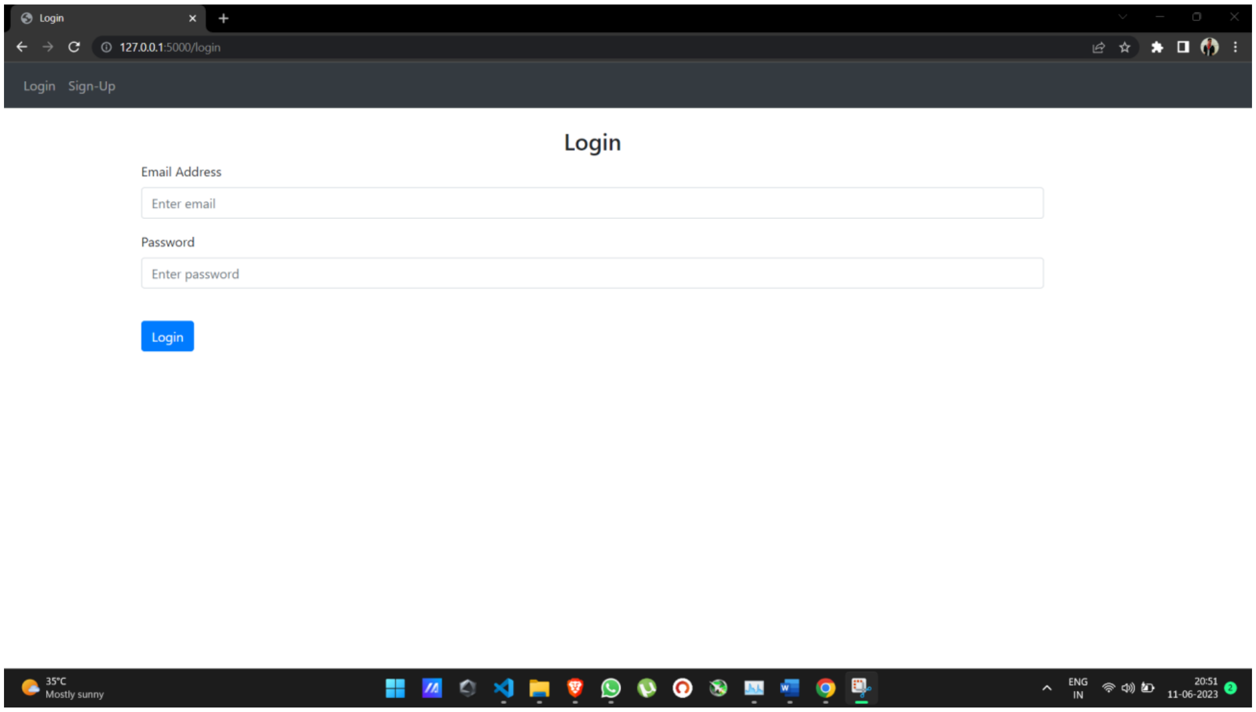
Password

Enter password

Password (Confirm)

