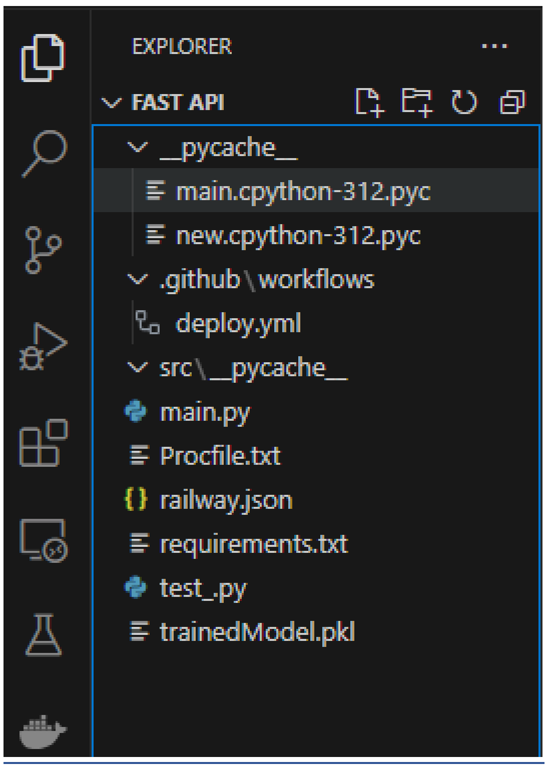
Deploying a Machine Learning Model Using FastAPI

Contents

Folder Architecture 2
main.py 3
GIT Hub link :-.......... 5
Deployed App Link :- 6

Folder Architecture

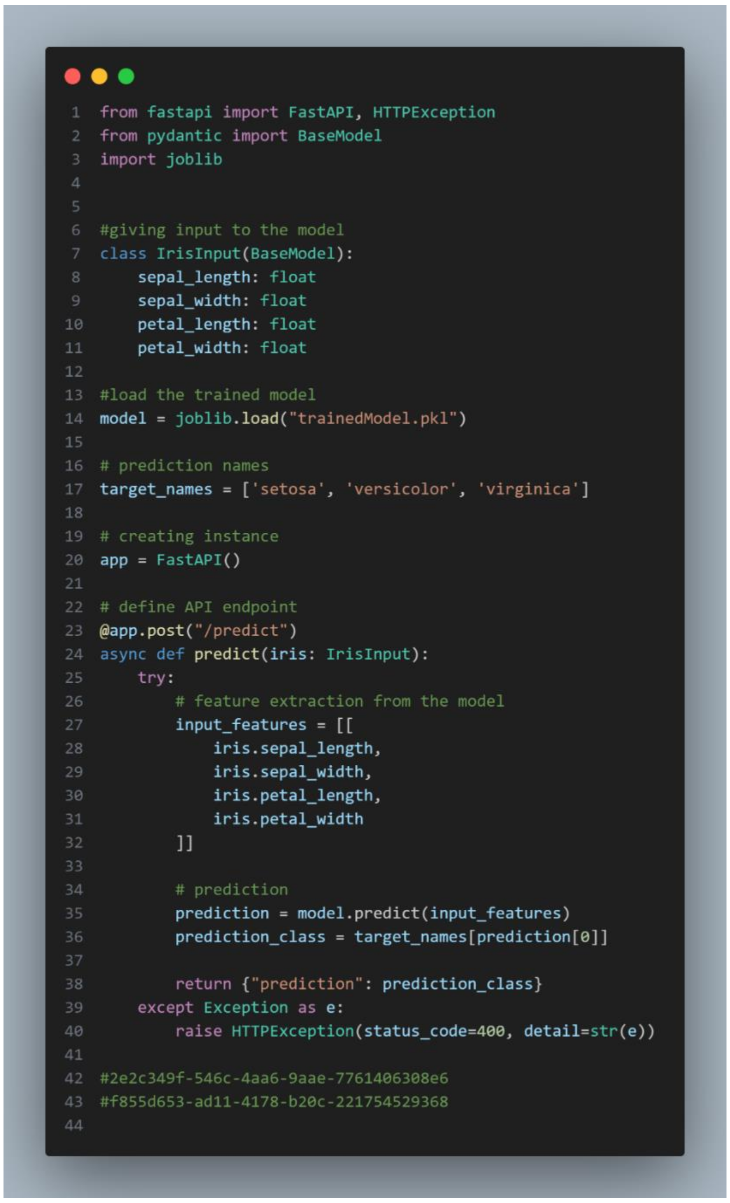


```
EXPLORER ...
FAST API
  _pycache_
    main.cpython-312.pyc
    new.cpython-312.pyc
  .github\workflows
    deploy.yml
  src\_pycache_
  main.py
  Procfile.txt
  railway.json
  requirements.txt
  test_.py
  trainedModel.pkl
```

