Group 3 Team 4
Sensor Manager, Workflow manager, Sample Applications

Team Mambers

Dishant Sharma (2022202019)
Udrasht Pal (2022201020)
Vikram Raj Bakshi (2022201044)
Nikhil Chawla (2022201045)

Functional Overview

Our team is responsible to develop 3 components SensorManager, Workflow Manager and Sample Applications

1) **Sensor Manager**

    a) It controls all the sensors deployed at various locations and acts as an interface between different modules and sensors for providing the data.
    b) It registers and configures sensors (type & instance) and controllers(type & instance) based on config files.
    c) The module provides flexibility to add different types of sensors & controllers to the platform and manage them efficiently.
    d) It uses Kafka to fetch sensor data and sends it to the application as required.

2) **Workflow Manager**

    a) It triggers all the modules one by one by just initiating the starting module.
    b) It used the workflowmanager.json file to generate the code for automatically starting the flow.

3) **Sample Application**

    a) It is uploaded when an application developer upload it on platform.
    b) After uploading it will be visible to users of that application who want to run that application.
    c) When user run it one instance of it goes to deployment manager for deployment.
    d) At the time of deployment sensor binding happens at this time these application binds with sensors data.

2.1. Functional Requirements

2.1.1. Registering Sensors : Whenever a sensor is added to the platform for the first type, sensor registration is performed so that its data can be retrieved.

2.1.2. Interaction with IoT Sensors : APIs are built to provide abstraction to the application developer.

2.1.3. Identification of Sensors for Binding : After sensor registration, all sensor-related information (like sensor id, sensor type, sensor data, etc.) is stored in the database. Whenever a binding request is received, the sensor manager will identify the sensor based on the parameters and send data for binding in specified format.

2.1.4 Upload application without writing code by uploading json or just by using UI.

2.2. Non-functional Requirement

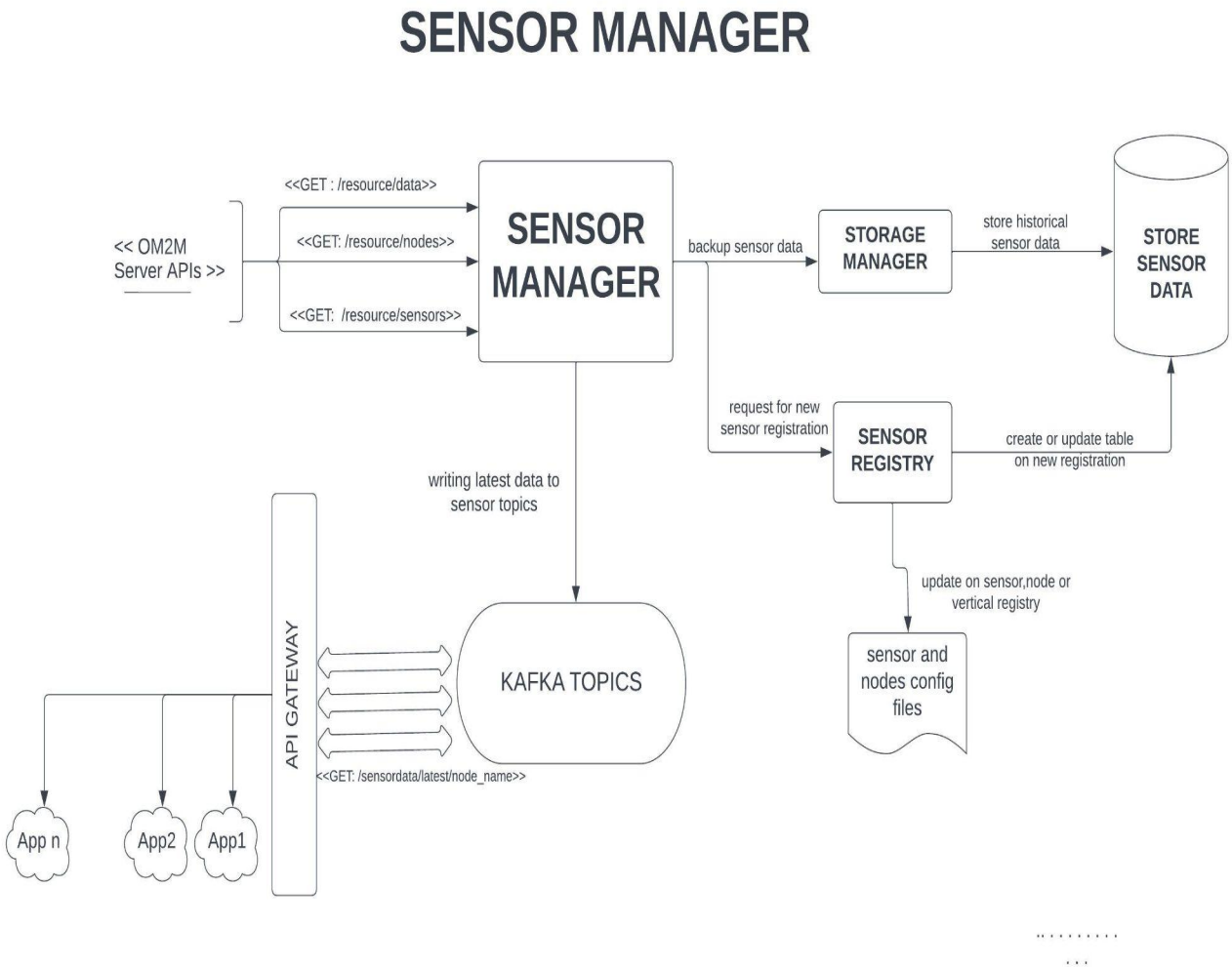
2.2.1. Scalability : Any number of sensors and controllers can be registered with the platform.

