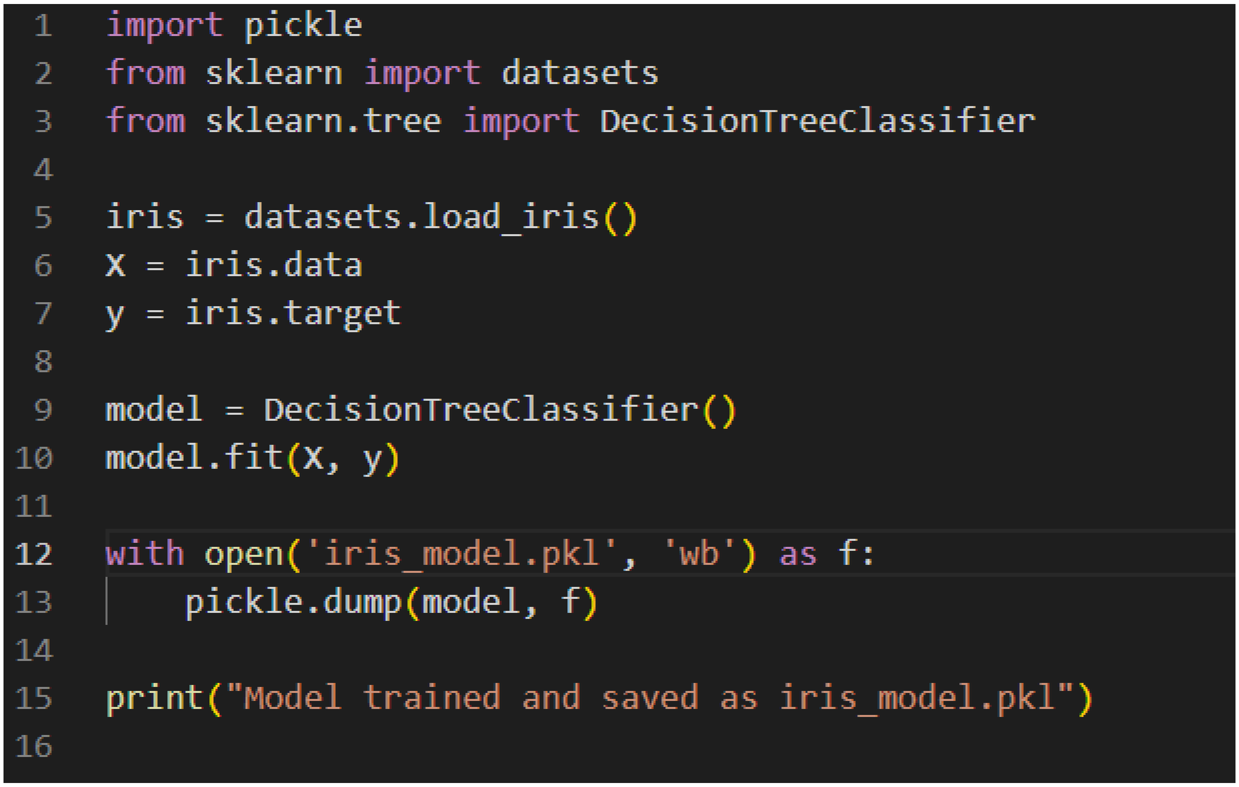
Week5: Cloud and API deployment

Name: Cloud and API deployment
Report date: 11/4/2024
Internship Batch: LISUM38
Version: 1.0
Data intake by: Serhat Uğur
Data intake reviewer: Data Glacier
Data storage location: https://github.com/serhatugur/data-science-internship

