INNOMATICS®
RESEARCH LABS

INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

Regex Matching Web App Development Project

Prepared By
Lingerkar Rithikha

About me

Currently pursuing a Master of Science in Data Science, my educational background is rooted in a strong foundation of analytical skills. My passion lies in unraveling the hidden patterns within datasets and extracting valuable insights. Through a previous internship in the insurance sector, I gained practical experience in data analysis, honing my ability to derive meaningful conclusions from complex datasets. My motivation to learn data science stems from a deep-seated curiosity and a desire to contribute to the field by leveraging data-driven approaches to solve real-world problems.

www.linkedin.com/in/rithikha-lingerkar-82b18128a

https://github.com/LingerkarRithikha/Regex-Matcher-App

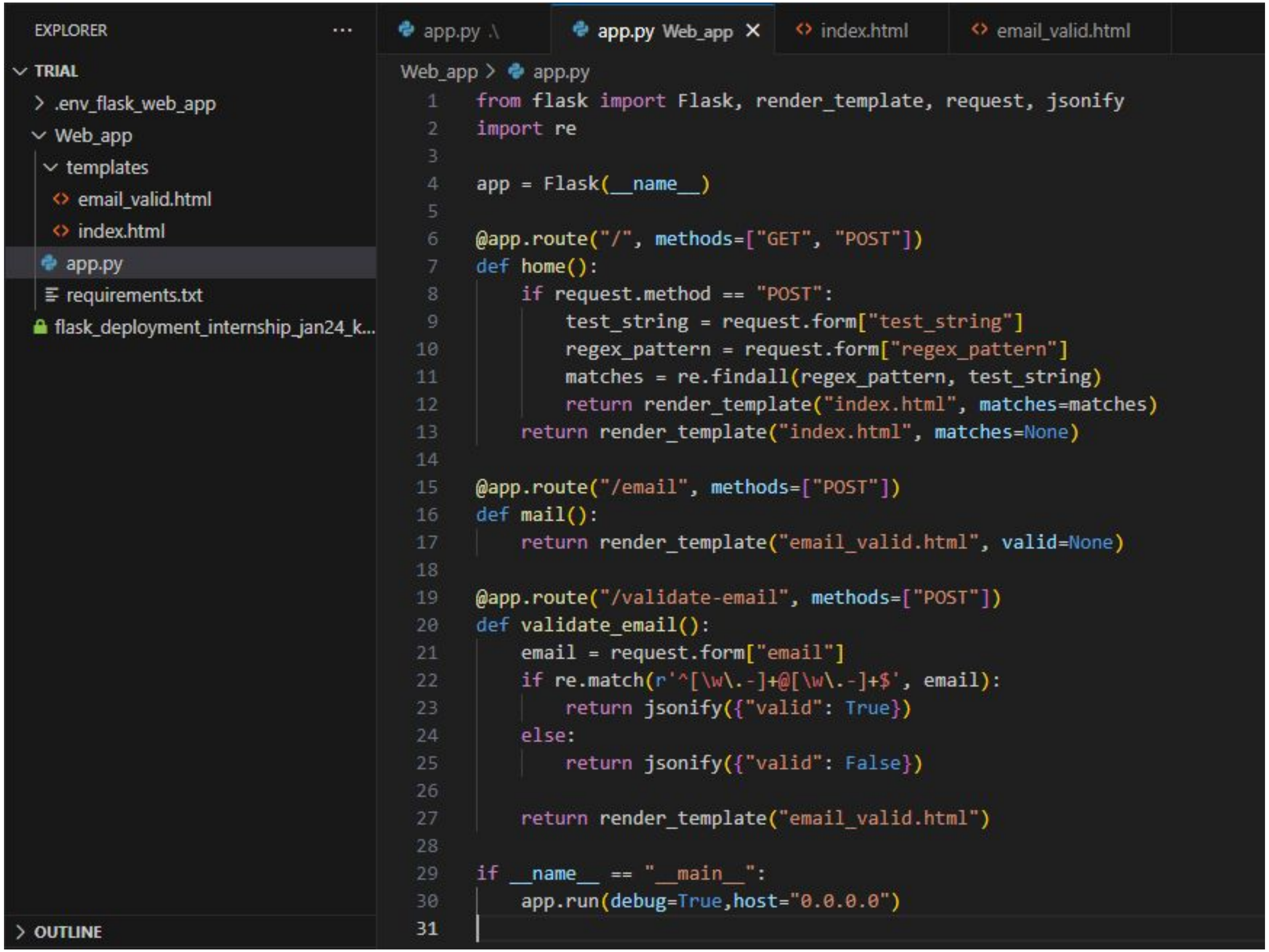