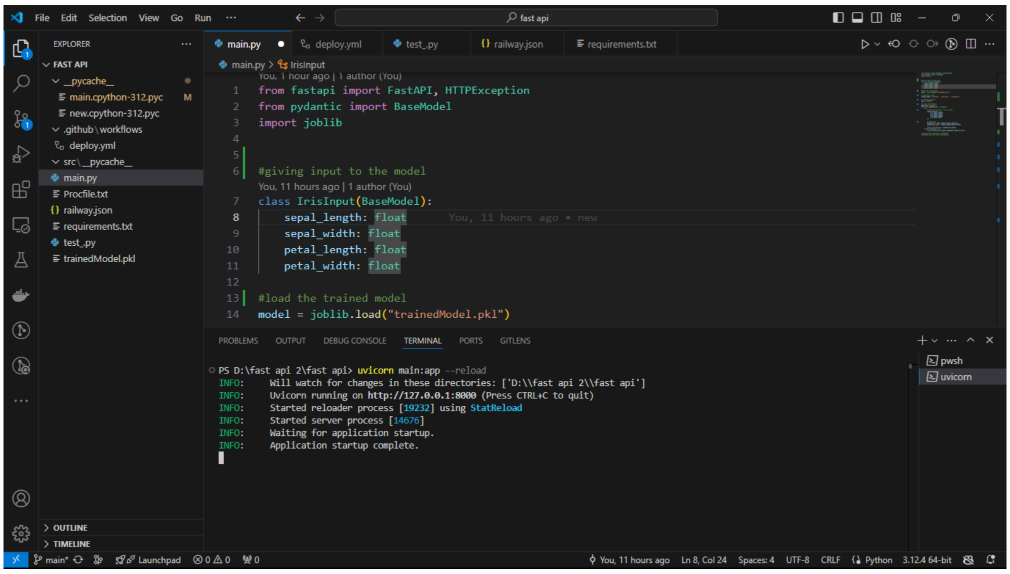
main.py

In this FastAPI code, I've created a machine learning model that can predict the species of an iris flower based on its features (sepal length, sepal width, petal length, and petal width). I've saved this trained model as a .pkl file using joblib. The FastAPI application loads this model and uses it to make predictions when it receives input data in the specified format.

- The FastAPI application defines a data model (IrisInput) to structure the input data.
- It loads the pre-trained model from the trainedModel.pkl file.
- It sets up an API endpoint (/predict) that takes input data, uses the model to predict the iris flower's species, and returns the result.
- If there's any issue with the prediction process, it returns an appropriate error message.

