

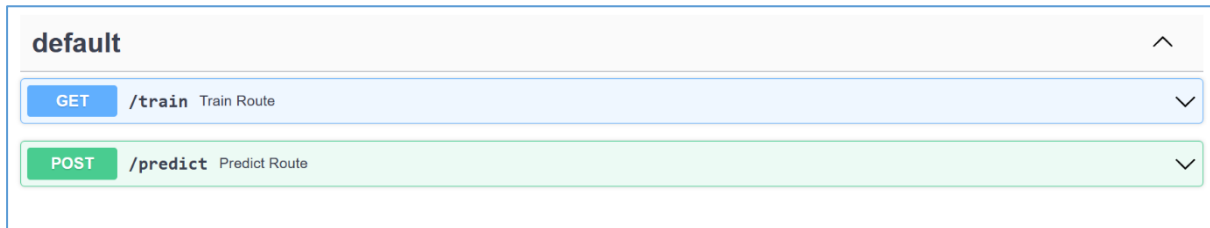
WIREFRAME DOCUMENTATION

Gesture Prediction using sensors

Written By	Prakhyath Bhandary
Revision Number:	1.0
Last date of revision:	10/12/2022

Home Page

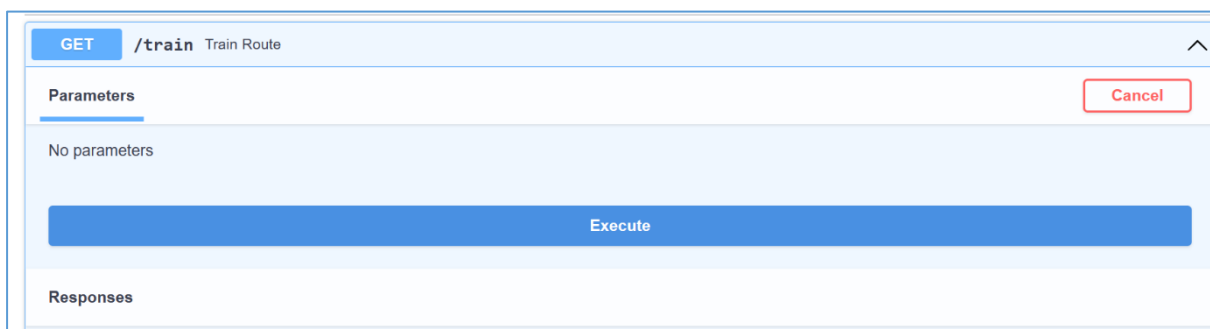
Here home page is created using FastApi. Home page is divided into two routes: Predict route and train route



The wireframe shows a container with a header labeled 'default' and an upward arrow. Below the header are two route entries. The first entry is for a GET request to '/train' labeled 'Train Route', with a dropdown arrow on the right. The second entry is for a POST request to '/predict' labeled 'Predict Route', also with a dropdown arrow on the right.

Train Route:

On clicking execute from train route, data is fetched from Cassandra and then ingested, validated, transformed and used for training the model. Then the model is evaluated and if the model performance is better compared to previous best model by a minimum chosen threshold value we push the model for production.



The wireframe shows the execution interface for the GET /train route. It includes a header with the route name and an upward arrow. Below the header is a 'Parameters' section with a 'Cancel' button. The parameters section is currently empty, showing 'No parameters'. Below the parameters section is a large blue 'Execute' button. At the bottom is a 'Responses' section.

Predict Route:

On clicking the predict route there will be an option to upload a csv file with the necessary inputs and the prediction will be displayed back.

The screenshot shows a REST client interface for a POST request to the `/predict` endpoint. The interface includes a 'Parameters' section with 'No parameters', a 'Request body' section with a 'required' label and a 'multipart/form-data' dropdown, and a file upload section with a 'file * required' label, a 'string(\$binary)' type, a 'Choose File' button, and a 'No file chosen' status. At the bottom, there is an 'Execute' button.

POST `/predict` Predict Route

Parameters

No parameters

Request body **required**

multipart/form-data

file * **required**

string(\$binary)

Choose File No file chosen

Execute

The screenshot shows the server response for the POST request to the `/predict` endpoint. The 'Request URL' is `http://127.0.0.1:8080/predict`. The 'Server response' section shows a 'Code' of 200 and a 'Details' tab. The 'Response body' is a JSON array of objects, each containing an 'idx' and a 'prediction'.

Request URL

`http://127.0.0.1:8080/predict`

Server response

Code Details

200

Response body

```
[
  {
    "idx": 0,
    "prediction": 1
  },
  {
    "idx": 1,
    "prediction": 1
  },
  {
    "idx": 2,
    "prediction": 1
  },
  {
    "idx": 3,
    "prediction": 6
  },
  {
    "idx": 4,
    "prediction": 2
  },
  {
    "idx": 5,
    "prediction": 2
  }
]
```