2.2.2. Accessibility of data : Application will specify in the configuration file the sensors that it will use and it will be able to access data from those sensors.

2.2.3. Monitoring : The module is monitored by the heart-beat module and sends continuous signals to it.

Block Diagram:-

Sensor Manager



List of SubSystems:-

- Kafka for communication
- Sensor Database
- Sensor
- Sensor Controller
- Configuration files

List of Services:-

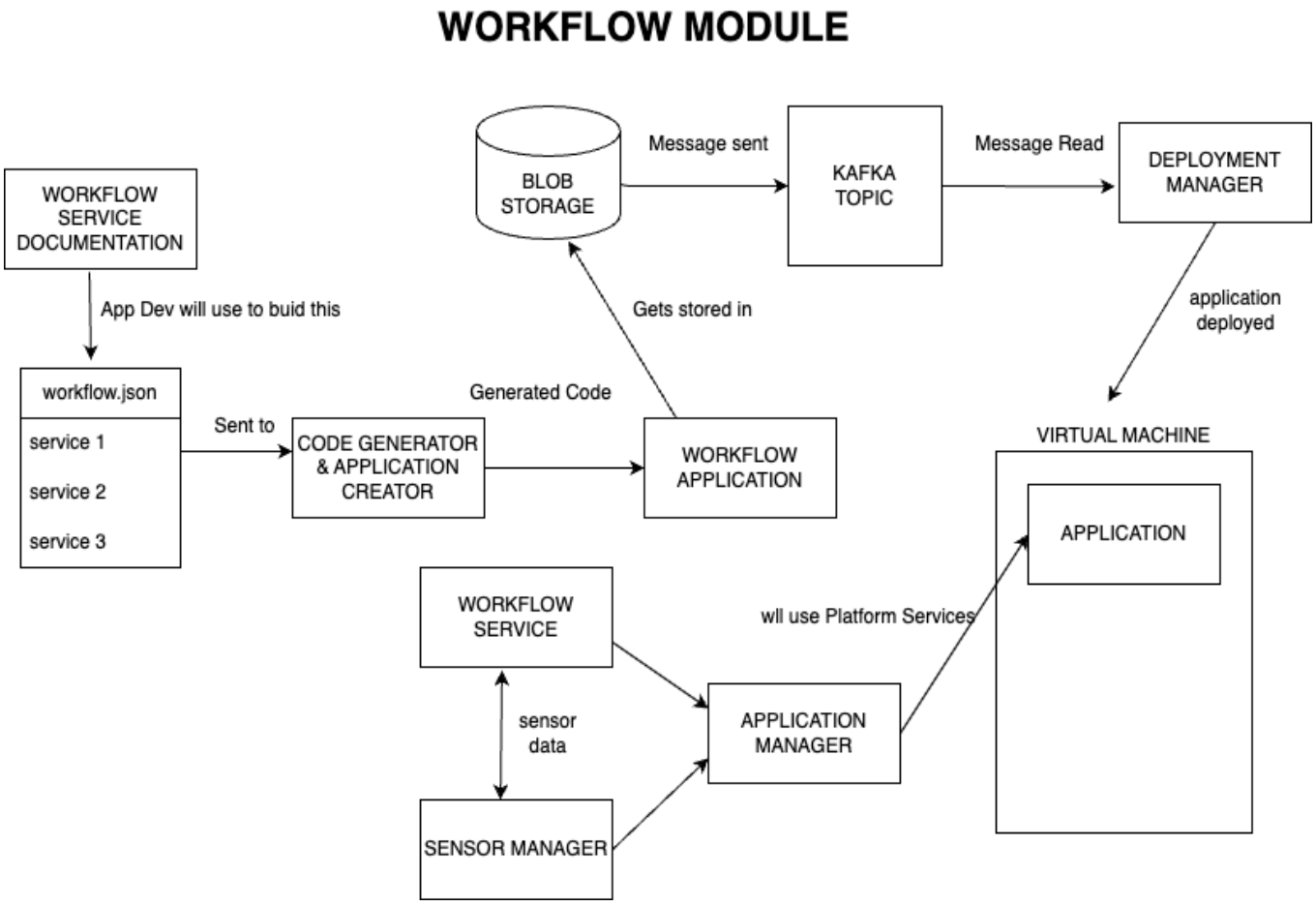
- **Sensor Manager**
    - Responsible for binding the correct sensor data stream with the corresponding algorithm.
    - The sensor manager will parse the Sensor Config. file and create schemas, tables, etc for these sensors.
    - When the sensor manager receives a data binding request, it uses the provided parameters to identify the corresponding sensor and sends the requested data in the specified format for binding.
    - If an application instance on a node requires data from a sensor, it will communicate with the sensor manager and provide the sensor's ID.
    - The sensor manager will use the ID to locate the correct sensor and send the requested data to the application in the required format.
    - It identifies IoT devices from the sensor database based on the specified parameters.

Interaction with other components

Interaction with Application Manager : Sensor manager is responsible for sharing the sensor data with the application manager, which will then store it in the application database along with other relevant information. The application manager can ask for data from some specific sensors by mentioning their unique IDs to the sensor manager.

Interaction with Node Manager : Application instances running on nodes will contact sensor manager to get sensor data and eventually send some control messages to the controller manager to take some action based on the sensor data or user requirement.

Interaction with Heart-beat Manager : Sensor & controller managers will continuously send heartbeat messages after some constant interval of time.



The Application Developer will make use of existing API i.e Platform Services to create a workflow.json and which will be sent to the Code Generator module. The Code Generator module will generate a code , and store that in the shared Blob Storage. Now, that will be picked by the Deployment Manager module and it will be deployed. The end point of the deployed workflow will be given to the Application Developer and can be used in the application for further use.

Lifecycle:

- When the platform is initialized the Code Generator submodule will be ready to accept the workflow.json file
- After generating code , this will be consumed by Deployment Manager and in return it will send the end point to access that

Functionality

- It serves the purpose of low code by allowing the application developer to just mention the flow of the code.
- It provides the end point of the workflow which can further be used in any application code thus making it reusable.

Interaction with other components

Communicate with application manager : Workflow communicate with application manager to send Application config to upload it in Application DB.