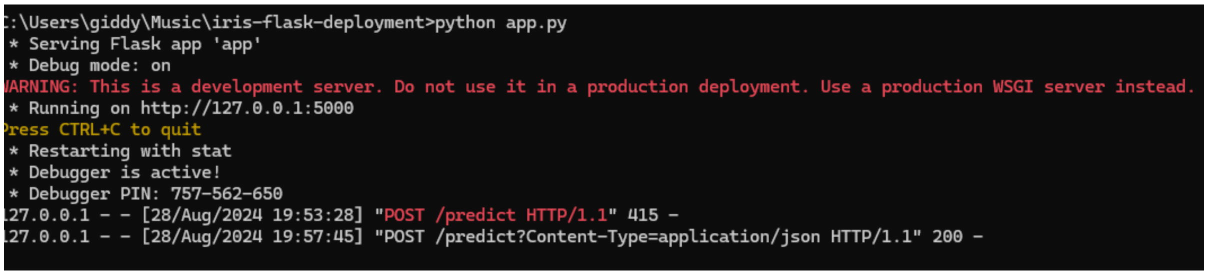
- **Name**: David Anthony
- **Batch Code**: LISUM36
- **Submission Date**: 08/28/2024
- **Submitted To**:Github and Canvas

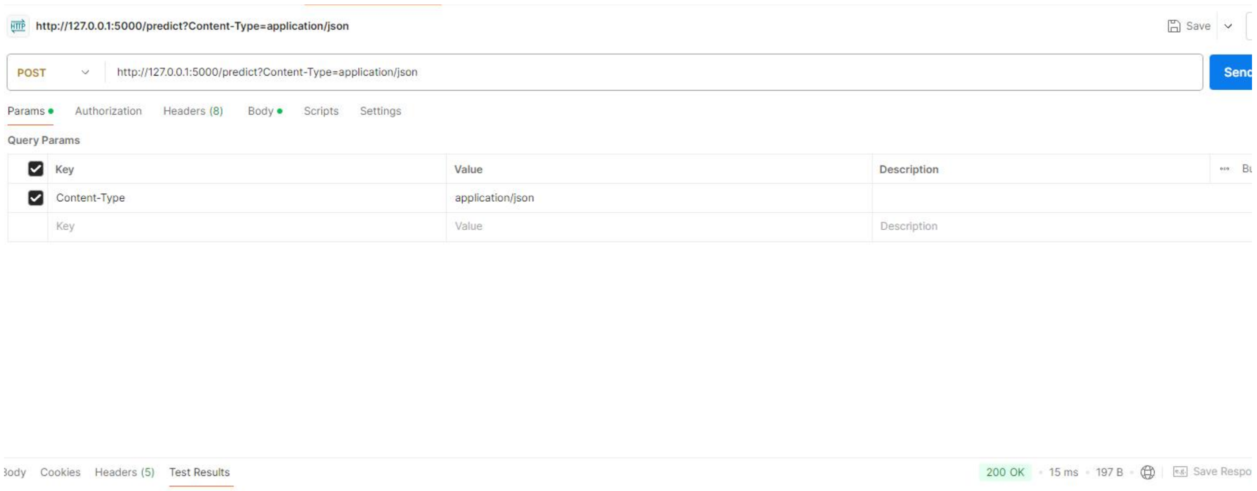
1. Flask Server running on



```
:\Users\giddy\Music\iris-flask-deployment>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: 757-562-650
127.0.0.1 - - [28/Aug/2024 19:53:28] "POST /predict HTTP/1.1" 415 -
127.0.0.1 - - [28/Aug/2024 19:57:45] "POST /predict?Content-Type=application/json HTTP/1.1" 200 -
```

   cmd

2. Post man Request and Post man Response:



http://127.0.0.1:5000/predict?Content-Type=application/json

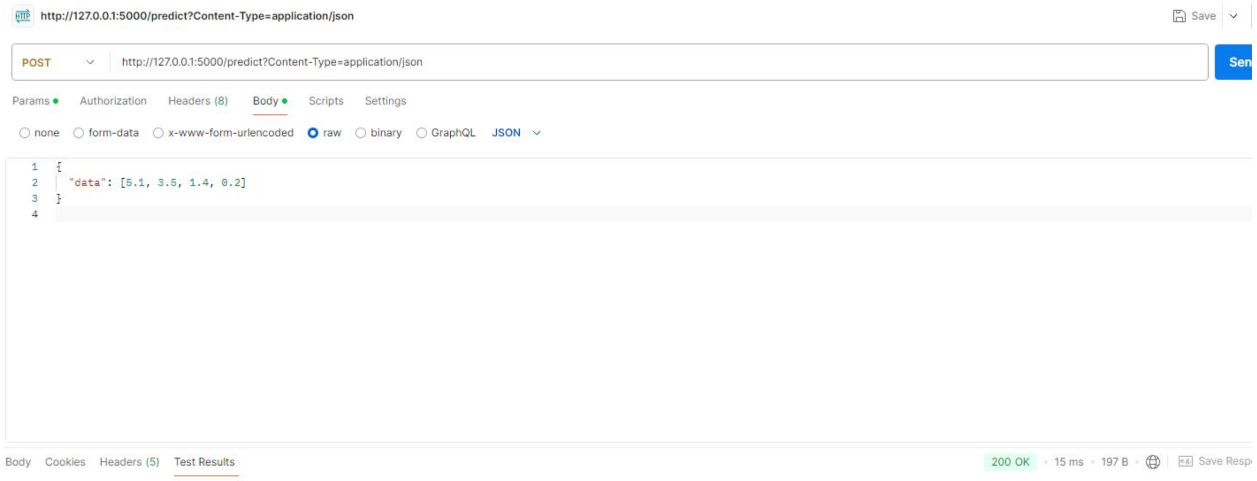
POST http://127.0.0.1:5000/predict?Content-Type=application/json