INNOMATICS RESEARCH LABS

Objective:

Our task is to replicate the core functionality of the website regex101.com. This entails creating a web application that allows users to input a test string and a regular expression (regex) and displays all the matches found.

Steps:

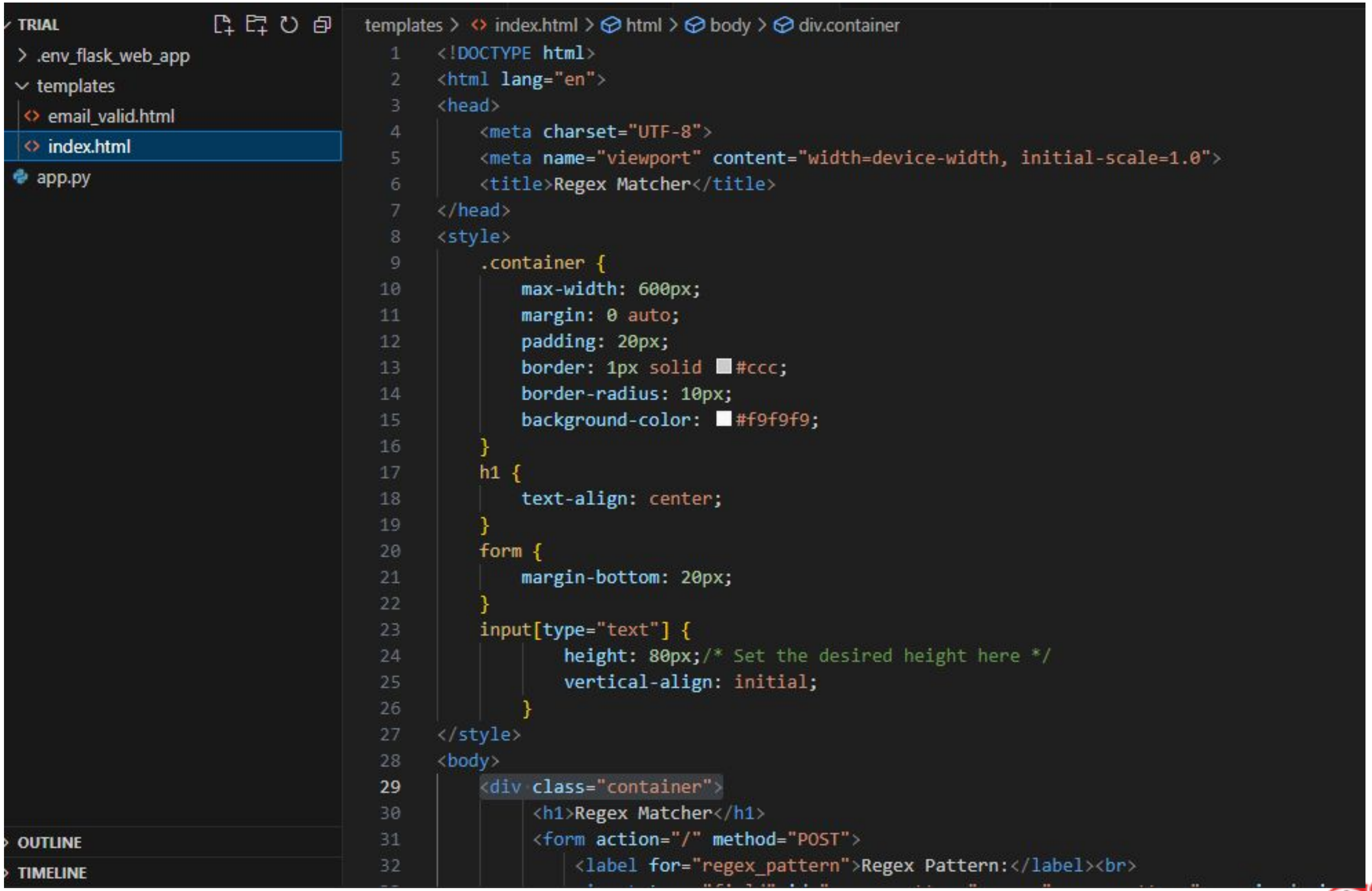
1. Create a new directory for your project and navigate into it.
2. Set up your virtual development environment:- Install Flask, a Python web framework, using pip if not already installed: `pip install Flask`.
3. Initialize a new Flask application:
  - Create a new Python file named `app.py`.
  - Import Flask and create a new Flask app instance.
 - Define a route for the home page ("/") where users can input the test string and regex.
 - Render an HTML template containing a form with fields for the test string and regex, and a submit button.



```python
from flask import Flask, render_template, request, jsonify
import re

app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])
def home():
    if request.method == "POST":
        test_string = request.form["test_string"]
        regex_pattern = request.form["regex_pattern"]
        matches = re.findall(regex_pattern, test_string)
        return render_template("index.html", matches=matches)
    return render_template("index.html", matches=None)

@app.route("/email", methods=["POST"])
def mail():
    return render_template("email_valid.html", valid=None)

@app.route("/validate-email", methods=["POST"])
def validate_email():
    email = request.form["email"]
    if re.match(r'^[\w\.-]+@[\w\.-]+$', email):
        return jsonify({"valid": True})
    else:
        return jsonify({"valid": False})

    return render_template("email_valid.html")

if __name__ == "__main__":
    app.run(debug=True,host="0.0.0.0")
```