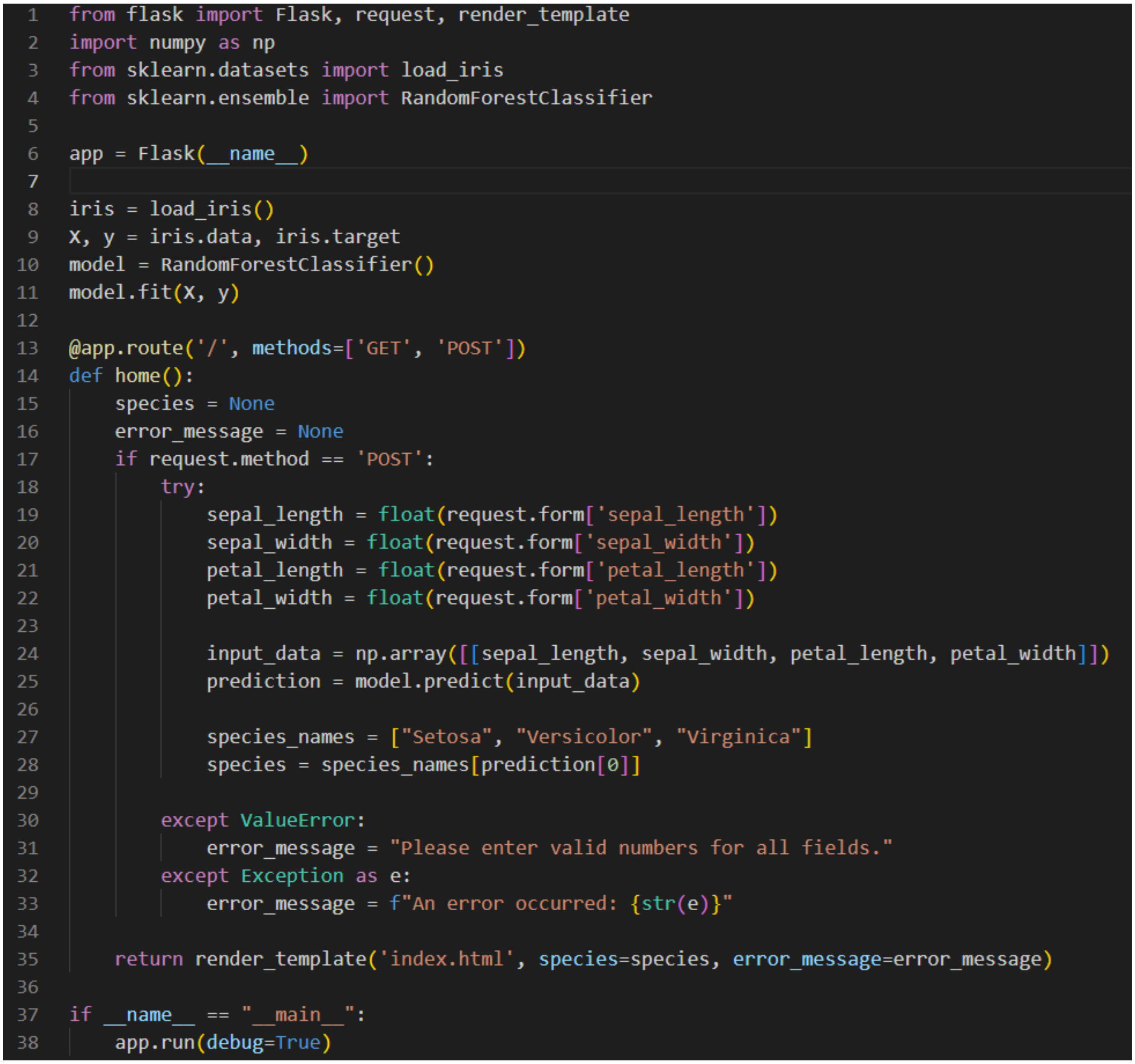
Proposed Approach: I used and trained the Iris dataset for my project. I will demonstrate the process from the beginning.

