Anjuman-I-Islam’s

M.H. Saboo Siddik Polytechnic



Subject name: Programming with python (PWP) - 22616

Department: COMPUTER ENGINEERING

SemeSter: SiXTH

micro project title: Chat Application (PyCHat)

Prepared by:

Year: 2023-24

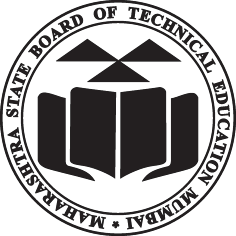
Names of Team Members with Roll Nos.

1. Abdurrahman Qureshi - 210451

2. Oaish Qazi - 210455

3. Shaikh Mohammed Hussain - 220486

Under the guidance of: Prof. Zaibunnisa Malik



**Maharashtra State**

**Board of Technical Education**

**Certificate**

This is to certify that Mr. Abdurrahman Qureshi of Sixth Semester of Diploma in Computer Engineering of Institute M.H. Saboo Siddik Polytechnic has successfully completed Micro-project work in subject Programming With Python (22616) for the academic year 2023- 2024 as prescribed in the I-Scheme Curriculum.

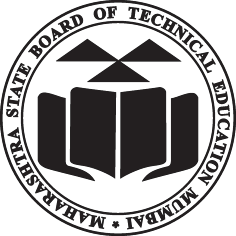
Place: …………………… Enrollment no: .…………………..

Date: …………………… Exam seat no: …………………....

|  |  |  |
| --- | --- | --- |
| **Signature** | **Signature** | **Signature** |
| **Project Guide** | **H. O. D** | **Principal** |

Seal of

Institute



**Maharashtra State**

**Board of Technical Education**

**Certificate**

This is to certify that Mr. Oaish Qazi of Sixth Semester of Diploma in Computer Engineering of Institute M.H. Saboo Siddik Polytechnic has successfully completed Micro-project work in subject Programming With Python (22616) for the academic year 2023-2024 as prescribed in the I-Scheme Curriculum.

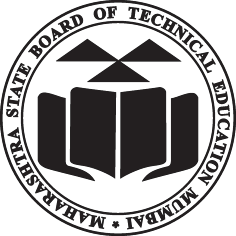
Place: …………………… Enrollment no: .…………………..

Date: …………………… Exam seat no: …………………....

|  |  |  |
| --- | --- | --- |
| **Signature** | **Signature** | **Signature** |
| **Project Guide** | **H. O. D** | **Principal** |

Seal of

Institute



**Maharashtra State**

**Board of Technical Education**

**Certificate**

This is to certify that Mr. Shaikh Mohammed Hussain of Sixth Semester of Diploma in Computer Engineering of Institute M.H. Saboo Siddik Polytechnic has successfully completed Micro-project work in subject Programming With Python (22616) for the academic year 2023-2024 as prescribed in the I-Scheme Curriculum.

Place: …………………… Enrollment no: .…………………..

Date: …………………… Exam seat no: …………………....

|  |  |  |
| --- | --- | --- |
| **Signature** | **Signature** | **Signature** |
| **Project Guide** | **H. O. D** | **Principal** |

Seal of

Institute

Acknowledgment

We wish to express our profound gratitude to our guide Ms. Zaibunnisa Malik who guided us endlessly in the framing and completion of the micro project. She guided us on all the main points in that micro project. We are indebted to his constant encouragement, cooperation, and help. It was his enthusiastic support that helped us in overcoming various obstacles in the micro-project.

We are also thankful to our Principal, HOD, faculty members and classmates of Computer Engineering department for extending their support and motivation in the completion of this micro-project.

Names of Team Members with Roll Nos.

1. Abdurrahman Qureshi - 210454

2. Oaish Qazi - 210455

3. Shaikh Mohammed Hussain - 220486

***Micro-Project Proposal***

**Chat Application**

**Annexure-I**

**1.0 Aims/Benefits of the Micro-Project**

* The aim of this micro-project is to create a real-time chat application using web technologies, with a specific focus on learning and applying Django for server-side programming. By undertaking this project, you can gain valuable hands-on experience in developing interactive web applications. The benefits of this endeavor include enhancing your understanding of Python-based web development, improving your proficiency in Django, and web application architecture, and achieving the creation of a functional chat application for personal or educational use. Additionally, this project presents the opportunity to potentially publish your application online, serving as a showcase of your web development skills.

**2.0 Course Outcomes Addressed**

* Develop python program to demonstrate use of Operators
* Perform operations on data structures in Python.
* Develop functions for given problem.
* Design classes for given problem.
* Handle exceptions.

**3.0 Proposed Methodology**

* Discussion of topic with guide and group members.
* Dividing the work and Gathering information about the project.
* Submission of project proposal. (Annexure I).
* Collection and Analysis of data / information from various sources.
* Set up the development environment (e.g., PyCharm, Django server).
* Design the user interface for the chat application using HTML/CSS templates.
* Implement Django routes to handle user registration, login, and chat functionality.
* Utilize Django-SocketIO for real-time communication between users.
* Store chat messages in a database (e.g., SQLite, MySQL) for persistence.
* Apply security measures (e.g., CSRF protection, password hashing) to protect against common web vulnerabilities.
* Test the Project for Bugs and Errors.
* Preparation of the project report (Annexure II).
* Microproject Submission and Viva.

**4.0 Action Plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Week** | **Details of activity** | **Planned**  **Start date** | **Planned**  **Finish date** | **Name of**  **Responsible Team Members** |
| 1 | 1 & 2 | Discussion and finalization of the |  |  | All |
| 2 | 3 & 4 | Dividing the work among group members. |  |  | All |
| 3 | 5 | Submission of micro project proposal  (Annexure I) |  |  | All |
| 4 | 6 | Collection of the information on the  Topic. |  |  | All |
| 6 | 7 | Collection of all relevant content / materials for the execution of the project. |  |  | All |
| 7 | 8 & 9 | Execution of collected data and preparing layout of the web app and building web app and websockets. |  |  | Oaish / Abdurrahman |
| 8 | 10 | Integration of frontend with backend |  |  | Oaish / Abdurrahman |
| 9 | 11 | Testing the project for bugs and errors |  |  | Oaish / Abdurrahman |
| 10 | 12 | Preparation of the project report  (Annexure II) |  |  | All |
| 11 | 13 | Microproject Submission and Viva |  |  | All |

**5.0 Resources Required**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name of Resource/material** | **Specifications** | **Quantity** | **Remarks** |
| 1 | Software |  | 1 of Each |  |
| Git, Firefox, Jetbrains  Pycharm |  |
| Pycharm |  |
|  |  |
| 2 | CASE Tools | Pycharm, Django, VS Code, Notepad | 1 of Each |  |
| 3 | Browser | Opera GX, Firefox, Chrome, Edge | 1 of Each |  |

**Names of Team Members with Roll Nos.**

1. Abdurrahman Qureshi - 210454

2. Oaish Qazi - 210455

3. Shaikh Mohammed Hussain - 220486

**(To be approved by the concerned teacher)**

\*\*\*\*\*\*\*\*\*\*\*

**Micro-Project Report**

1. **Rationale**
2. **Chat Application**

With the increasing popularity of online communication and social networking, there is a growing demand for real-time chat applications. Building a chat application using Django REST Framework provides an opportunity to leverage Django's powerful features for web development, including authentication, authorization, and database management. Additionally, using RESTful APIs allows for seamless integration with frontend frameworks and ensures scalability and maintainability of the application.

