**ML-Models-Monitoring System Architecture**

****

In this architecture there are 2 main components i.e ML-Monitor and ML-Models-Server and database system:

1. **ML-Monitor**

ML-Monitor server requests the data to ML-Models Server and then ML-Models server process the requests and revert back the response

1. **ML-Models Server**

ML-Models server finds the data in the database and returns the result to the ML-Monitor server.

1. **Models**

Models are the actual machine learning models which are running in background and based on the actions they are putting monitoring data on to the database

**ML-Models-Monitoring System Code Structure**

Project consists of two main applications i.e ml\_monitor and ml\_models. Each application is based on flask and its directory structure & file conventions. Same overview is presented below:

* Pipfile -> Pipenv configuration file
* Pipfile.lock-> Contains packages dependencies configuration
* app.py -> Entry point of the application
* src -> Application specific modules
* models -> Database interface api is defined in this module
* routes -> APIs are defined in this module for the applications
* usecases -> All the use cases implemented by the application are defined in this module
* Tests -> Test cases are defined in this module
* config.py -> All the application configuration variables are placed in this file
* db.json -> Database schema file
* run\_model.py -> This program trains the ml models (housing\_price, payment\_fraud) and push the result to the database
* test.py -> By executing this file, it initiates the testing and shows results of test cases
* housing\_price & payment\_fraud -> Machine Learning models