1) Model Building



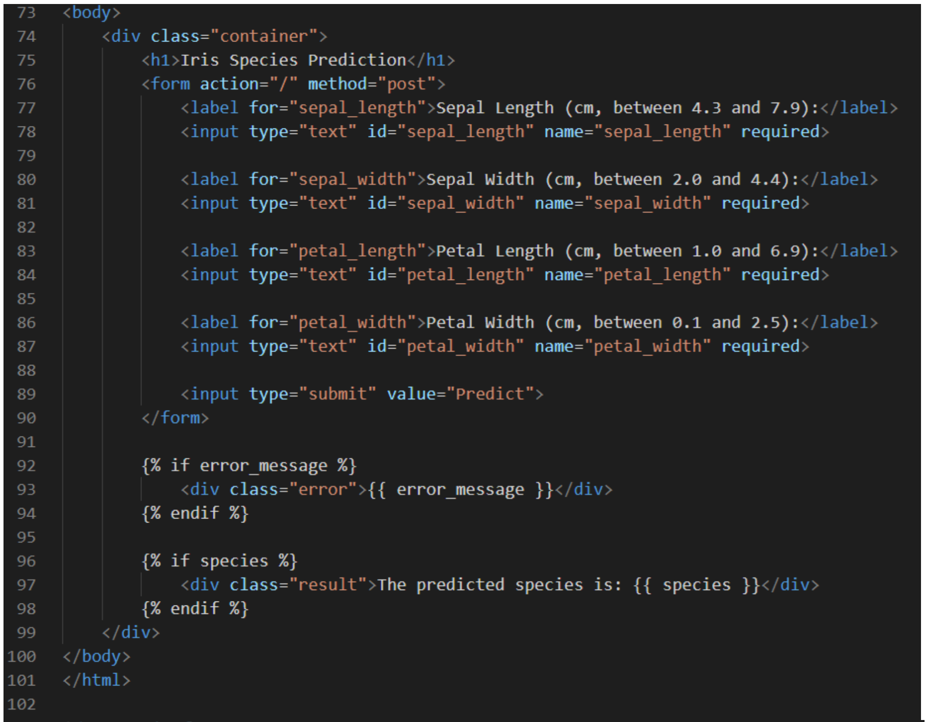
```python
import pickle
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier

iris = datasets.load_iris()
X = iris.data
y = iris.target

model = DecisionTreeClassifier()
model.fit(X, y)

with open('iris_model.pkl', 'wb') as f:
    pickle.dump(model, f)

print("Model trained and saved as iris_model.pkl")

```

2) Using Flask to illustrate the model on the Web

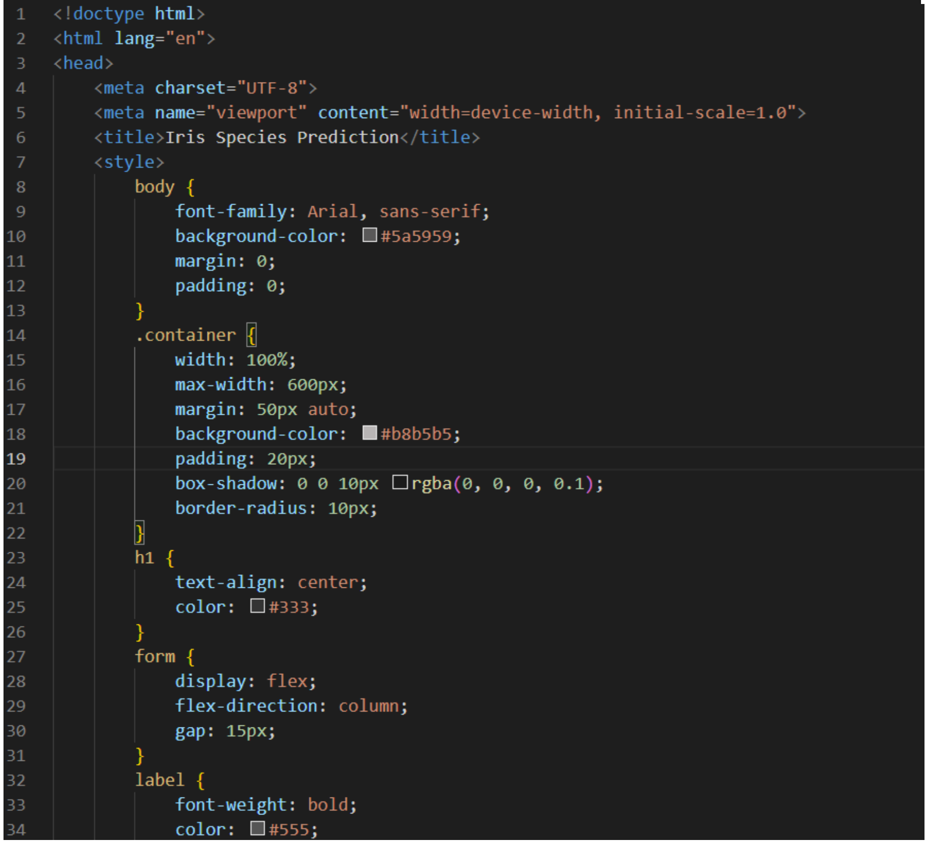


```python
from flask import Flask, request, render_template
import numpy as np
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier

app = Flask(__name__)

iris = load_iris()
X, y = iris.data, iris.target
model = RandomForestClassifier()
model.fit(X, y)

@app.route('/', methods=['GET', 'POST'])
def home():
    species = None
    error_message = None
    if request.method == 'POST':
        try:
            sepal_length = float(request.form['sepal_length'])
            sepal_width = float(request.form['sepal_width'])
            petal_length = float(request.form['petal_length'])
            petal_width = float(request.form['petal_width'])

            input_data = np.array([[sepal_length, sepal_width, petal_length, petal_width]])
            prediction = model.predict(input_data)

            species_names = ["Setosa", "Versicolor", "Virginica"]
            species = species_names[prediction[0]]

        except ValueError:
            error_message = "Please enter valid numbers for all fields."
        except Exception as e:
            error_message = f"An error occurred: {str(e)}"

    return render_template('index.html', species=species, error_message=error_message)

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

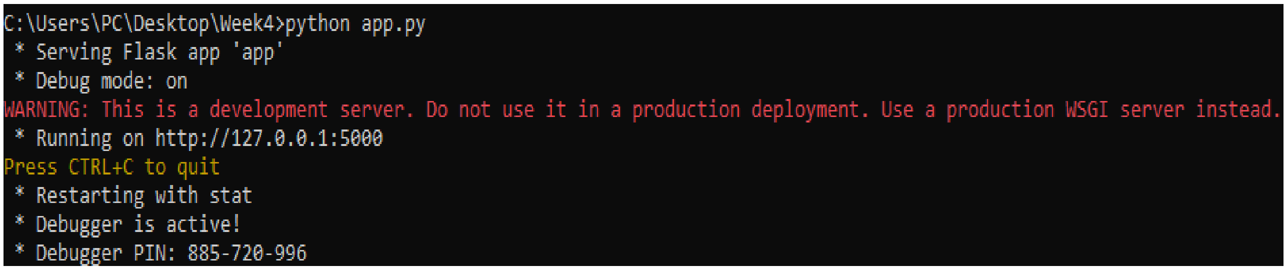
3) Creating an interface to display the model