Confirm password

Submit



Login

127.0.0.1:5000/login

Login Sign-Up

Login

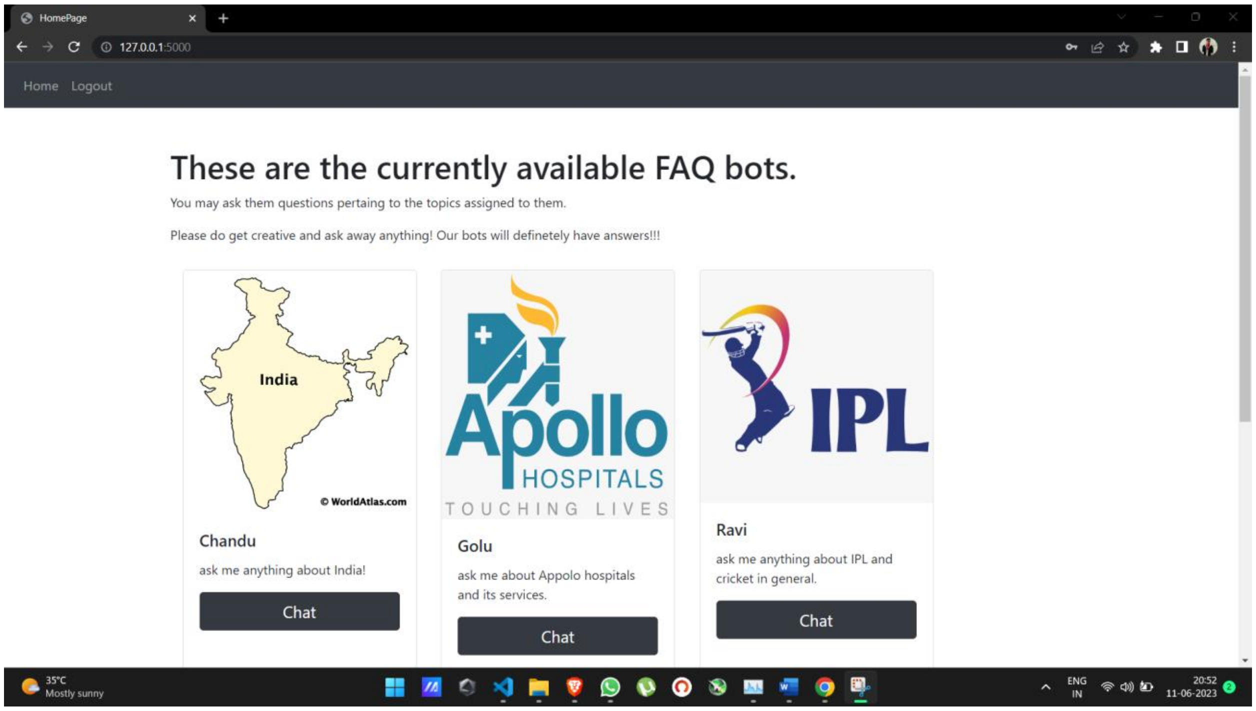
Email Address

Enter email

Password

Enter password

Login



Home Logout

These are the currently available FAQ bots.

You may ask them questions pertaing to the topics assigned to them.

Please do get creative and ask away anything! Our bots will definetely have answers!!!

Chandu

ask me anything about India!

Chat

Golu

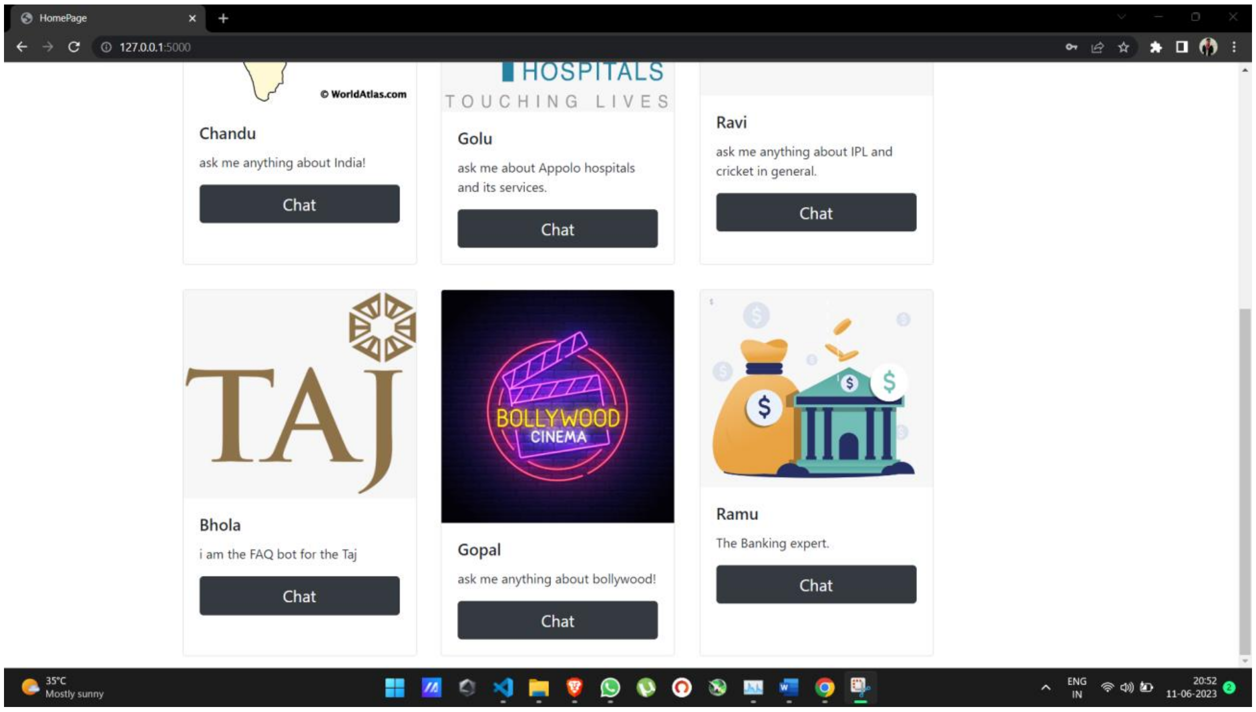
ask me about Appolo hospitals and its services.