**2.0 Aims/Benefits of the Micro-Project:**

The aim of this micro-project is to create a real-time chat application using web technologies, with a specific focus on learning and applying Django for server-side programming. By undertaking this project, you can gain valuable hands-on experience in developing interactive web applications. The benefits of this endeavor include enhancing your understanding of Python-based web development, improving your proficiency in Django, and web application architecture, and achieving the creation of a functional chat application for personal or educational use. Additionally, this project presents the opportunity to potentially publish your application online, serving as a showcase of your web development skills

1. **Course Outcomes Achieved**

* Develop python program to demonstrate use of Operators
* Perform operations on data structures in Python.
* Develop functions for given problem.
* Design classes for given problem.
* Handle exceptions.

1. **Literature Review**

• Research existing chat applications and their architectures.

• Study Django and Jinja documentation and tutorials.

• Review best practices for securing web applications.

• Explore WebSocket communication in web development.

• Learn about database integration for web applications (e.g., SQLAlchemy).

1. **Actual Methodology Followed**

* Discussed the topic with guide and group members.
* Divided the work and Gathered information about the project.
* Submitted project proposal. (Annexure I).
* Collected and Analyzed data / information from various sources.
* The development environment set up (e.g., PyCharm, Django server).
* Designed the user interface for the chat application using HTML/CSS templates.
* Implemented Django routes to handle user registration, login, and chat functionality.
* Utilized Django-SocketIO for real-time communication between users.
* Stored chat messages in a database (e.g., SQLite, MySQL) for persistence.
* Tested the Project for Bugs and Errors.
* Prepared the project report (Annexure II).
* Microproject Submitted with Viva.

**6.0 Actual Resources Used** (Mention the actual resources used).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name of Resource/material** | **Specifications** | **Quantity** | **Remarks** |
| 1 | Software | Pycharm, FireFox, | 1 |  |
| 2 | Websites | web.whatsapp.com, youtube.com, geeksforgeeks.com | 1 |  |

**7.0 Outputs of the Micro-Projects**

**Java Swing**

**Java Swing tutorial** is a part of Java Foundation Classes (JFC) that is *used to create window-based applications*. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

Unlike AWT, Java Swing provides platform-independent and lightweight components.

The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

* **Difference between AWT and Swing**

There are many differences between java awt and swing that are given below.

|  |  |  |
| --- | --- | --- |
| No. | Java AWT | Java Swing |
| 1) | AWT components are **platform-dependent**. | Java swing components are **platform-independent**. |
| 2) | AWT components are **heavyweight**. | Swing components are **lightweight**. |
| 3) | AWT **doesn't support pluggable look and feel**. | Swing **supports pluggable look and feel**. |
| 4) | AWT provides **less components** than Swing. | Swing provides **more powerful components** such as tables, lists, scrollpanes, colorchooser, tabbedpane etc. |
| 5) | AWT **doesn't follows MVC**(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view. | Swing **follows MVC**. |

* **Hierarchy of Java Swing classes**
* The hierarchy of java swing API is given below**.**

****

* **Commonly used Methods of Component class**

The methods of Component class are widely used in java swing that are given below.

|  |  |
| --- | --- |
| Method | Description |
| public void add(Component c) | add a component on another component. |
| public void setSize(int width,int height) | sets size of the component. |
| public void setLayout(LayoutManager m) | sets the layout manager for the component. |
| public void setVisible(boolean b) | sets the visibility of the component. It is by default false. |

* **Java JDBC**

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database. There are four types of JDBC drivers:

* + JDBC-ODBC Bridge Driver,
  + Native Driver,
  + Network Protocol Driver, and
  + Thin Driver

We can use JDBC API to access tabular data stored in any relational database. By the help of JDBC API, we can save, update, delete and fetch data from the database. It is like Open Database Connectivity (ODBC) provided by Microsoft.



The current version of JDBC is 4.3. It is the stable release since 21st September, 2017. It is based on the X/Open SQL Call Level Interface. The **java.sql** package contains classes and interfaces for JDBC API. A list of popular *interfaces* of JDBC API are given below:

* Driver interface
* Connection interface
* Statement interface
* PreparedStatement interface
* CallableStatement interface
* ResultSet interface
* ResultSetMetaData interface
* DatabaseMetaData interface
* RowSet interface
  + - A list of popular *classes* of JDBC API are given below:
* DriverManager class
* Blob class
* Clob class
* Types class
* **Why Should We Use JDBC**

Before JDBC, ODBC API was the database API to connect and execute the query with the database. But, ODBC API uses ODBC driver which is written in C language (i.e. platform dependent and unsecured). That is why Java has defined its own API (JDBC API) that uses JDBC drivers (written in Java language).

We can use JDBC API to handle database using Java program and can perform the following activities:

1. Connect to the database
2. Execute queries and update statements to the database
3. Retrieve the result received from the database.

There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:



* **Servlets**

**Servlet** technology is used to create a web application (resides at server side and generates a dynamic web page).

**Servlet** technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language. However, there were many disadvantages to this technology. We have discussed these disadvantages below.

There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

* **What is a Servlet?**

Servlet can be described in many ways, depending on the context.

* Servlet is a technology which is used to create a web application.
* Servlet is an API that provides many interfaces and classes including documentation.
* Servlet is an interface that must be implemented for creating any Servlet.
* Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
* Servlet is a web component that is deployed on the server to create a dynamic web page.



* **What is a web application?**

A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.

* **CGI (Common Gateway Interface)**

CGI technology enables the web server to call an external program and pass HTTP request information to the external program to process the request. For each request, it starts a new process.



* **Disadvantages of CGI**

There are many problems in CGI technology:

1. If the number of clients increases, it takes more time for sending the response.
2. For each request, it starts a process, and the web server is limited to start processes.
3. It uses platform dependent language e.g. [C](https://www.javatpoint.com/c-programming-language-tutorial), [C++](https://www.javatpoint.com/cpp-tutorial), [perl](https://www.javatpoint.com/perl-tutorial).

* **Advantages of Servlet**



There are many advantages of Servlet over CGI. The web container creates threads for handling the multiple requests to the Servlet. Threads have many benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The advantages of Servlet are as follows:

1. Better performance: because it creates a thread for each request, not process.
2. Portability: because it uses Java language.
3. Robust: [JVM](https://www.javatpoint.com/jvm-java-virtual-machine) manages Servlets, so we don't need to worry about the memory leak, [garbage collection](https://www.javatpoint.com/Garbage-Collection), etc.
4. Secure: because it uses java language.