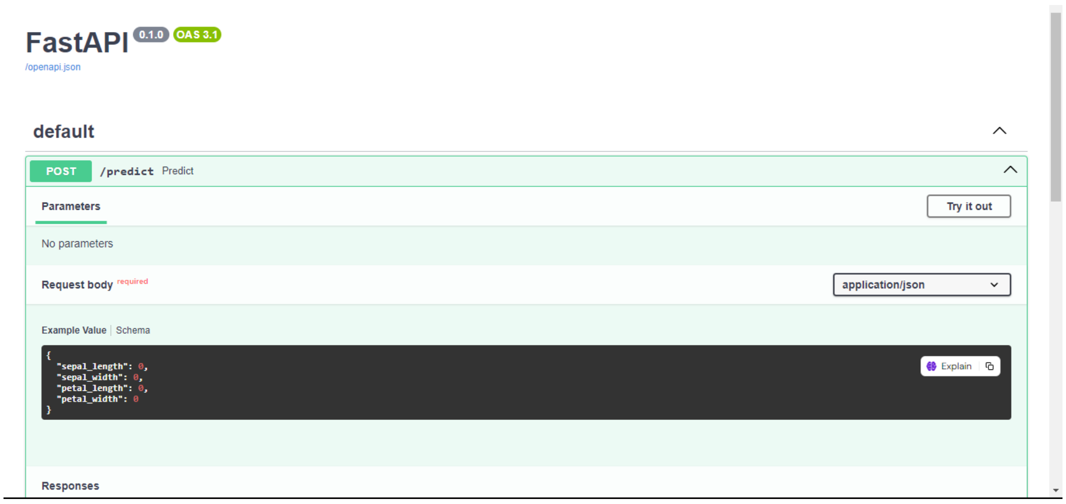


```python
from fastapi import FastAPI, HTTPException
from pydantic import BaseModel
import joblib


#giving input to the model
class IrisInput(BaseModel):
    sepal_length: float
    sepal_width: float
    petal_length: float
    petal_width: float

#load the trained model
model = joblib.load("trainedModel.pkl")

# prediction names
target_names = ['setosa', 'versicolor', 'virginica']

# creating instance
app = FastAPI()

# define API endpoint
@app.post("/predict")
async def predict(iris: IrisInput):
    try:
        # feature extraction from the model
        input_features = [[
            iris.sepal_length,
            iris.sepal_width,
            iris.petal_length,
            iris.petal_width
        ]]

        # prediction
        prediction = model.predict(input_features)
        prediction_class = target_names[prediction[0]]

        return {"prediction": prediction_class}
    except Exception as e:
        raise HTTPException(status_code=400, detail=str(e))

#2e2c349f-546c-4aa6-9aae-7761406308e6
#f855d653-ad11-4178-b20c-221754529368

```



```python
from fastapi import FastAPI, HTTPException
from pydantic import BaseModel
import joblib


#giving input to the model
class IrisInput(BaseModel):
    sepal_length: float
    sepal_width: float
    petal_length: float
    petal_width: float

#load the trained model
model = joblib.load("trainedModel.pkl")
```

```
PS D:\fast api 2\fast api> uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['D:\\fast api 2\\fast api']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [19232] using StatReload
INFO:     Started server process [14676]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```



FastAPI 0.1.0 OAS 3.1

/openapi.json

default

POST /predict Predict

Parameters

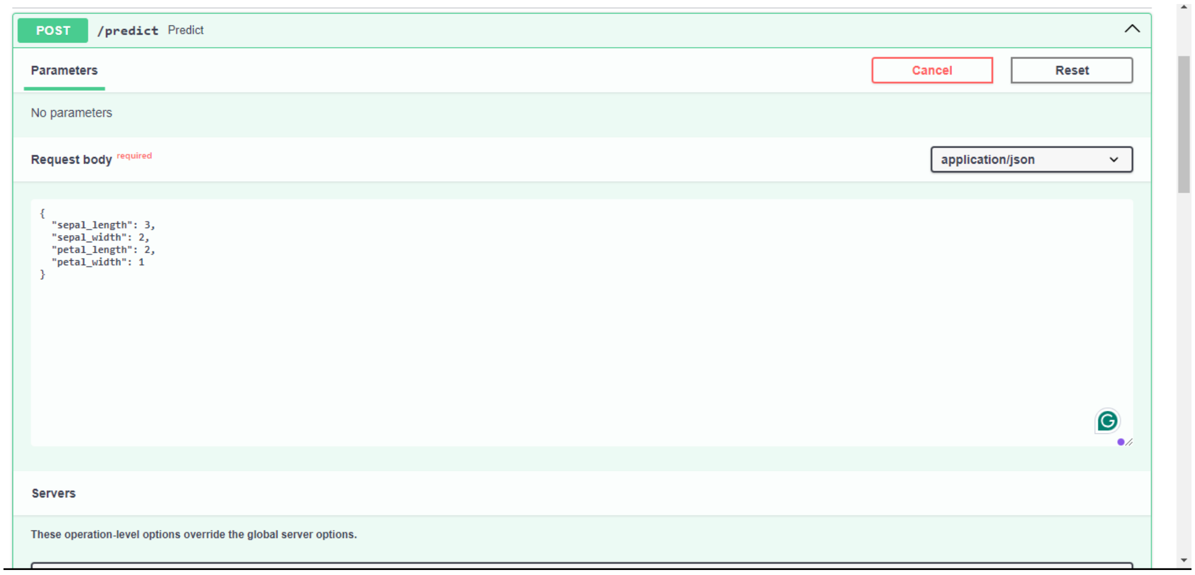
No parameters

Request body required: application/json

Example Value | Schema