Chat

Ravi

ask me anything about IPL and cricket in general.

Chat



Chandu

ask me anything about India!

Chat

Golu

ask me about Appolo hospitals and its services.

Chat

Ravi

ask me anything about IPL and cricket in general.

Chat

Bhola

i am the FAQ bot for the Taj

Chat

Gopal

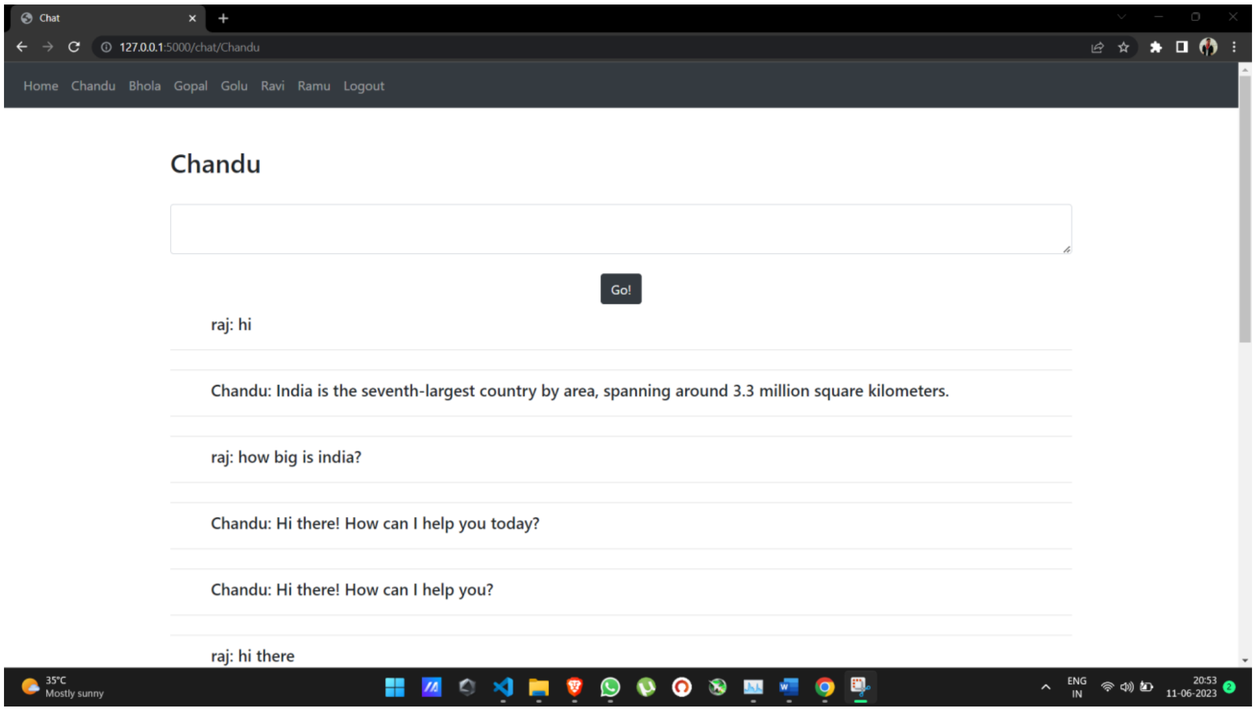
ask me anything about bollywood!

Chat

Ramu

The Banking expert.