**Registration/templates/registration/login.html**

{% extends "registration/RegisterBase.html" %}  
{% block title %} Login {% endblock %}  
{% load crispy\_forms\_tags %}  
  
{% block content %}  
 <div class="sign-up form" style="max-width: 600px">  
 <h1 class="mt-2">User Login</h1>  
 <hr class="mt-0 mb-4">  
 <form method="post" class="form-group">  
 {% csrf\_token %}  
 {{form|crispy}}  
 <p>Don't have an account?<a href="/signup"> Create one.</a></p>  
 <button type="submit" class="btn btn-success">Login</button>  
  
 </form>  
 </div>  
{% endblock %}

**Registration/templates/registration/signup.html**

{% extends "registration/RegisterBase.html" %}  
{% block title %} Sign Up {% endblock %}  
  
{% load crispy\_forms\_tags %}  
 {% for msg in message %}  
 <script>  
 window.alert(msg)  
 </script>  
 {% endfor %}  
{% block content %}  
  
 <div class="sign-up form" style="max-width: 600px">  
 <h1 class="mt-2">{{heading}}</h1>  
 <hr class="mt-0 mb-4">  
  
 <form method="post" class="form-group" style="max-width: 600px;">  
 {% csrf\_token %}  
 {{form|crispy}}  
 <p>Already have an account?<a href="/login">Login here.</a></p>  
 <button type="submit" class="btn btn-success" >Sign Up</button>  
  
 </form>  
 </div>  
{% endblock %}

**Registration/forms.py**

from django import forms  
from django.contrib.auth.forms import UserCreationForm  
from django.contrib.auth.models import User  
from chat.models import UserProfile  
  
  
class SignUpForm(forms.Form):  
  
 username = forms.CharField(min\_length=5, max\_length=20)  
 name = forms.CharField(max\_length=25, label="Name")  
 email = forms.EmailField(label="Email")  
 password1 = forms.CharField(label='Password', widget=forms.PasswordInput)  
 password2 = forms.CharField(label="Confirm Password", widget=forms.PasswordInput)  
  
 def validate\_username(self):  
 username = self.cleaned\_data['username']  
 r = User.objects.filter(username=username)  
 if r.count():  
 return None  
 return username  
  
 def validate\_password(self):  
 password1 = self.cleaned\_data['password1']  
 password2 = self.cleaned\_data['password2']  
 if not password1 or not password2:  
 return None  
 elif password2 != password1:  
 return None  
 return password1  
  
 def validate\_email(self):  
 email = self.cleaned\_data['email']  
 r = UserProfile.objects.filter(email=email)  
 if r.count():  
 return None  
 return email  
  
 def save(self, commit=True):  
 user = User.objects.create\_user(  
 self.cleaned\_data['username'],  
 self.cleaned\_data['email'],  
 self.cleaned\_data['password1']  
 )  
 return user

**Registration/views.py**

from django.shortcuts import render, redirect  
from .forms import SignUpForm  
from django.contrib.auth import login, authenticate  
from chat.models import UserProfile  
  
  
def SignUp(request):  
 *"""  
 Sign up view* ***:param*** *request:* ***:return****:  
 """* message = []  
 if request.method == "POST":  
 form = SignUpForm(request.POST)  
 if form.is\_valid():  
 name = form.cleaned\_data.get('name')  
 email = form.validate\_email()  
 username = form.validate\_username()  
 password = form.validate\_password()  
 if not email:  
 message.append("Email already registered!")  
 elif not password:  
 message.append("Passwords don't match!")  
 elif not username:  
 message.append("Username already registered!")  
 else:  
 print("SUCCESS!!!!")  
 form.save()  
 user = authenticate(username=username, password=password)  
 login(request, user)  
 profile = UserProfile(email=email, name=name, username=username)  
 profile.save()  
 return redirect("/")  
 else:  
 form = SignUpForm()  
 return render(request, "registration/signup.html", {"form": form, "heading": "Sign Up", "message": message})

chat/views.py

from django.shortcuts import render, HttpResponse, redirect  
from .models import UserProfile, Friends, Messages  
from django.views.decorators.csrf import csrf\_exempt  
from django.http.response import JsonResponse  
from rest\_framework.parsers import JSONParser  
from chat.serializers import MessageSerializer  
  
  
def getFriendsList(id):  
 *"""  
 Get the list of friends of the user* ***:param****: user id* ***:return****: list of friends  
 """* try:  
 user = UserProfile.objects.get(id=id)  
 ids = list(user.friends\_set.all())  
 friends = []  
 for id in ids:  
 num = str(id)  
 fr = UserProfile.objects.get(id=int(num))  
 friends.append(fr)  
 return friends  
 except:  
 return []  
  
  
def getUserId(username):  
 *"""  
 Get the user id by the username* ***:param*** *username:* ***:return****: int  
 """* use = UserProfile.objects.get(username=username)  
 id = use.id  
 return id  
  
  
def index(request):  
 *"""  
 Return the home page* ***:param*** *request:* ***:return****:  
 """* if not request.user.is\_authenticated:  
 print("Not Logged In!")  
 return render(request, "chat/index.html", {})  
 else:  
 username = request.user.username  
 id = getUserId(username)  
 friends = getFriendsList(id)  
 return render(request, "chat/Base.html", {'friends': friends})  
  
  
def search(request):  
 *"""  
 Search users page* ***:param*** *request:* ***:return****:  
 """* users = list(UserProfile.objects.all())  
 for user in users:  
 if user.username == request.user.username:  
 users.remove(user)  
 break  
  
 if request.method == "POST":  
 print("SEARCHING!!")  
 query = request.POST.get("search")  
 user\_ls = []  
 for user in users:  
 if query in user.name or query in user.username:  
 user\_ls.append(user)  
 return render(request, "chat/search.html", {'users': user\_ls, })  
  
 try:  
 users = users[:10]  
 except:  
 users = users[:]  
 id = getUserId(request.user.username)  
 friends = getFriendsList(id)  
 return render(request, "chat/search.html", {'users': users, 'friends': friends})  
  
  
def addFriend(request, name):  
 *"""  
 Add a user to the friend's list  
 :param request:  
 :param name:  
 :return:  
 """* username = request.user.username  
 id = getUserId(username)  
 friend = UserProfile.objects.get(username=name)  
 curr\_user = UserProfile.objects.get(id=id)  
 print(curr\_user.name)  
 ls = curr\_user.friends\_set.all()  
 flag = 0  
 for username in ls:  
 if username.friend == friend.id:  
 flag = 1  
 break  
 if flag == 0:  
 print("Friend Added!!")  
 curr\_user.friends\_set.create(friend=friend.id)  
 friend.friends\_set.create(friend=id)  
 return redirect("/search")  
  
  
def chat(request, username):  
 *"""  
 Get the chat between two users.  
 :param request:  
 :param username:  
 :return:  
 """* friend = UserProfile.objects.get(username=username)  
 id = getUserId(request.user.username)  
 curr\_user = UserProfile.objects.get(id=id)  
 messages = Messages.objects.filter(sender\_name=id, receiver\_name=friend.id) | Messages.objects.filter(sender\_name=friend.id, receiver\_name=id)  
  
 if request.method == "GET":  
 friends = getFriendsList(id)  
 return render(request, "chat/messages.html",  
 {'messages': messages,  
 'friends': friends,  
 'curr\_user': curr\_user, 'friend': friend})  
  