```html
<body>
    <div class="container">
        <h1>Iris Species Prediction</h1>
        <form action="/" method="post">
            <label for="sepal_length">Sepal Length (cm, between 4.3 and 7.9):</label>
            <input type="text" id="sepal_length" name="sepal_length" required>

            <label for="sepal_width">Sepal Width (cm, between 2.0 and 4.4):</label>
            <input type="text" id="sepal_width" name="sepal_width" required>

            <label for="petal_length">Petal Length (cm, between 1.0 and 6.9):</label>
            <input type="text" id="petal_length" name="petal_length" required>

            <label for="petal_width">Petal Width (cm, between 0.1 and 2.5):</label>
            <input type="text" id="petal_width" name="petal_width" required>

            <input type="submit" value="Predict">
        </form>

        {% if error_message %}
            <div class="error">{{ error_message }}</div>
        {% endif %}

        {% if species %}
            <div class="result">The predicted species is: {{ species }}</div>
        {% endif %}
    </div>
</body>
</html>

```

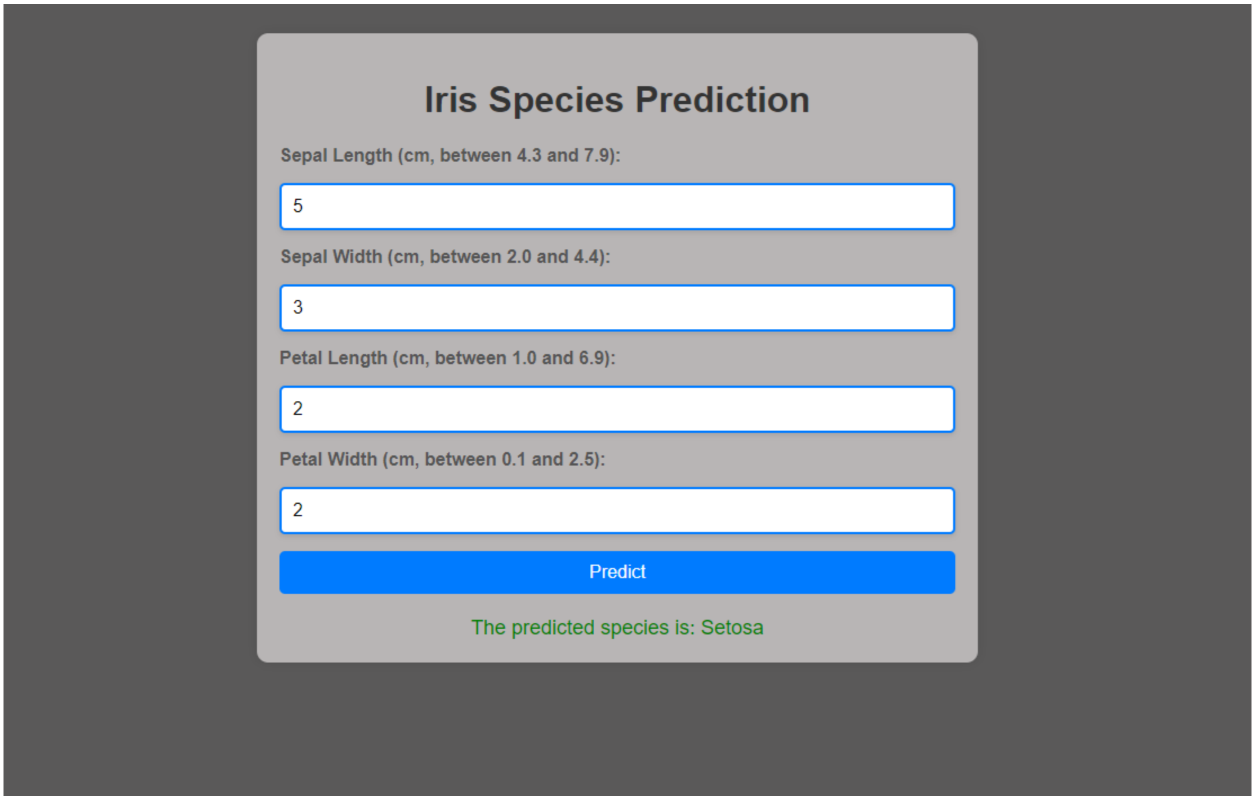


```html
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Iris Species Prediction</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #5a5959;
            margin: 0;
            padding: 0;
        }
        .container {
            width: 100%;
            max-width: 600px;
            margin: 50px auto;
            background-color: #b8b5b5;
            padding: 20px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            border-radius: 10px;
        }
        h1 {
            text-align: center;
            color: #333;
        }
        form {
            display: flex;
            flex-direction: column;
            gap: 15px;
        }
        label {
            font-weight: bold;
            color: #555;
```