Chat


Chat
127.0.0.1:5000/chat/Chandu
Home Chandu Bhola Gopal Golu Ravi Ramu Logout
Go!
35°C Mostly sunny
ENG IN
20:53 11-06-2023

Chandu

raj: hi

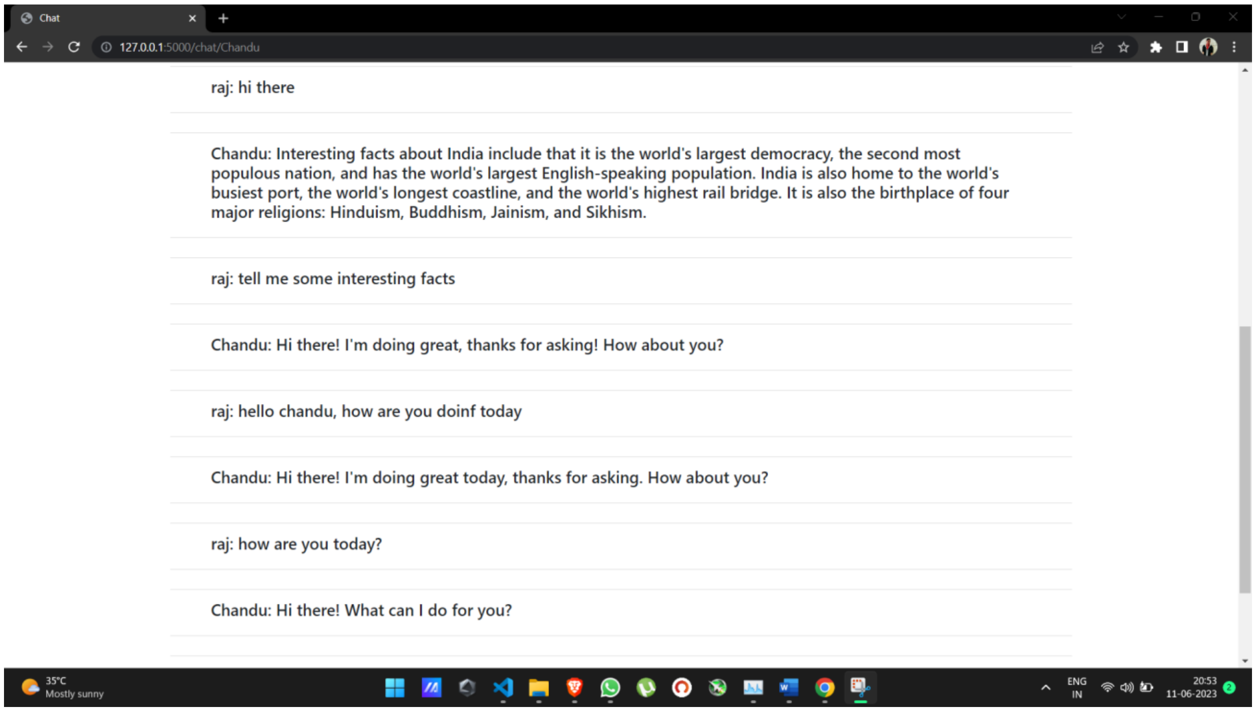
Chandu: India is the seventh-largest country by area, spanning around 3.3 million square kilometers.

raj: how big is india?

Chandu: Hi there! How can I help you today?

Chandu: Hi there! How can I help you?

raj: hi there



raj: hi there

Chandu: Interesting facts about India include that it is the world's largest democracy, the second most populous nation, and has the world's largest English-speaking population. India is also home to the world's busiest port, the world's longest coastline, and the world's highest rail bridge. It is also the birthplace of four major religions: Hinduism, Buddhism, Jainism, and Sikhism.