  
@csrf\_exempt  
def message\_list(request, sender=None, receiver=None):  
 if request.method == 'GET':  
 messages = Messages.objects.filter(sender\_name=sender, receiver\_name=receiver, seen=False)  
 serializer = MessageSerializer(messages, many=True, context={'request': request})  
 for message in messages:  
 message.seen = True  
 message.save()  
 return JsonResponse(serializer.data, safe=False)  
  
 elif request.method == "POST":  
 data = JSONParser().parse(request)  
 serializer = MessageSerializer(data=data)  
 if serializer.is\_valid():  
 serializer.save()  
 return JsonResponse(serializer.data, status=201)  
 return JsonResponse(serializer.errors, status=400)

**chat/models.py:**

from django.db import models  
  
  
class UserProfile(models.Model):  
  
 name = models.CharField(max\_length=25)  
 email = models.EmailField(unique=True)  
 username = models.CharField(max\_length=20, unique=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.name}"  
  
  
class Messages(models.Model):  
  
 description = models.TextField()  
 sender\_name = models.ForeignKey(UserProfile, on\_delete=models.CASCADE, related\_name='sender')  
 receiver\_name = models.ForeignKey(UserProfile, on\_delete=models.CASCADE, related\_name='receiver')  
 time = models.TimeField(auto\_now\_add=True)  
 seen = models.BooleanField(default=False)  
 timestamp = models.DateTimeField(auto\_now\_add=True)  
  
 def \_\_str\_\_(self):  
 return f"To: {self.receiver\_name} From: {self.sender\_name}"  
  
 class Meta:  
 ordering = ('timestamp',)  
  
  
class Friends(models.Model):  
  
 user = models.ForeignKey(UserProfile, on\_delete=models.CASCADE)  
 friend = models.IntegerField()  
  
 def \_\_str\_\_(self):  
 return f"{self.friend}"