Code for app.py

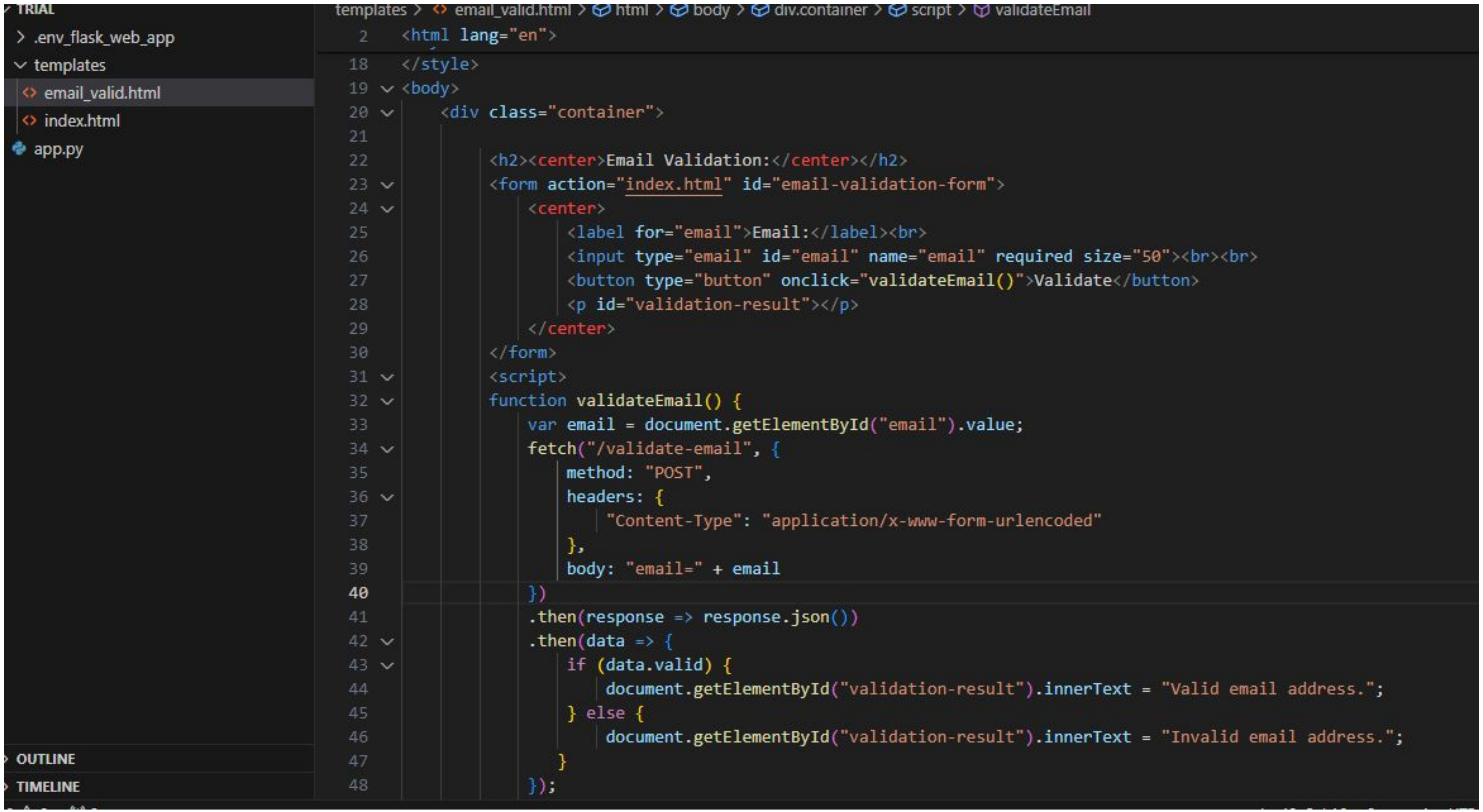
4. Create the HTML template:
   - Create a new directory named `templates` within your project directory.
   - Inside the `templates` directory, create a new HTML file named `index.html`.
   - Design the HTML form with input fields for the test string and regex, and a submit button.
5. Define a route to handle form submission:
   - Define a new route ("/results") in your `app.py` file to handle form submission.
   - Extract the test string and regex submitted by the user from the form data.
   - Use Python `re` module to perform regex matching on the test string.
   - Store the matched strings in a list.
6. Render the results:
   - Pass the list of matched strings to the HTML template.
   - Modify the HTML template to display the matched strings below the input form.
7. Test your application:
   - Run your Flask application (`python app.py`).
   - Open a web browser and navigate to http://localhost:5000 to access your application.
   - Input various test strings and regex patterns to ensure the application displays the correct matches.
8. Implement a new route where a user can validate if a given email id is valid or not.



