Group 3 - Team 5

User Interface, Sensor Manager, Platform Initializer, Workflow Manager

Members:-

- Harshit Kashyap (2022201050)
- Prashant Kumar (2022201058)
- Rishabh Gupta (2022202011)

Functional Overview:-

Our team is responsible for developing 4 components User Interface, Sensor Manager, Platform Initializer, Workflow Manager

1) **User Interface**

   a) There is a login page where each user(Platform admin, Application developer, End-user) can log in and access their respective services.
   b) Each respective page contains many fields which can be accessed individually by all three of them.

2) **Sensor Manager**

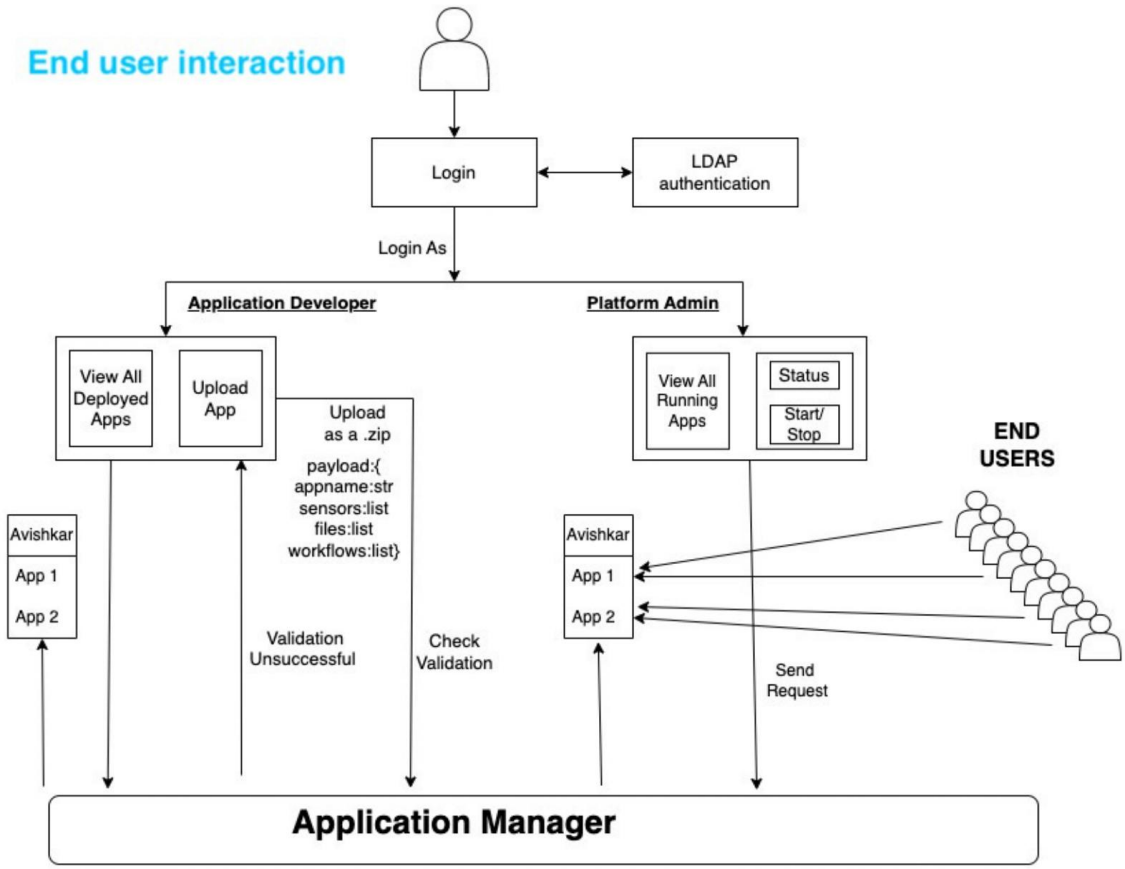
   a) It controls all the sensors deployed at various locations and acts as an interface between different modules and sensors for providing the data.
   b) It registers and configures sensors (type & instance) and controllers(type & instance) based on config files.
   c) The module provides flexibility 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.