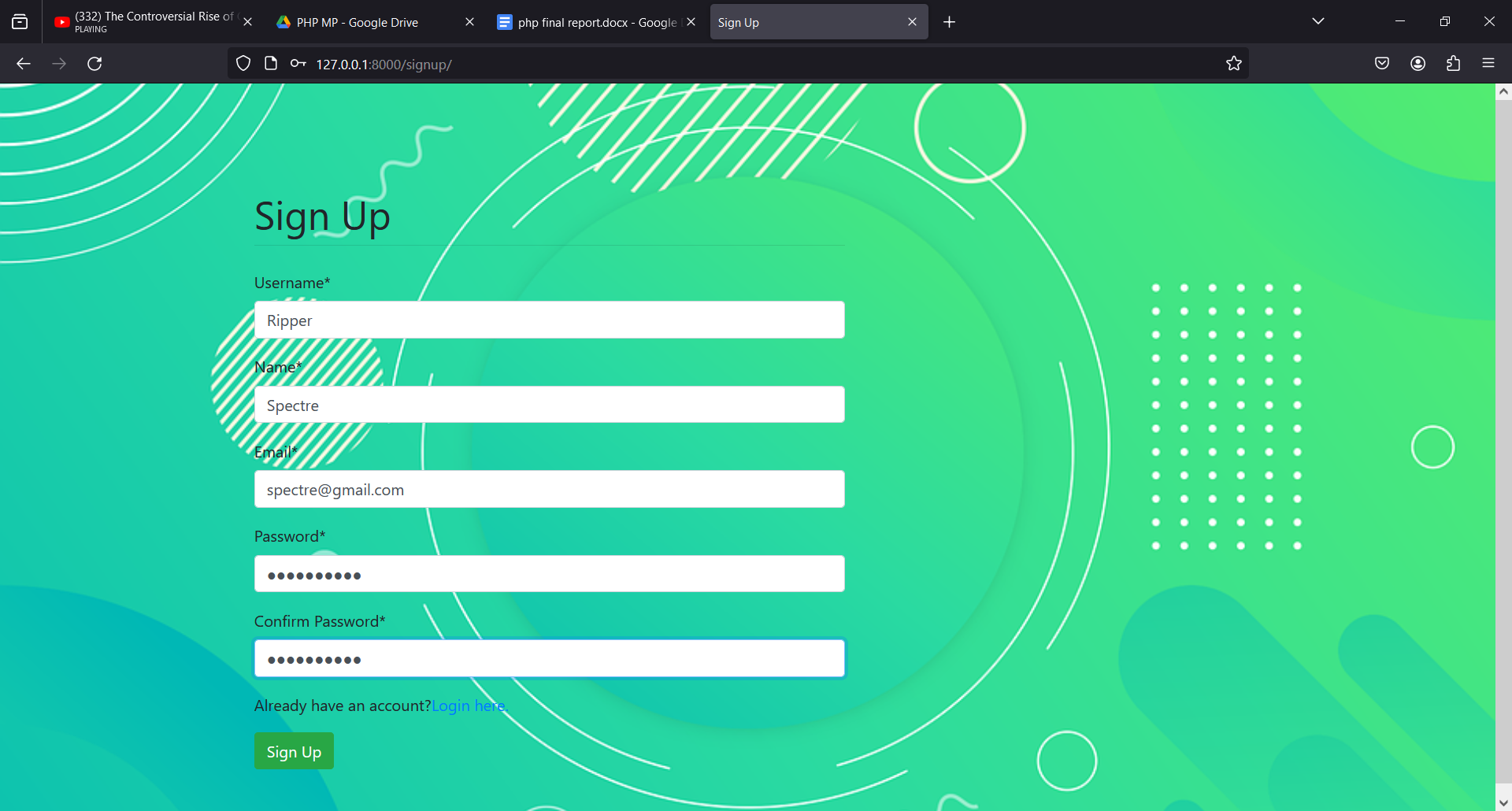
**static/js.chat.js**

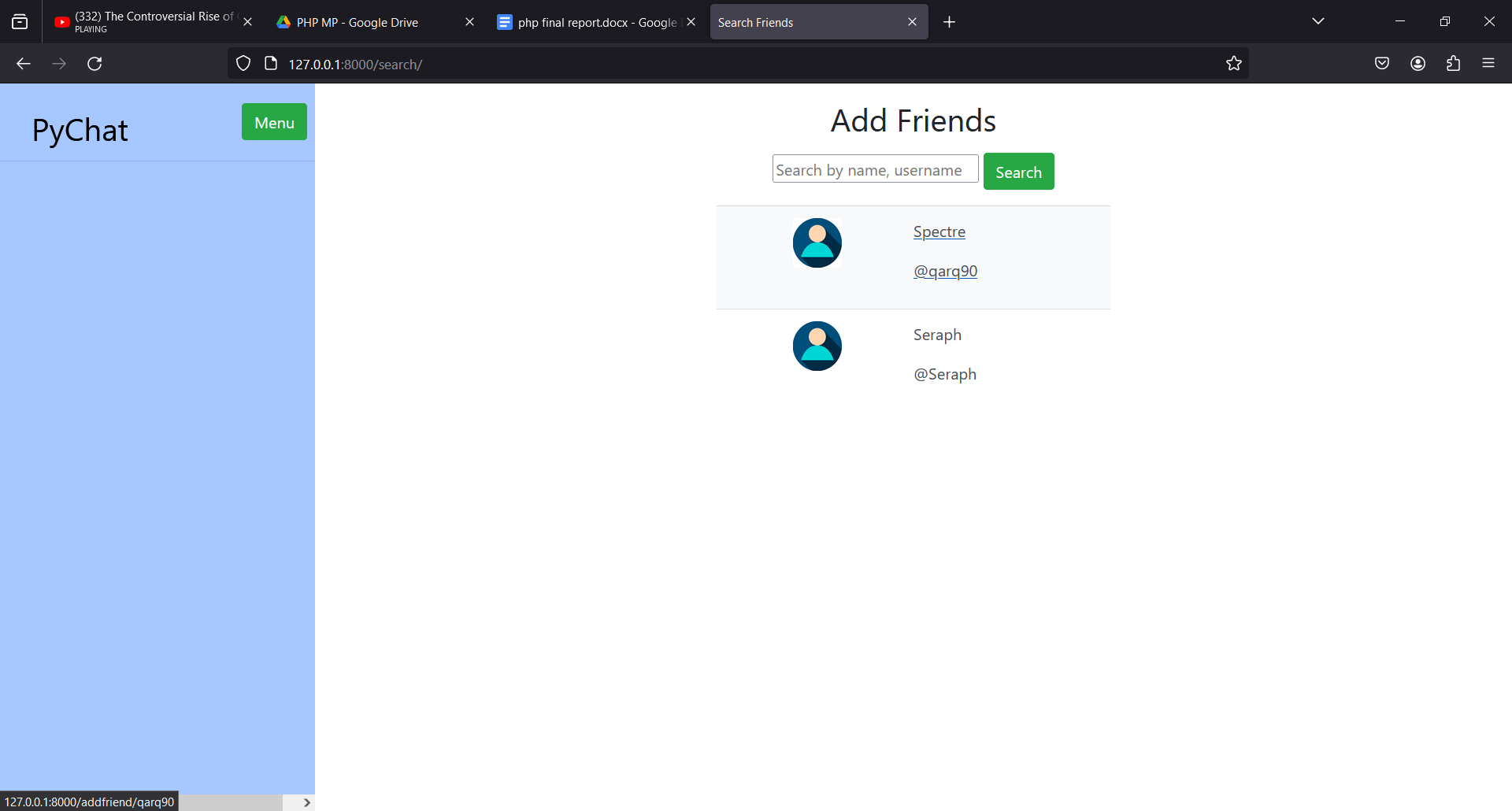
const ***text\_box*** = '<div class="container darker">' +  
 '<img src="{% static 'images/user\_image.jpg' %}" alt="Avatar" class="right" style="width:100%;">'  
 + '<p>{message}</p>' +  
 '<span class="time-right">{time}</span>'  
 + '</div>';  
  
function send(sender, receiver, message, time){  
 ***console***.log("YES! WORKING")  
 $.post('api/messages', '{"sender:" "' + sender + '", "receiver": "' + receiver + '", "message": "' +  
 message + '"}', function(data){  
 var field = ***text\_box***.replace('{message}', message);  
 field.replace('{time}', time)  
 $('#board').append(field);  
 })  
}  
  
function receive(){  
 $.get('api/messages/' + sender\_id + '/' + receiver\_id, function(data){  
 ***console***.log(data);  
 if(data.length() !== 0){  
 for(var i = 0; i < data.length; i++){  
 var field = ***text\_box***.replace('{message}', message);  
 $('#board').append(field);  
 }  
 }  
 })  
}