Params ● | Authorization | Headers (8) | Body ● | Scripts | Settings

Query Params

	Key	Value	Description
✓	Content-Type	application/json	
	Key	Value	Description

Body | Cookies | Headers (5) | Test Results — 200 OK · 15 ms · 197 B



http://127.0.0.1:5000/predict?Content-Type=application/json

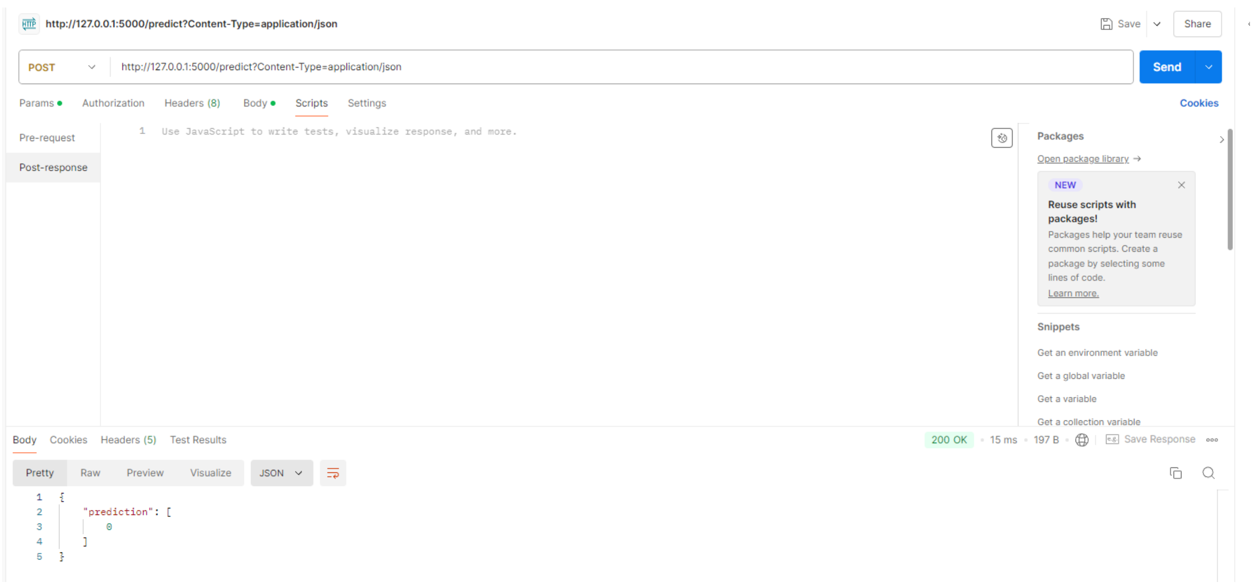
POST http://127.0.0.1:5000/predict?Content-Type=application/json

Params ● | Authorization | Headers (8) | Body ● | Scripts | Settings

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

```
{
  "data": [5.1, 3.5, 1.4, 0.2]
}
```

Body | Cookies | Headers (5) | Test Results — 200 OK · 15 ms · 197 B



```
POST http://127.0.0.1:5000/predict?Content-Type=application/json
```

```
Use JavaScript to write tests, visualize response, and more.
```

```
200 OK · 15 ms · 197 B
```

```
{
    "prediction": [
        0
    ]
}
```

Conclusion

In this project, we successfully deployed a machine learning model using Flask and verified its functionality through Postman. Here's a brief overview of the steps and results:

3. **Model Deployment**:
    - We began by developing and training a simple machine learning model on toy data. This model was then saved and prepared for deployment.
    - Using Flask, we created a web application that hosts our trained model. This application was set up to handle HTTP requests and provide predictions based on input data.
4. **Testing with Postman**:
    - To ensure our Flask application was working correctly, we used Postman to send test requests to our Flask server. This involved setting up the request with the appropriate headers and body, and then examining the response.
    - A successful response was received, showing that our model was correctly integrated with the Flask server. The response contained the predicted output in JSON format, confirming that the deployment was successful.
5. **Documentation**:
    - We documented each step of the process, including screenshots of the Flask server running, Postman request setup, and the response received. This documentation helps in verifying the deployment and serves as a clear record of the work completed.

In summary, the deployment and testing of the model were successful. The Flask web application responded accurately to the test data provided, demonstrating that the integration of machine learning models into web applications is both feasible and effective. This project showcases the practical application of deploying machine learning models and provides a foundation for future projects involving similar technologies.

BY: DAVID ANTHONY