3) **Platform Initializer**

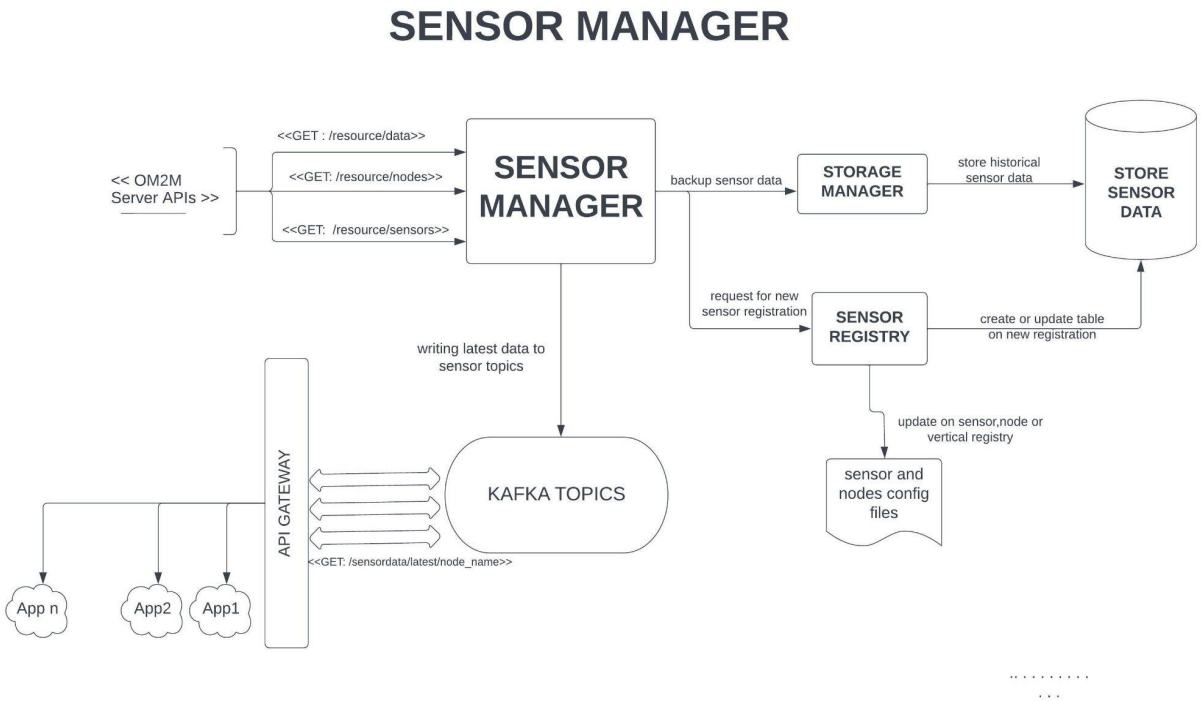
a) Platform Initializer can be considered a Bootstrapper module.
b) This component will initialize all the other modules (services or various managers).
c) Platform initializer will create the necessary infrastructure to support our IOT platform.
d) The platform initializer will achieve this by initializing virtual machines on the host and setting up the environment on virtual machines. It deploys all the module