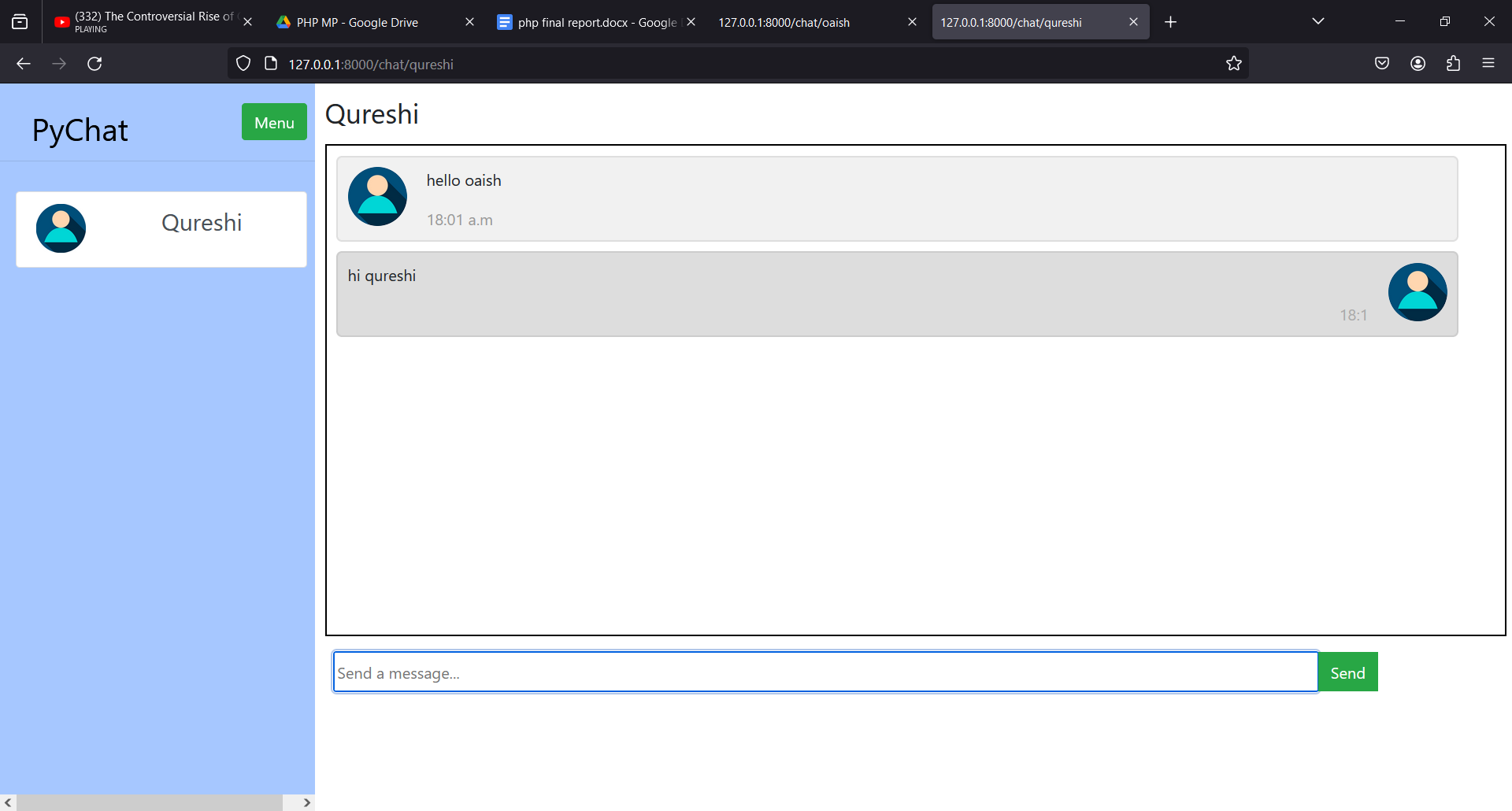
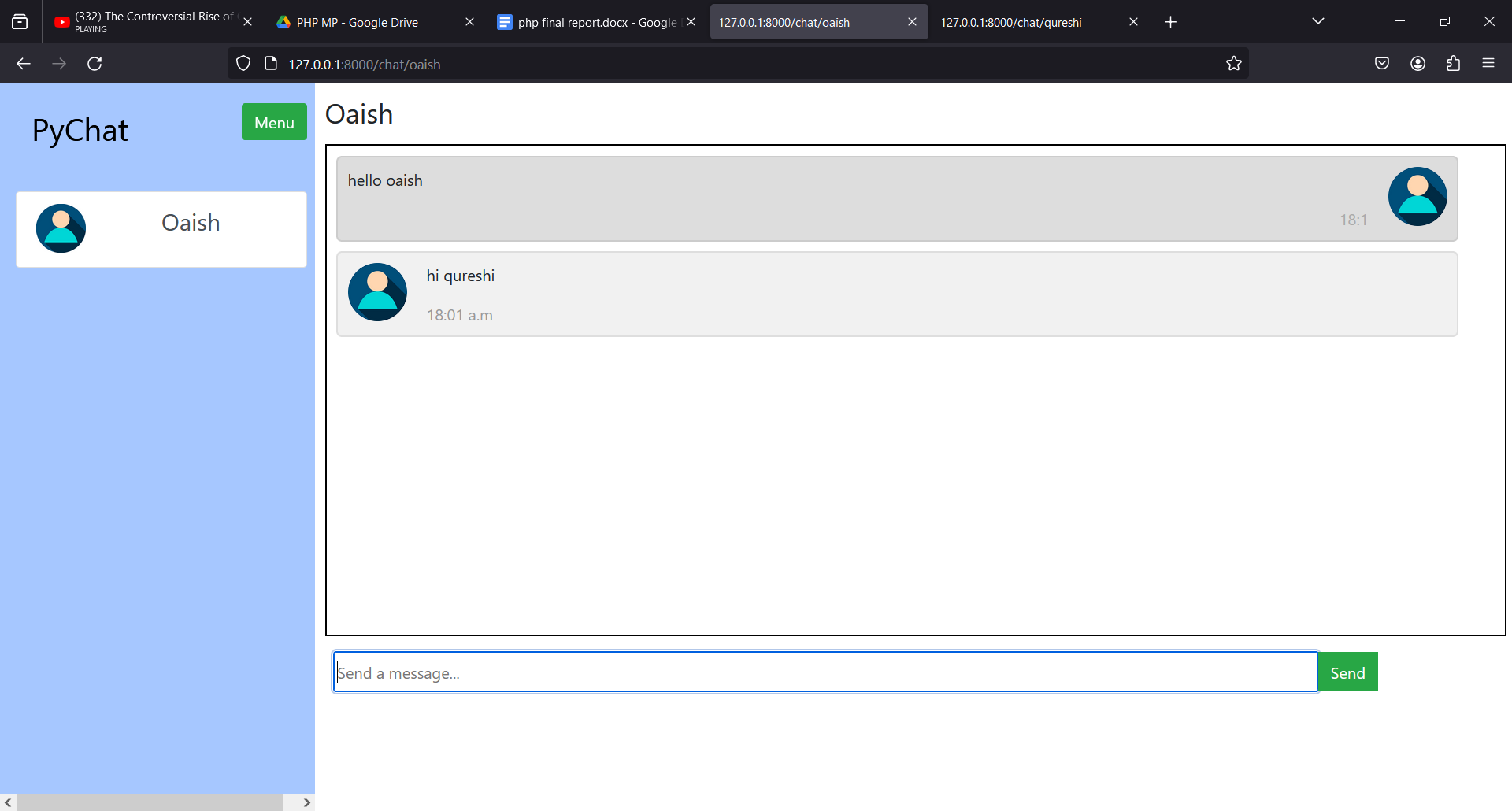
**rest\_framework/docs/js/api.js**

var ***responseDisplay*** = 'data'  
var ***coreapi*** = ***window***.***coreapi***var ***schema*** = ***window***.***schema***function normalizeKeys(arr) {  
 var \_normarr = [];  
 for (var i = 0; i < arr.length; i++) {  
 \_normarr = \_normarr.concat(arr[i].split(' > '));  
 }  
 return \_normarr;  
}  
  
function normalizeHTTPHeader(str) {  
 *// Capitalize HTTP headers for display.* return (str.charAt(0).toUpperCase() + str.substring(1))  
 .replace(/-(.)/g, function ($1) {  
 return $1.toUpperCase()  
 })  
 .replace(/(Www)/g, function ($1) {  
 return 'WWW'  
 })  
 .replace(/(Xss)/g, function ($1) {  
 return 'XSS'  
 })  
 .replace(/(Md5)/g, function ($1) {  
 return 'MD5'  
 })  
}  
  
function formEntries(form) {  
 *// Polyfill for new FormData(form).entries()* var formData = new FormData(form)  
 if (formData.entries !== undefined) {  
 return ***Array***.from(formData.entries())  
 }  
  
 var entries = []  
  
 for (var i = 0; i < form.elements.length; i++) {  
 var element = form.elements[i]  
  
 if (!element.name) {  
 continue  
 }  
  
 if (element.type === 'file') {  
 for (var j = 0; j < element.files.length; j++) {  
 entries.push([element.name, element.files[j]])  
 }  
 } else if (element.type === 'select-multiple' || element.type === 'select-one') {  
 for (var j = 0; j < element.selectedOptions.length; j++) {  
 entries.push([element.name, element.selectedOptions[j].value])  
 }  
 } else if (element.type === 'checkbox') {  
 if (element.checked) {  
 entries.push([element.name, element.value])  
 }  
 } else {  
 entries.push([element.name, element.value])  
 }  
 }  
  
 return entries  
}  
  
$(function () {  
 var $selectedAuthentication = $('#selected-authentication')  
 var $authControl = $('#auth-control')  
 var $authTokenModal = $('#auth\_token\_modal')  
 var $authBasicModal = $('#auth\_basic\_modal')  
 var $authSessionModal = $('#auth\_session\_modal')  
  
 *// Language Control* $('#language-control li').click(function (event) {  
 event.preventDefault()  
 var $languageMenuItem = $(this).find('a')  
 var $languageControls = $(this).closest('ul').find('li')  
 var $languageControlLinks = $languageControls.find('a')  
 var language = $languageMenuItem.data('language')  
  
 $languageControlLinks.not('[data-language="' + language + '"]').parent().removeClass('active')  
 $languageControlLinks.filter('[data-language="' + language + '"]').parent().addClass('active')  
  
 $('#selected-language').text(language)  
  
 var $codeBlocks = $('pre.highlight')  
 $codeBlocks.not('[data-language="' + language + '"]').addClass('hide')  
 $codeBlocks.filter('[data-language="' + language + '"]').removeClass('hide')  
 })  
  