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Regex Matcher</title>
</head>
<style>
    .container {
        max-width: 600px;
        margin: 0 auto;
        padding: 20px;
        border: 1px solid #ccc;
        border-radius: 10px;
        background-color: #f9f9f9;
    }
    h1 {
        text-align: center;
    }
    form {
        margin-bottom: 20px;
    }
    input[type="text"] {
            height: 80px;/* Set the desired height here */
            vertical-align: initial;
        }
</style>
<body>
    <div class="container">
        <h1>Regex Matcher</h1>
        <form action="/" method="POST">
            <label for="regex_pattern">Regex Pattern:</label><br>
```

Code of Index.html

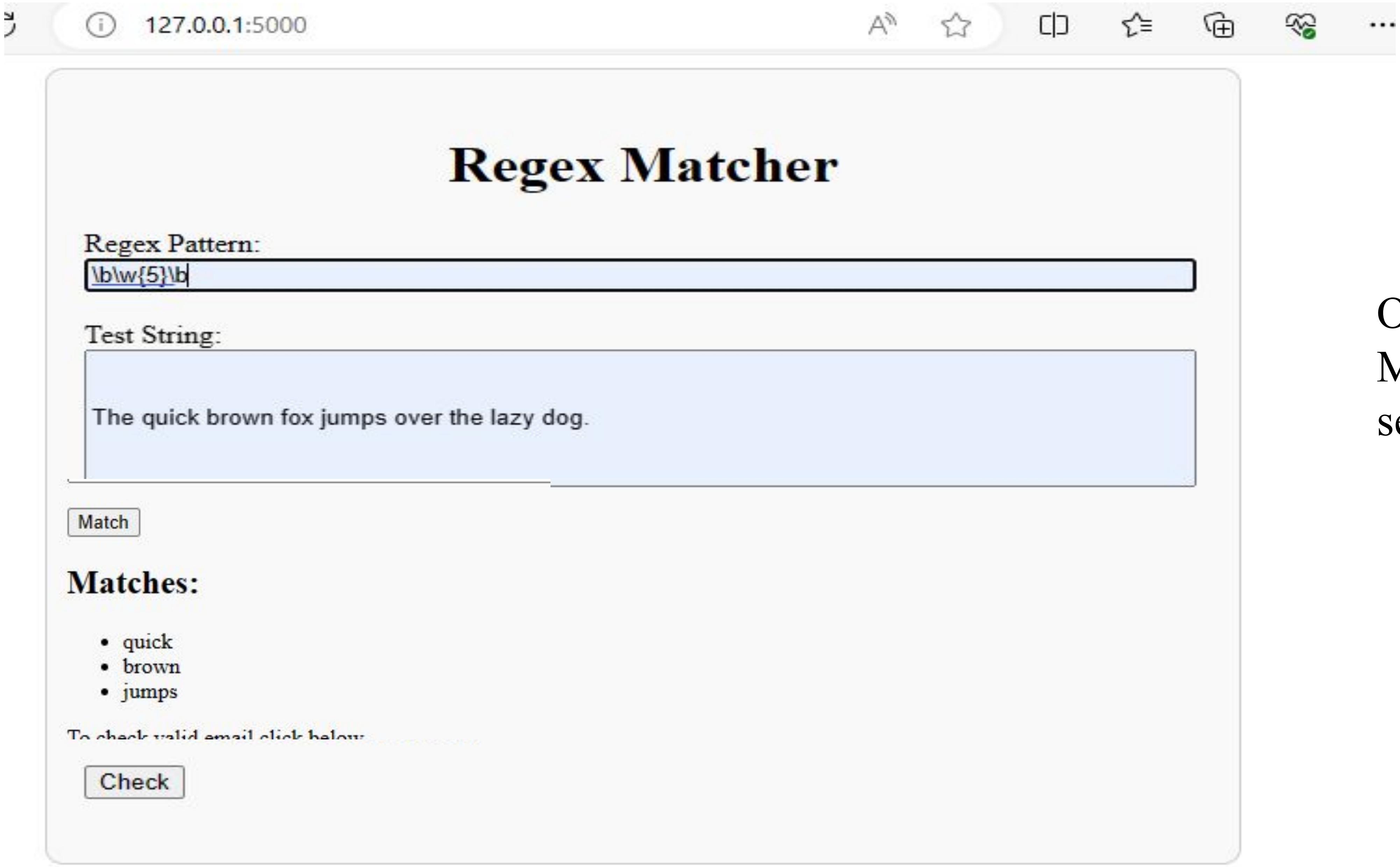
Code for Email validation (HTML)



TRIAL
.env_flask_web_app
templates
email_valid.html
index.html
app.py
OUTLINE
TIMELINE
templates > email_valid.html > html > body > div.container > script > validateEmail