```
C:\Users\PC\Desktop\Week4>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 885-720-996
```

4) Interface and Prediction



Iris Species Prediction

Sepal Length (cm, between 4.3 and 7.9): 5

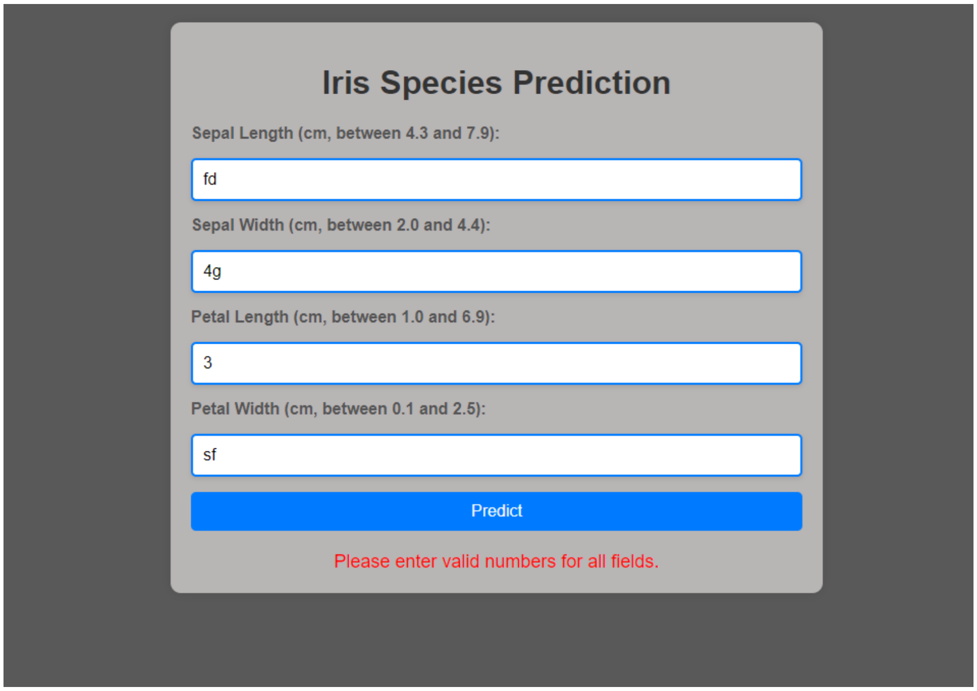
Sepal Width (cm, between 2.0 and 4.4): 3

Petal Length (cm, between 1.0 and 6.9): 2

Petal Width (cm, between 0.1 and 2.5): 2

The predicted species is: Setosa

If you enter letters instead of numbers, you will receive the following warning;



Iris Species Prediction

Sepal Length (cm, between 4.3 and 7.9): fd

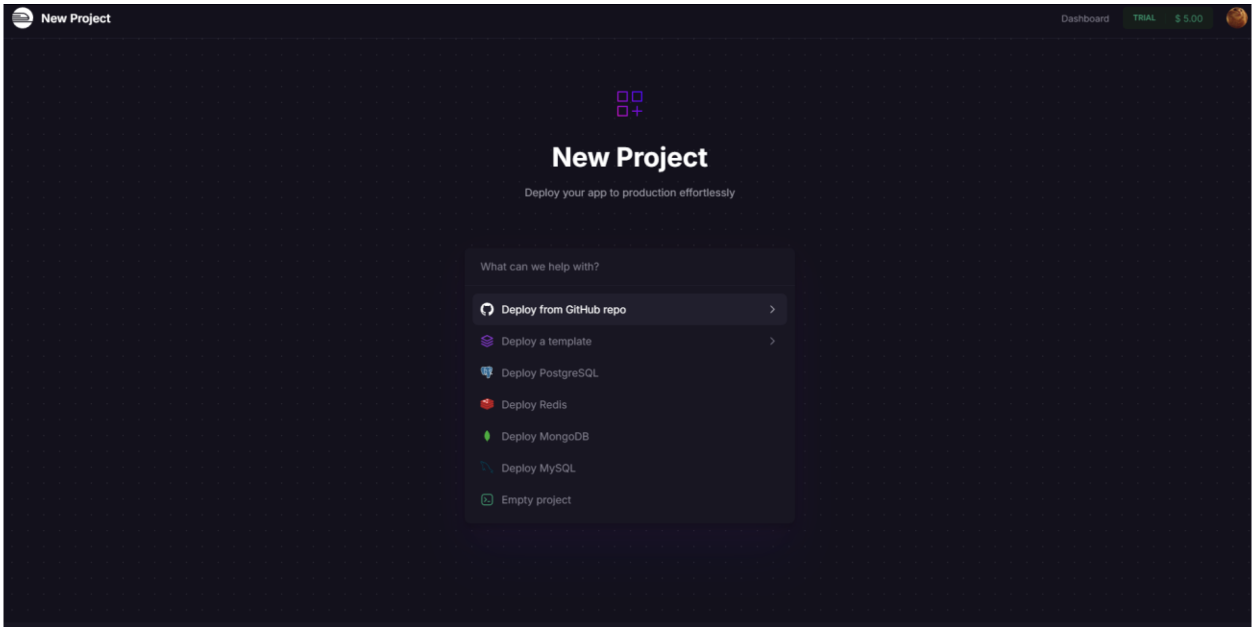
Sepal Width (cm, between 2.0 and 4.4): 4g

Petal Length (cm, between 1.0 and 6.9): 3