 *// API Explorer* $('form.api-interaction').submit(function (event) {  
 event.preventDefault()  
  
 var $form = $(this).closest('form')  
 var $requestMethod = $form.find('.request-method')  
 var $requestUrl = $form.find('.request-url')  
 var $toggleView = $form.closest('.modal-content').find('.toggle-view')  
 var $responseStatusCode = $form.find('.response-status-code')  
 var $meta = $form.find('.meta')  
 var $responseRawResponse = $form.find('.response-raw-response')  
 var $requestAwaiting = $form.find('.request-awaiting')  
 var $responseRaw = $form.find('.response-raw')  
 var $responseData = $form.find('.response-data')  
 var key = normalizeKeys($form.data('key'))  
 var params = {}  
 var entries = formEntries($form.get()[0])  
  
 for (var i = 0; i < entries.length; i++) {  
 var entry = entries[i]  
 var paramKey = entry[0]  
 var paramValue = entry[1]  
 var $elem = $form.find('[name="' + paramKey + '"]')  
 var dataType = $elem.data('type') || 'string'  
  
 if (dataType === 'integer' && paramValue) {  
 var value = parseInt(paramValue)  
 if (!isNaN(value)) {  
 params[paramKey] = value  
 }  
 } else if (dataType === 'number' && paramValue) {  
 var value = parseFloat(paramValue)  
 if (!isNaN(value)) {  
 params[paramKey] = value  
 }  
 } else if (dataType === 'boolean' && paramValue) {  
 var value = {  
 'true': true,  
 'false': false  
 }[paramValue.toLowerCase()]  
 if (value !== undefined) {  
 params[paramKey] = value  
 }  
 } else if (dataType === 'array' && paramValue) {  
 try {  
 params[paramKey] = ***JSON***.parse(paramValue)  
 } catch (err) {  
 *// Ignore malformed JSON* }  
 } else if (dataType === 'object' && paramValue) {  
 try {  
 params[paramKey] = ***JSON***.parse(paramValue)  
 } catch (err) {  
 *// Ignore malformed JSON* }  
 } else if (dataType === 'string' && paramValue) {  
 params[paramKey] = paramValue  
 }  
 }  
  
 $form.find(':checkbox').each(function (index) {  
 *// Handle unselected checkboxes* var name = $(this).attr('name')  
 if (!params.hasOwnProperty(name)) {  
 params[name] = false  
 }  
 })  
  
 function requestCallback(request) {  
 *// Fill in the "GET /foo/" display.* var parser = ***document***.createElement('a')  
 parser.href = request.url  
 var method = request.options.method  
 var path = parser.pathname + parser.hash + parser.search  
  
 $requestMethod.text(method)  
 $requestUrl.text(path)  
 }  
  
 function responseCallback(response, responseText) {  
 *// Display the 'Data'/'Raw' control.* $toggleView.removeClass('hide')  
  
 *// Fill in the "200 OK" display.* $responseStatusCode.removeClass('label-success').removeClass('label-danger')  
 if (response.ok) {  
 $responseStatusCode.addClass('label-success')  
 } else {  
 $responseStatusCode.addClass('label-danger')  
 }  
 $responseStatusCode.text(response.status)  
 $meta.removeClass('hide')  
  
 *// Fill in the Raw HTTP response display.* var panelText = 'HTTP/1.1 ' + response.status + ' ' + response.statusText + '\n'  
 response.headers.forEach(function (header, key) {  
 panelText += normalizeHTTPHeader(key) + ': ' + header + '\n'  
 })  
 if (responseText) {  
 panelText += '\n' + responseText  
 }  
 $responseRawResponse.text(panelText)  
 }  
  
 *// Instantiate a client to make the outgoing request.* var options = {  
 requestCallback: requestCallback,  
 responseCallback: responseCallback  
 }  
  
 *// Setup authentication options.* if (***window***.auth && ***window***.auth.type === 'token') {  
 *// Header authentication* options.auth = new ***coreapi***.auth.TokenAuthentication({  
 scheme: ***window***.auth.scheme,  
 token: ***window***.auth.token  
 })  
 } else if (***window***.auth && ***window***.auth.type === 'basic') {  
 *// Basic authentication* options.auth = new ***coreapi***.auth.BasicAuthentication({  
 username: ***window***.auth.username,  
 password: ***window***.auth.password  
 })  
 } else if (***window***.auth && ***window***.auth.type === 'session') {  
 *// Session authentication* options.auth = new ***coreapi***.auth.SessionAuthentication({  
 csrfCookieName: 'csrftoken',  
 csrfHeaderName: 'X-CSRFToken'  
 })  
 }  
  
 var client = new ***coreapi***.Client(options)  
 client.action(***schema***, key, params).then(function (data) {  
 var response = ***JSON***.stringify(data, null, 2)  
 $requestAwaiting.addClass('hide')  
 $responseRaw.addClass('hide')  
 $responseData.addClass('hide').text('').jsonView(response)  
  
 if (***responseDisplay*** === 'data') {  
 $responseData.removeClass('hide')  
 } else {  
 $responseRaw.removeClass('hide')  
 }  
 }).catch(function (error) {  
 var response = ***JSON***.stringify(error.content, null, 2)  
 $requestAwaiting.addClass('hide')  
 $responseRaw.addClass('hide')  
 $responseData.addClass('hide').text('').jsonView(response)  
  
 if (***responseDisplay*** === 'data') {  
 $responseData.removeClass('hide')  
 } else {  
 $responseRaw.removeClass('hide')  
 }  
 })  
 })  
  
 *// 'Data'/'Raw' control* $('.toggle-view button').click(function () {  
 var $modalContent = $(this).closest('.modal-content')  
 var $modalResponseRaw = $modalContent.find('.response-raw')  
 var $modalResponseData = $modalContent.find('.response-data')  
  
 ***responseDisplay*** = $(this).data('display-toggle')  
  
 $(this).removeClass('btn-default').addClass('btn-info').siblings().removeClass('btn-info')  
  
 if (***responseDisplay*** === 'raw') {  
 $modalResponseRaw.removeClass('hide')  
 $modalResponseData.addClass('hide')  
 } else {  
 $modalResponseData.removeClass('hide')  
 $modalResponseRaw.addClass('hide')  
 }  
 })  
  
 *// Authentication: none* $authControl.find("[data-auth='none']").click(function (event) {  
 event.preventDefault()  
 ***window***.auth = null  
 $selectedAuthentication.text('none')  
 $authControl.find("[data-auth]").closest('li').removeClass('active')  
 $authControl.find("[data-auth='none']").closest('li').addClass('active')  
 })  
  
 *// Authentication: token* $('form.authentication-token-form').submit(function (event) {  
 event.preventDefault()  
 var $form = $(this).closest('form')  
 var scheme = $form.find('input#scheme').val()  
 var token = $form.find('input#token').val()  
 ***window***.auth = {  
 'type': 'token',  
 'scheme': scheme,  
 'token': token  
 }  
 $selectedAuthentication.text('token')  
 $authControl.find("[data-auth]").closest('li').removeClass('active')  
 $authControl.find("[data-auth='token']").closest('li').addClass('active')  
 $authTokenModal.modal('hide')  
 })  
  
 *// Authentication: basic* $('form.authentication-basic-form').submit(function (event) {  
 event.preventDefault()  
 var $form = $(this).closest('form')  
 var username = $form.find('input#username').val()  
 var password = $form.find('input#password').val()  
 ***window***.auth = {  
 'type': 'basic',  
 'username': username,  
 'password': password  
 }  
 $selectedAuthentication.text('basic')  
 $authControl.find("[data-auth]").closest('li').removeClass('active')  
 $authControl.find("[data-auth='basic']").closest('li').addClass('active')  
 $authBasicModal.modal('hide')  
 })  
  
 *// Authentication: session* $('form.authentication-session-form').submit(function (event) {  
 event.preventDefault()  
 ***window***.auth = {  
 'type': 'session'  
 }  
 $selectedAuthentication.text('session')  
 $authControl.find("[data-auth]").closest('li').removeClass('active')  
 $authControl.find("[data-auth='session']").closest('li').addClass('active')  
 $authSessionModal.modal('hide')  
 })  
})

**Output:**





**8.0 Skills Developed** / **Learning outcome of this Micro-Project**

The following skills are developed:

1) **Identifying:** Identifying the problem and cause of problem in the area related and prepare project proposals before starting the project.