```
<html lang="en">
</style>
<body>
    <div class="container">

        <h2><center>Email Validation:</center></h2>
        <form action="index.html" id="email-validation-form">
            <center>
                <label for="email">Email:</label><br>
                <input type="email" id="email" name="email" required size="50"><br><br>
                <button type="button" onclick="validateEmail()">Validate</button>
                <p id="validation-result"></p>
            </center>
        </form>
        <script>
        function validateEmail() {
            var email = document.getElementById("email").value;
            fetch("/validate-email", {
                method: "POST",
                headers: {
                    "Content-Type": "application/x-www-form-urlencoded"
                },
                body: "email=" + email
            })
            .then(response => response.json())
            .then(data => {
                if (data.valid) {
                    document.getElementById("validation-result").innerText = "Valid email address.";
                } else {
                    document.getElementById("validation-result").innerText = "Invalid email address.";
                }
            });
```

INNOMATICS RESEARCH LABS


127.0.0.1:5000

Regex Matcher

Regex Pattern:

\b\w{5}\b

Test String:

The quick brown fox jumps over the lazy dog.

Match

Matches:

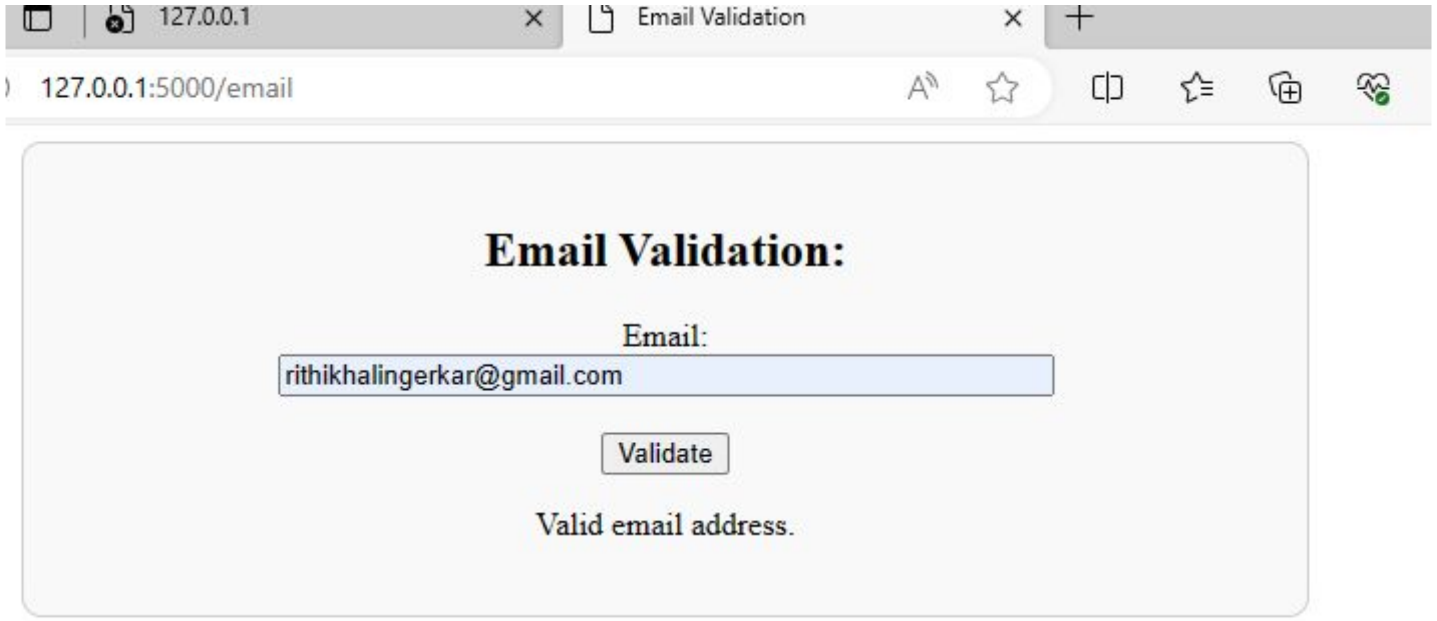
- quick
- brown
- jumps

To check valid email click below

Check

Output of Regex Matcher (on local server)

INNOMATICS RESEARCH LABS



Email Validation:

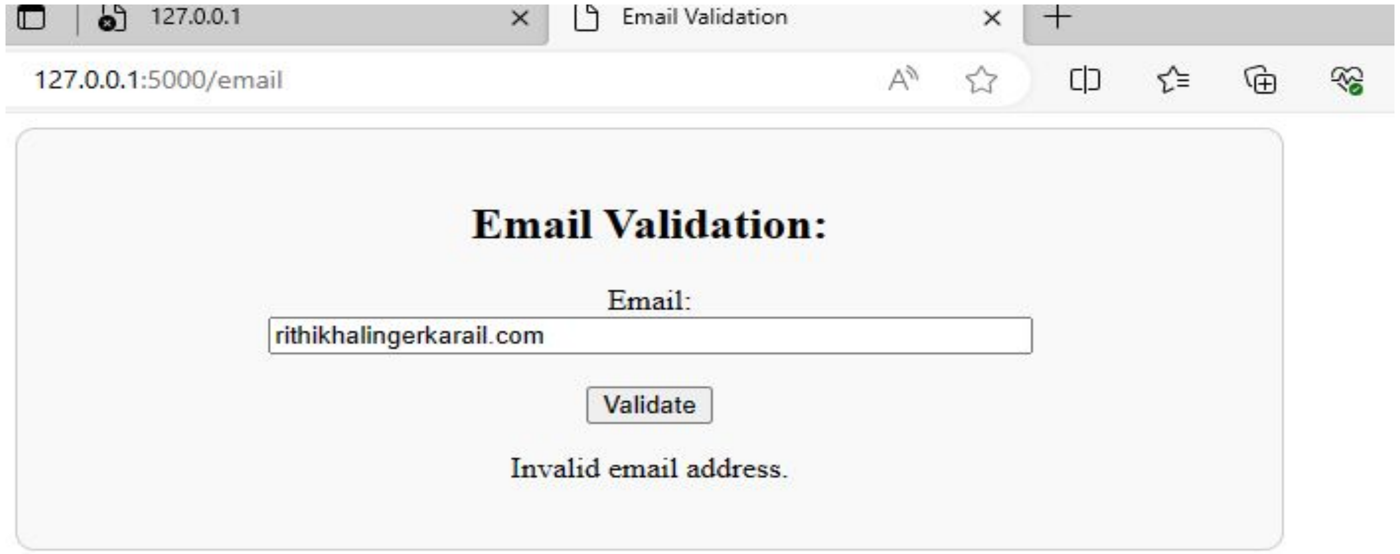
Email:

rithikhalingerkar@gmail.com

Validate

Valid email address.

Output of Email Validation (on local server)



Email Validation:

Email:

rithikhalingerkarail.com

Validate

Invalid email address.

9. Deploy the application on AWS Cloud.

To deploy the Regex Application on AWS cloud i have followed the following notes where we have created an EC2 Instance and then hosted our web app on AWS.

Notes : https://aws-deployment-tutorial-for-flask-app.streamlit.app/

URL : http://16.171.149.170:5000/

INNOMATICS RESEARCH LABS

THANK YOU

INNOMATICS RESEARCH LABS