```json
{
  "sepal_length": 0,
  "sepal_width": 0,
  "petal_length": 0,
  "petal_width": 0
}
```

Responses



POST /predict Predict

Parameters

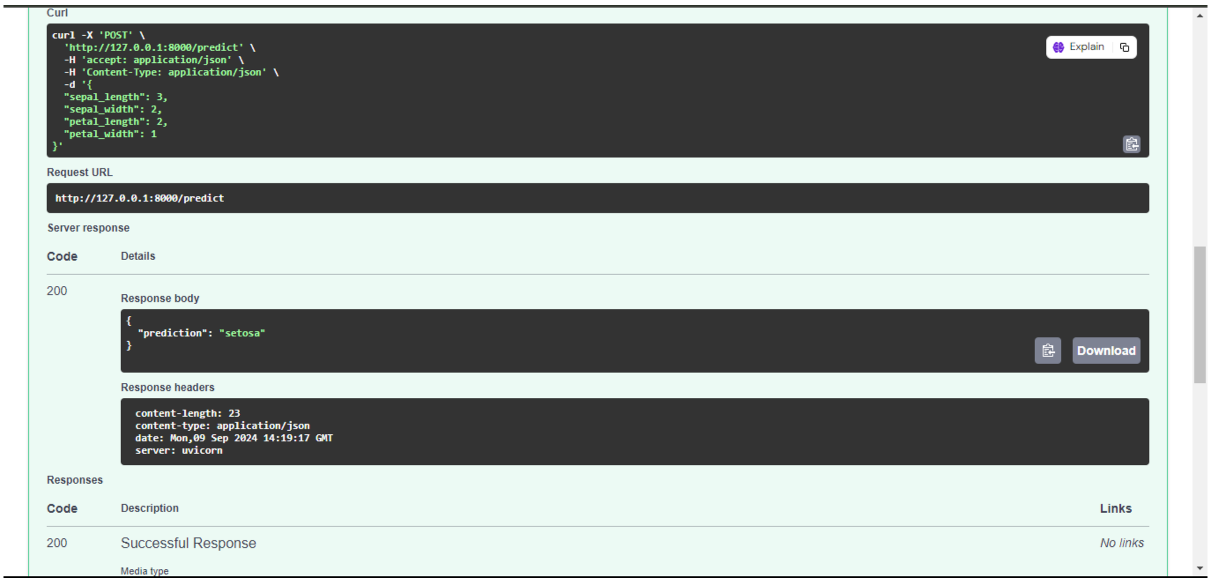
No parameters

Request body required: application/json

```json
{
  "sepal_length": 3,
  "sepal_width": 2,
  "petal_length": 2,
  "petal_width": 1
}
```

Servers

These operation-level options override the global server options.