2) **Designing:** Designing of micro project with minimum required resources (low cost).

3) **Teamwork:** Learn to work in a team and boost individual confidence.

4) **Time Management:** Timely completion of micro project as scheduled.

5) **Problem-solving:** Develop good problem-solving skills.

6) **Technical Writing:** Preparing a report of the proposed plan and final report.

7) **Confidence:** Confidently, answer the questions asked about the project.

8) **Acknowledgement:** Acknowledge the help rendered by others in the success of the project.

**9.0 Applications of this Micro-Project**

* Educational Tool
* Teaching Resource
* Personal Portfolio
* Intranet Communication

**(To be evaluated by the concerned teacher)**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Annexure IV**

**Micro Project Evaluation Sheet**

**Name of Student: Abdurrahman Qureshi Enrollment No.: 2100020112**

**Name of Programme: Computer Engineering Semester: SIXTH**

**Course Title: Programming with python (PWP) Code: 22616**

**Title of the Micro-Project: Chat Application**

**Course Outcomes Achieved: -**

* Develop python program to demonstrate use of Operators
* Perform operations on data structures in Python.
* Develop functions for given problem.
* Design classes for given problem.
* Handle exceptions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr No.** | **Characteristics to be assessed** | **Poor**  **(Marks 1 - 3)** | **Average**  **(Marks 4 - 5)** | **Good**  **(Marks 6 - 8)** | **Excellent**  **(Marks 9- 10)** | **Sub Total** |
| (A) Process and Product Assessment (Convert above total marks out of 6 Marks) | | | | | | |
| 1 | Relevance to the course |  |  |  |  |  |
| 2 | Literature Review/information collection |  |  |  |  |
| 3 | Completion of the Target as per project proposal |  |  |  |  |
| 4 | Analysis of Data and representation |  |  |  |  |
| 5 | Quality of Prototype/Model |  |  |  |  |
| 6 | Report Preparation |  |  |  |  |
| (B) Individual Presentation/ Viva (Convert above total marks out of 4 Marks) | | | | | | |
| 7 | Presentation |  |  |  |  |  |
| 8 | Viva |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **(A)**  **Process and Product Assessment**  **(6 marks)** | **(B)**  **Individual Presentation & viva**  **(4 marks)** | **Total Marks**  **10** |
|  |  |  |

**Comments/Suggestions about teamwork/leadership/inter-personal communication (if any)**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**Name and designation of the Teacher Prof. Zaibunnisa Malik Ma’am**

**Dated Signature .................................................................**

**Micro Project Evaluation Sheet**

**Name of Student: Qazi Mohd Oaish Enrollment No.: 2100020108**

**Name of Programme: Computer Engineering Semester: SIXTH**

**Course Title: Programming with python (PWP) Code: 22616**

**Title of the Micro-Project: Chat Application**

**Course Outcomes Achieved: -**

* Develop python program to demonstrate use of Operators
* Perform operations on data structures in Python.
* Develop functions for given problem.
* Design classes for given problem.
* Handle exceptions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr No.** | **Characteristics to be assessed** | **Poor**  **(Marks 1 - 3)** | **Average**  **(Marks 4 - 5)** | **Good**  **(Marks 6 - 8)** | **Excellent**  **(Marks 9- 10)** | **Sub Total** |
| (A) Process and Product Assessment (Convert above total marks out of 6 Marks) | | | | | | |
| 1 | Relevance to the course |  |  |  |  |  |
| 2 | Literature Review/information collection |  |  |  |  |
| 3 | Completion of the Target as per project proposal |  |  |  |  |
| 4 | Analysis of Data and representation |  |  |  |  |
| 5 | Quality of Prototype/Model |  |  |  |  |
| 6 | Report Preparation |  |  |  |  |
| (B) Individual Presentation/ Viva (Convert above total marks out of 4 Marks) | | | | | | |
| 7 | Presentation |  |  |  |  |  |
| 8 | Viva |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **(A)**  **Process and Product Assessment**  **(6 marks)** | **(B)**  **Individual Presentation & viva**  **(4 marks)** | **Total Marks**  **10** |
|  |  |  |

**Comments/Suggestions about teamwork/leadership/inter-personal communication (if any)**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**Name and designation of the Teacher Prof. Zaibunnisa Malik Ma’am**

**Dated Signature .................................................................**

**Micro Project Evaluation Sheet**

**Name of Student: Shaikh Mohammed Hussain Enrollment No.: 2200020625**

**Name of Programme: Computer Engineering Semester: SIXTH**

**Course Title: Programming with python (PWP) Code: 22616**

**Title of the Micro-Project: Chat Application**

**Course Outcomes Achieved: -**

* Develop python program to demonstrate use of Operators
* Perform operations on data structures in Python.
* Develop functions for given problem.
* Design classes for given problem.
* Handle exceptions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr No.** | **Characteristics to be assessed** | **Poor**  **(Marks 1 - 3)** | **Average**  **(Marks 4 - 5)** | **Good**  **(Marks 6 - 8)** | **Excellent**  **(Marks 9- 10)** | **Sub Total** |
| (A) Process and Product Assessment (Convert above total marks out of 6 Marks) | | | | | | |
| 1 | Relevance to the course |  |  |  |  |  |
| 2 | Literature Review/information collection |  |  |  |  |
| 3 | Completion of the Target as per project proposal |  |  |  |  |
| 4 | Analysis of Data and representation |  |  |  |  |
| 5 | Quality of Prototype/Model |  |  |  |  |
| 6 | Report Preparation |  |  |  |  |
| (B) Individual Presentation/ Viva (Convert above total marks out of 4 Marks) | | | | | | |
| 7 | Presentation |  |  |  |  |  |
| 8 | Viva |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **(A)**  **Process and Product Assessment**  **(6 marks)** | **(B)**  **Individual Presentation & viva**  **(4 marks)** | **Total Marks**  **10** |
|  |  |  |

**Comments/Suggestions about teamwork/leadership/inter-personal communication (if any)**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**............................................................................................................................................................................**

**Name and designation of the Teacher Prof. Zaibunnisa Malik Ma’am**

**Dated Signature .................................................................**