raj: tell me some interesting facts

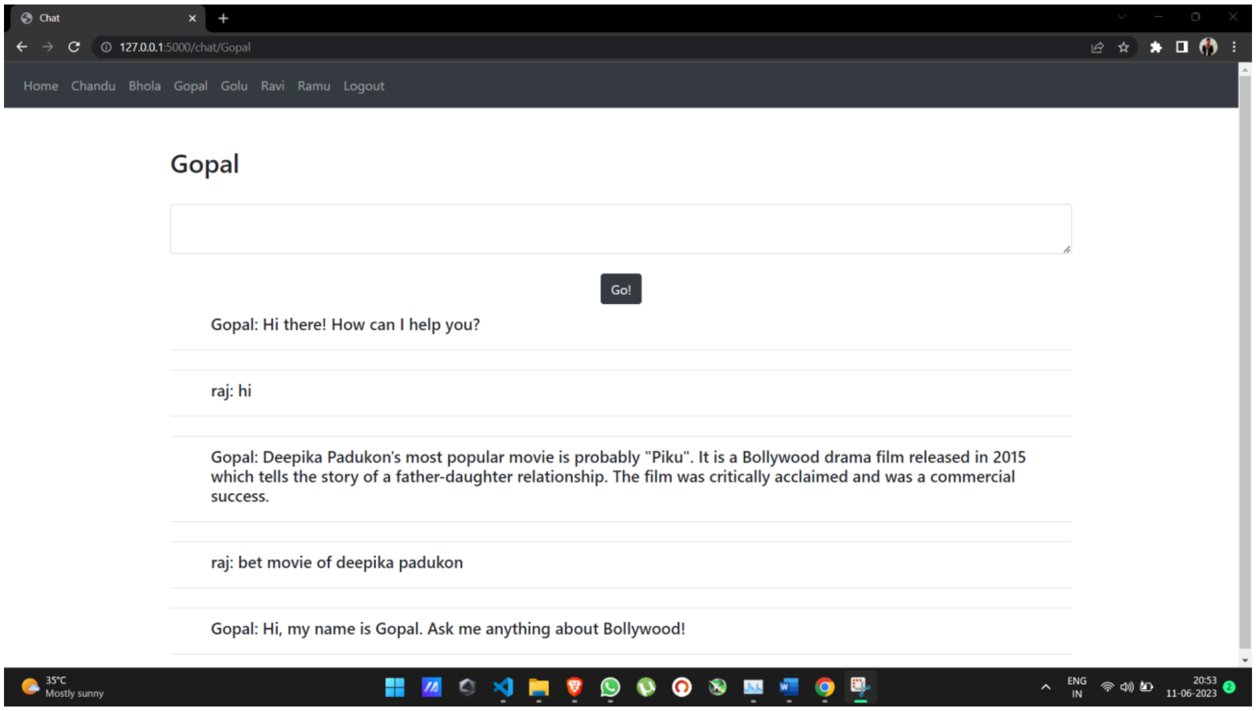
Chandu: Hi there! I'm doing great, thanks for asking! How about you?

raj: hello chandu, how are you doinf today

Chandu: Hi there! I'm doing great today, thanks for asking. How about you?

raj: how are you today?

Chandu: Hi there! What can I do for you?


Chat
127.0.0.1:5000/chat/Gopal
Home Chandu Bhola Gopal Golu Ravi Ramu Logout
Go!
35°C
Mostly sunny
20:53
11-06-2023

Gopal

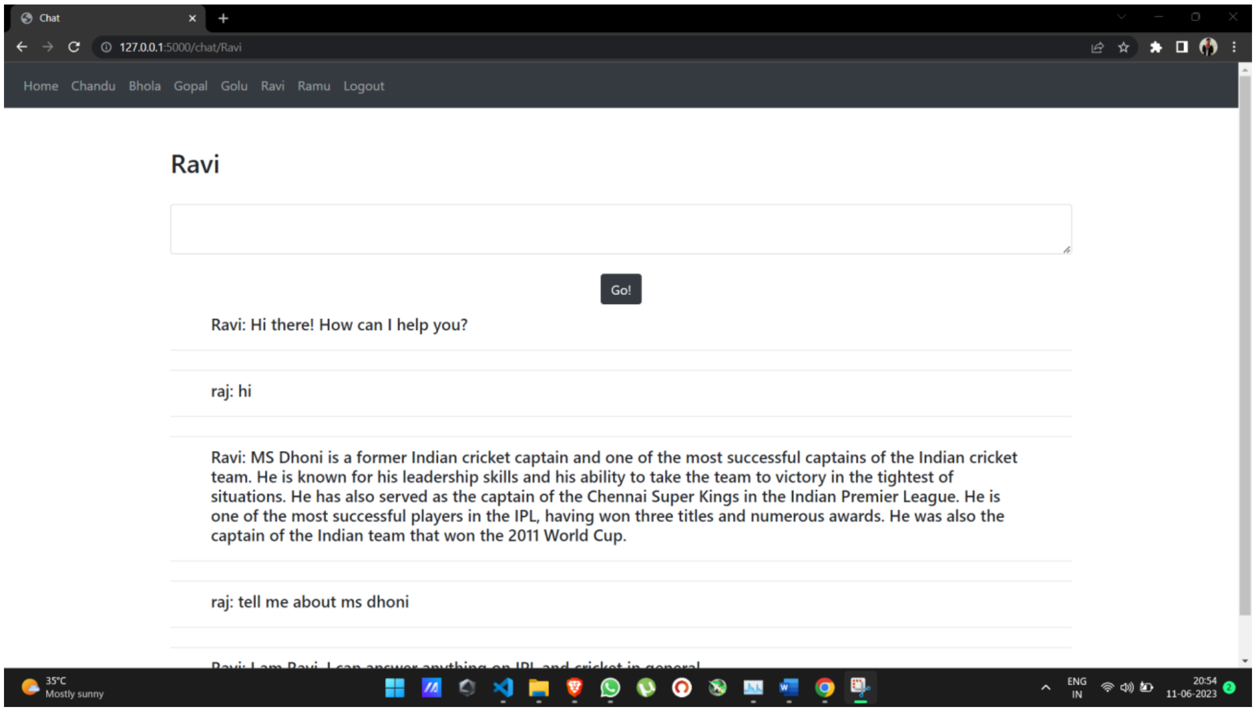
Gopal: Hi there! How can I help you?