```
curl -X 'POST' \
  'http://127.0.0.1:8000/predict' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "sepal_length": 3,
  "sepal_width": 2,
  "petal_length": 2,
  "petal_width": 1
}'
```

```
http://127.0.0.1:8000/predict
```

Code	Details
200	Response body

```
{
  "prediction": "setosa"
}
```

Response headers

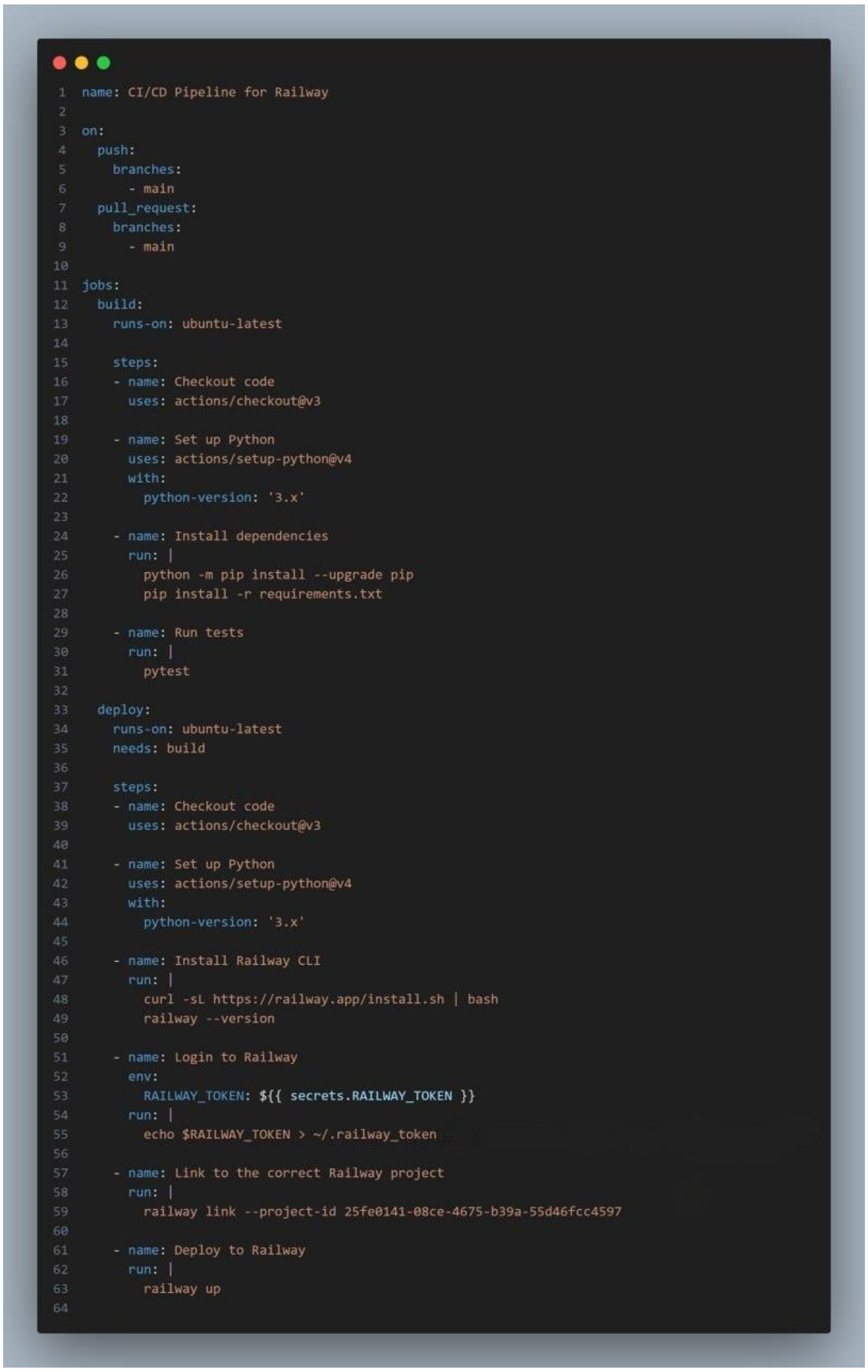
```
content-length: 23
content-type: application/json
date: Mon,09 Sep 2024 14:19:17 GMT
server: uvicorn
```

Code	Description	Links
200	Successful Response	No links

(This images will show the results of the swagger Ui.)

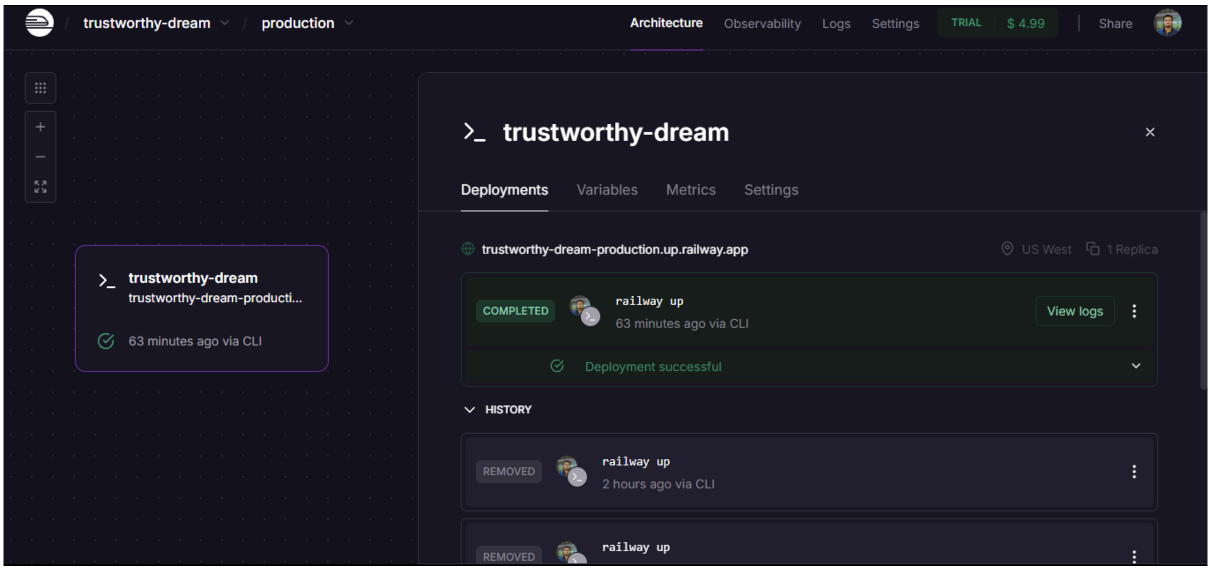
GIT Hub link :-

https://github.com/SerujanSatkunanathan/fast-api-Railway-model-deployement