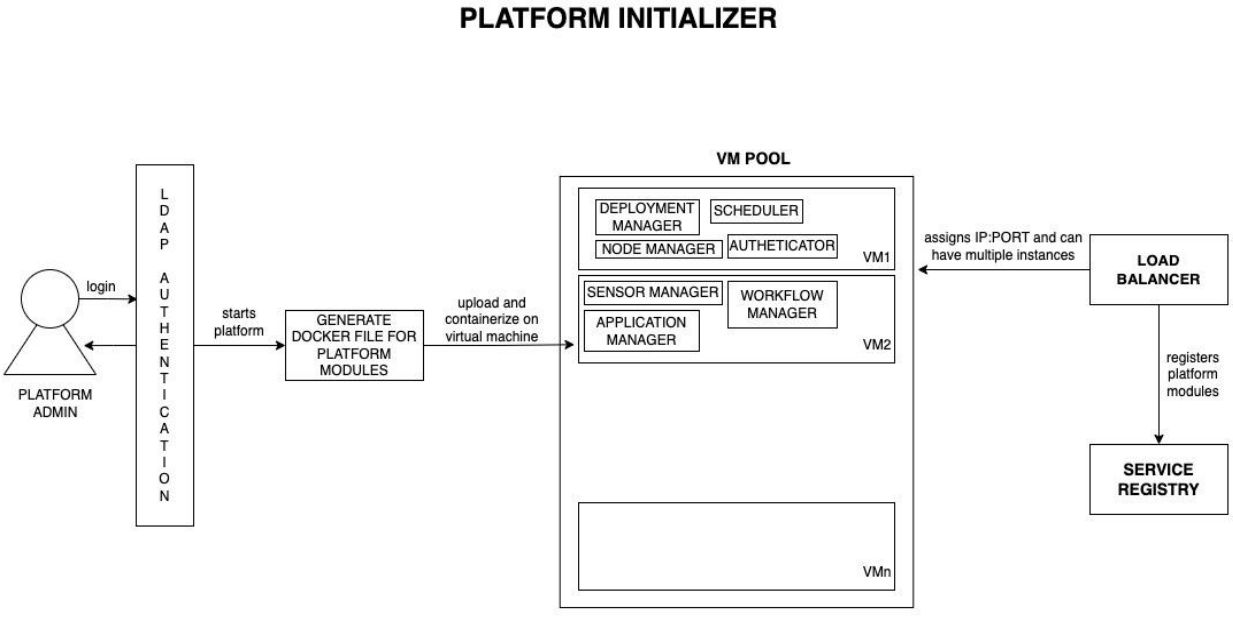
4) **Workflow Manager**
   a) It triggers all the modules one by one by just initiating the starting module.
   b) It used the workflowmanager.json file to generate the code for automatically starting the flow.