Petal Width (cm, between 0.1 and 2.5): sf

Please enter valid numbers for all fields.

5) Deployment

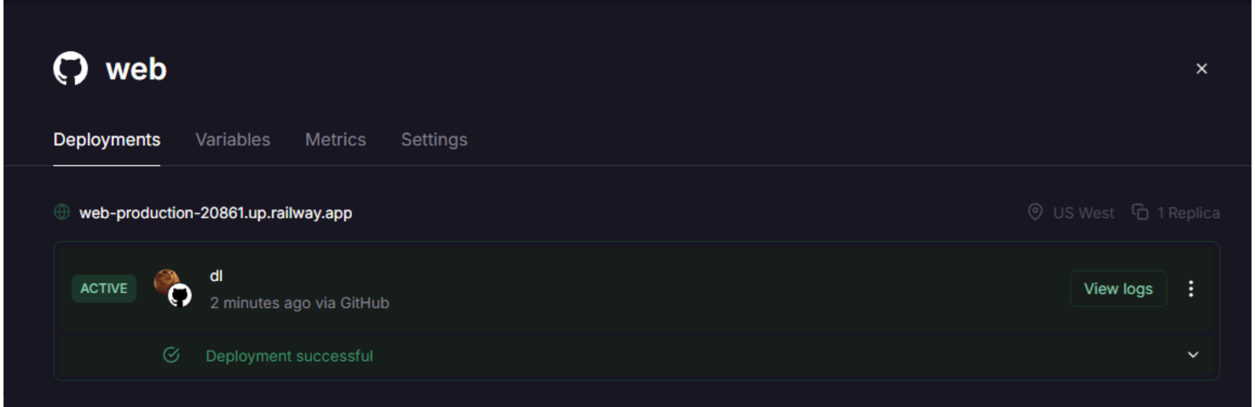


New Project

Deploy your app to production effortlessly

What can we help with?

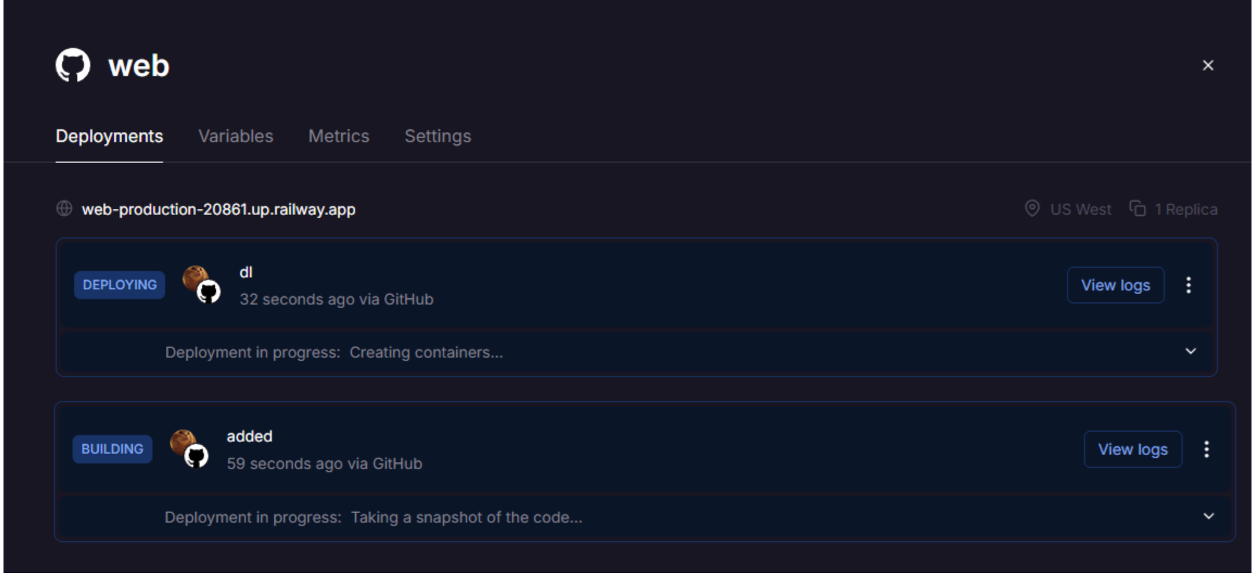
- Deploy from GitHub repo
- Deploy a template
- Deploy PostgreSQL
- Deploy Redis
- Deploy MongoDB
- Deploy MySQL
- Empty project



web-production-20861.up.railway.app

ACTIVE dl
2 minutes ago via GitHub

Deployment successful



web-production-20861.up.railway.app

DEPLOYING dl
32 seconds ago via GitHub

Deployment in progress: Creating containers...

BUILDING added
59 seconds ago via GitHub

Deployment in progress: Taking a snapshot of the code...