raj: hi

Gopal: Deepika Padukon's most popular movie is probably "Piku". It is a Bollywood drama film released in 2015 which tells the story of a father-daughter relationship. The film was critically acclaimed and was a commercial success.

raj: bet movie of deepika padukon

Gopal: Hi, my name is Gopal. Ask me anything about Bollywood!



Ravi

Ravi: Hi there! How can I help you?

raj: hi

Ravi: MS Dhoni is a former Indian cricket captain and one of the most successful captains of the Indian cricket team. He is known for his leadership skills and his ability to take the team to victory in the tightest of situations. He has also served as the captain of the Chennai Super Kings in the Indian Premier League. He is one of the most successful players in the IPL, having won three titles and numerous awards. He was also the captain of the Indian team that won the 2011 World Cup.

raj: tell me about ms dhoni

Ravi: I am Ravi. I can answer anything on IPL and cricket in general

Conclusion

In conclusion, the development of the FAQ chatbots website using Flask deployment on AWS EC2 has been a successful endeavour. The project aimed to provide a user-friendly platform for accessing multiple chatbots specialized in answering questions on various topics. Using Flask and ConvAI, we were able to create a responsive and interactive user interface integrated with topic-specific chatbot modules.

The system design effectively incorporated key components such as the Flask application, chatbot modules, SQLite database for data management, and deployment on AWS EC2 for scalability and reliability. The implementation of these modules allowed users to easily input their questions and receive accurate responses from the chatbots.

The project successfully addressed the problem of efficiently providing relevant information to users by leveraging the capabilities of chatbot technology and web development frameworks. The deployment on AWS EC2 ensured the website's accessibility and performance.

Moving forward, potential improvements could include enhancing the chatbot modules' natural language processing capabilities, expanding the knowledge bases, and incorporating user feedback for continuous improvement. Overall, the FAQ chatbots website serves as a valuable resource for users seeking quick and accurate answers to their queries, providing a seamless and intuitive user experience.

References

[1] Flask Documentation. (n.d.). Retrieved from https://flask.palletsprojects.com/

[2] ConvAI Documentation. (n.d.). Retrieved from https://github.com/DeepPavlov/convai

[3] Amazon EC2 Documentation. (n.d.). Retrieved from https://aws.amazon.com/ec2/

[4] SQLite Documentation. (n.d.). Retrieved from https://www.sqlite.org/docs.html

[5] Pallets Projects. (n.d.). Retrieved from https://palletsprojects.com/

[6] DeepPavlov. (n.d.). Retrieved from https://github.com/deepmipt/DeepPavlov

[7] Python Software Foundation. (n.d.). Retrieved from https://www.python.org/

[8] W3Schools. (n.d.). Retrieved from https://www.w3schools.com/

[9] Flask-WTF Documentation. (n.d.). Retrieved from https://flask-wtf.readthedocs.io/

[10] Jinja Documentation. (n.d.). Retrieved from https://jinja.palletsprojects.com/