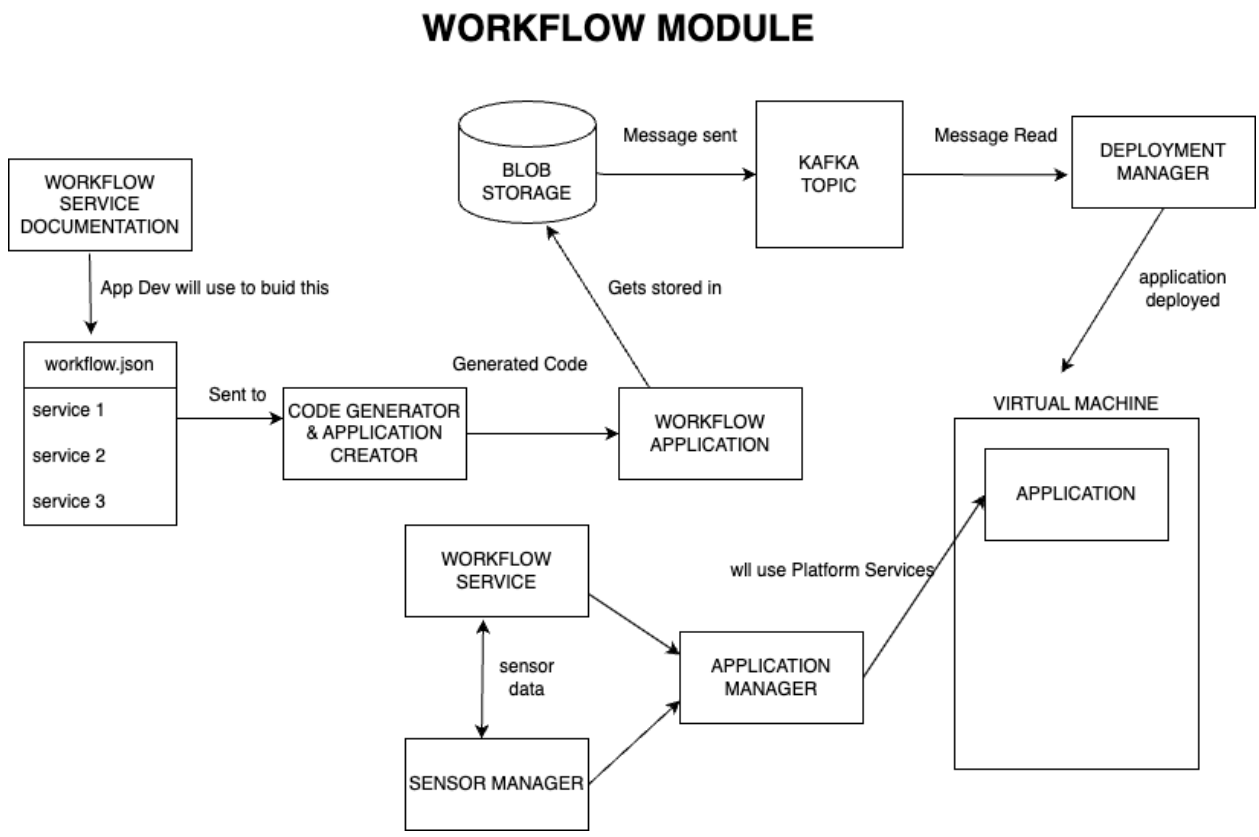
Block Diagram:-



Sensor Manager







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.

List of SubSystems:-

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

List of Services:-

- **User Interface**
    - First, each type of stakeholder (Platform admin, Application developer, End-user) will come and log in with their credentials.
    - **Platform Admin** can see platform status (it is on/off) if is it off then platform admin can start the platform and can see all uploaded apps, and deployed apps.
    - It can also view all module's status whether it is in the running stage or not and their logs.
    - It can check the health, CPU utilization, Disk utilization, and memory utilization of each virtual machine and module.
    - It can view each sensor's live data and load balancer status.

    - **Application Developers** can upload any new sample apps and workflows through this platform.
    - Can see their uploaded and deployed apps and workflows.

- Can view scheduled apps and deployment progress status.

- **End Users** can deploy and schedule uploaded apps.
- Can see all uploaded and deployed apps.
- It can also view scheduled apps and deployment progress status.

- **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.

- **Platform Initializer**
  - The platform initializer initializes the host machine(virtual machine) so that it fires up the platform services required.
  - The services of the platform initializer are.
    1. Installs the required dependencies on the host machine(virtual machines in our case)
    2. Initialise and start all the platform services(modules and managers)

    The components that will be initialized on the platform are

    ➔ Application Manager

➔ Scheduler
➔ Application Deployer
➔ Node Manager
➔ Deployment manager
➔ Sensor manager
➔ Monitoring and Fault tolerance and so on

- The platform initializer can run all the platform services (or modules) on one single virtual machine or then can be deployed on multiple virtual machines in round robin manner.

- **Workflow Manager**
  - 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 endpoint of the deployed workflow will be given to the Application Developer and can be used in the application for further use.

Interaction with other components

- **User Interface and LDAP** - After submitting their respective credentials, they will be verified by LDAP. If authentication is a successful entity then it will be able to access the platform.
- **User Interface and Application Manager:** The list of services asked by stakeholders will be fulfilled by the application manager like a list of uploaded apps, deployed apps, VM's health, etc.
- **Application Manager and Sensor Manager**: The 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 application information and passed to the scheduler in the form of metadata with the application's other information.

- **Sensor Manager and Node Manager**: If there is an application instance running on a node, it may need to establish communication with the sensor manager to exchange data with one or more sensors. This could involve sending or receiving data streams.
- **Platform Initializer and All Modules**: The platform initializer will initialize all the modules to get them up and running.
- **Workflow Manager and Deployment Manager**: The code generated by the Workflow manager using the code generator module is accessed by the deployment manager for deploying it.