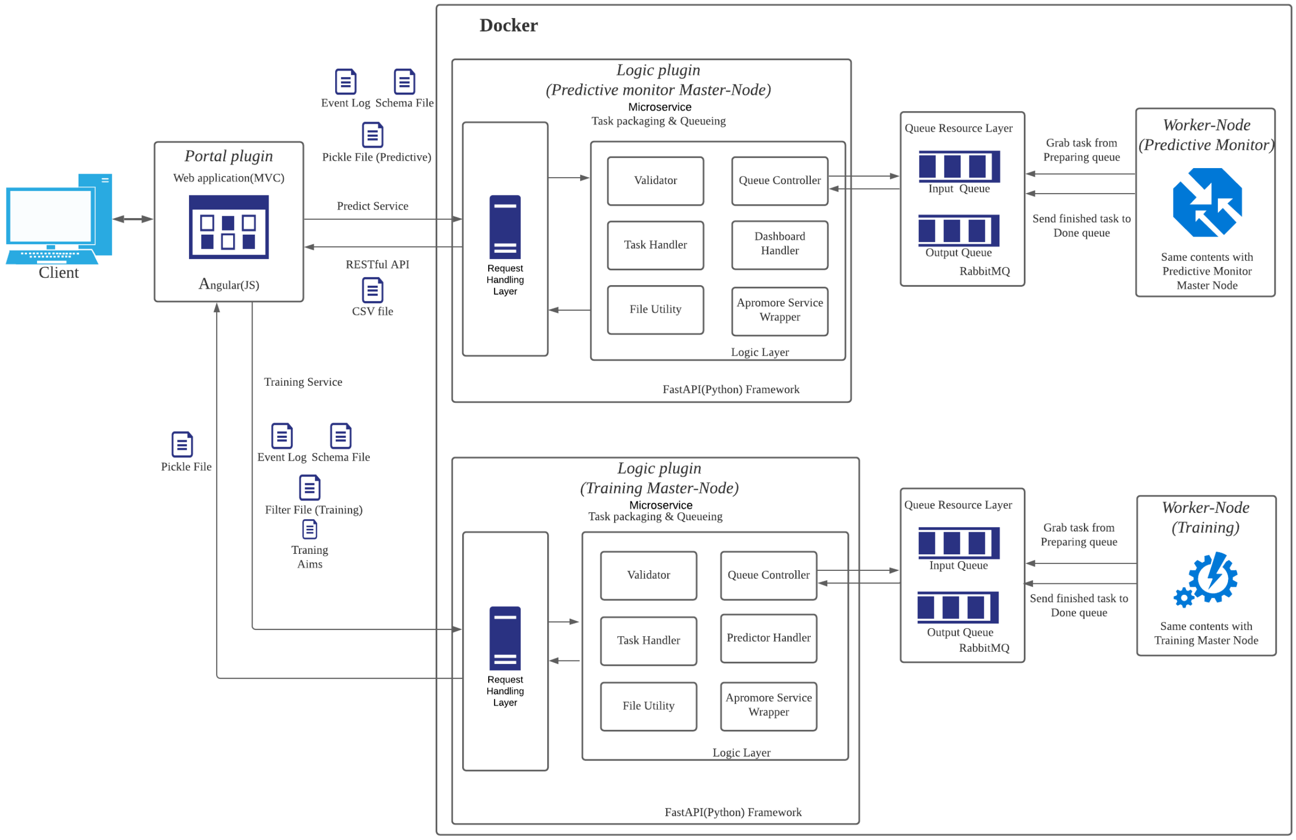
System Architecture Diagram

Version Control

Version	Date	Update notes
v1	02 Apr 2021	- The initial draft of a high-level architecture diagram based on client documents and client meeting
v2	08 Apr 2021	- Restructured architecture based on next client meeting
v3	12 Apr 2021	- Added descriptions to components
v4	18 Apr 2021	- Restructure architecture based on last client meeting, including master-slave architecture and more granular modules
v5	01 May 2021	- Separate two services into two images, and add and remove some modules according to the last client meeting
v6(Current)	03 May 2021	- Make the system more clearly that the Master node and the Worker node would pass task by the queueing system. - Change Predictor Handler to Dashboard Handler in Predictive monitor service - Remove Filter Handler - Change names from Preparing queue, Done queue to Input queue, Output queue



Description

In v2 of the system architecture, we changed over to a simpler client-server module and a layered architecture on the back-end to make the system lighter and more concise, according to the client's new specification.

In v4 of the system architecture, the master-slave architecture was implemented, in which one instance of the microservice acts as a master node and assigns other instances set as worker nodes to handle the requests. This is done with Docker Compose, with each instance of the microservice containerized and communicating with each other. The separation of modules has also been made more granular.

In v5 of the system architecture, the service of Predictive Monitoring and Training are separated into two images. The distribution method is took placed by a producer-consumer module between Master and Worker node. And the client clarified that the monitor and dashboard for the system is only a form for display information. Therefore, monitor and dashboard modules have been removed.

Terminology

- Deployment
    - Portal Plugin - Act as the web application. Validate requests and temporarily saves and displays monitors.
    - Master Node - Responsible for packaging requests into tasks, placing them in the preparing queue. Also returns tasks from the done queue and sends back results or generate files that the client needed.
    - Worker Node - Responsible for taking enqueued tasks, and processing them, and placing them back in the done queue
- Request Handling Layer - Acts as an intermediary between the user's requests and the other modules
- Queue Resource Layer - Acts as a middle layer for tasks that passes within the system
- Logic Layer - Encapsulate all the modules required to process requests for training or prediction
    - Modules
        - Queue Controller - Acts as an intermediary between the task queue and other modules
        - Task Handler - Handles the CRUD of the task and checks the status of a task
        - Dashboard Handler - Handles the CRUD of predictor, temporarily saves data using in the Predictive Monitor service
        - Predictor Handler - Handles the CRUD of predictor, temporarily saves data using in the Training service
        - Validator - Validates the event log formats with schema files
        - File Utility - Handles processing, serializing and converting files, ensuring that they are of the proper format depending on the request
        - Apromore Service Wrapper - API that calls methods in Apromore's machine learning library

Note: The validation of request would be handled in Portal Plugin, which is the server-side of the web application. No need to involve it with the logic layer.

Note: Monitor is a container that holds name and predictors' information. No need to involve it with the logic layer.

Note: CRUD refers to Create, Read, Update, Delete