```yaml
name: CI/CD Pipeline for Railway

on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v3

    - name: Set up Python
      uses: actions/setup-python@v4
      with:
        python-version: '3.x'

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Run tests
      run: |
        pytest

  deploy:
    runs-on: ubuntu-latest
    needs: build

    steps:
    - name: Checkout code
      uses: actions/checkout@v3

    - name: Set up Python
      uses: actions/setup-python@v4
      with:
        python-version: '3.x'

    - name: Install Railway CLI
      run: |
        curl -sL https://railway.app/install.sh | bash
        railway --version

    - name: Login to Railway
      env:
        RAILWAY_TOKEN: ${{ secrets.RAILWAY_TOKEN }}
      run: |
        echo $RAILWAY_TOKEN > ~/.railway_token

    - name: Link to the correct Railway project
      run: |
        railway link --project-id 25fe0141-08ce-4675-b39a-55d46fcc4597

    - name: Deploy to Railway
      run: |
        railway up

```

Deployed App Link :- **https://trustworthy-dream-production.up.railway.app/**



trustworthy-dream
trustworthy-dream-producti...
63 minutes ago via CLI

>_ trustworthy-dream

Deployments Variables Metrics Settings

trustworthy-dream-production.up.railway.app US West 1 Replica

COMPLETED `railway up` 63 minutes ago via CLI View logs

Deployment successful

HISTORY

REMOVED `railway up` 2 hours ago via CLI

REMOVED `railway up`