IAS
Group 2

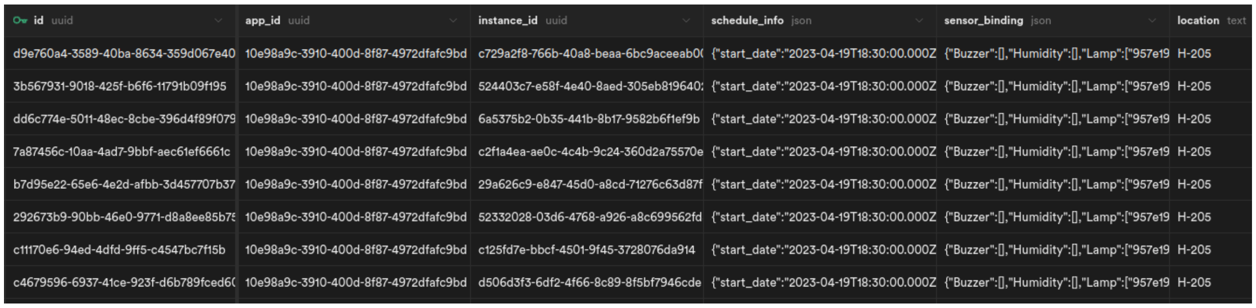
App packaging:

Steps:
- End user logs in
- They see available apps on the platform
- Select an app
- Select required sensors, validated by config.json
- Select scheduling properties
- Selected sensors gets stored in the db for this app instance (as multiple instances of the same app can be run)
- Platform runs/schedules it

The app developer submits the app.zip(contains the code along with bindings.json) and the config.json.

Package files:
- App.zip
- Config.json
- Sched_info.json (when end user gives scheduling info)

Workflow when end user triggers the application:



id uuid	app_id uuid	instance_id uuid	schedule_info json	sensor_binding json	location text
d9e760a4-3589-40ba-8634-359d067e40	10e98a9c-3910-400d-8f87-4972dfafc9bd	c729a2f8-766b-40a8-beaa-6bc9aceeab0	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
3b567931-9018-425f-b6f6-11791b09f195	10e98a9c-3910-400d-8f87-4972dfafc9bd	524403c7-e58f-4e40-8aed-305eb819640	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
dd6c774e-5011-48ec-8cbe-396d4f89f079	10e98a9c-3910-400d-8f87-4972dfafc9bd	6a5375b2-0b35-441b-8b17-9582b6f1ef9b	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
7a87456c-10aa-4ad7-9bbf-aec61ef6661c	10e98a9c-3910-400d-8f87-4972dfafc9bd	c2f1a4ea-ae0c-4c4b-9c24-360d2a75570e	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
b7d95e22-65e6-4e2d-afbb-3d457707b37	10e98a9c-3910-400d-8f87-4972dfafc9bd	29a626c9-e847-45d0-a8cd-71276c63d87f	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
292673b9-90bb-46e0-9771-d8a8ee85b75	10e98a9c-3910-400d-8f87-4972dfafc9bd	52332028-03d6-4768-a926-a8c699562fd	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
c11170e6-94ed-4dfd-9ff5-c4547bc7f15b	10e98a9c-3910-400d-8f87-4972dfafc9bd	c125fd7e-bbcf-4501-9f45-3728076da914	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205
c4679596-6937-41ce-923f-d6b789fced6	10e98a9c-3910-400d-8f87-4972dfafc9bd	d506d3f3-6df2-4f66-8c89-8f5bf7946cde	{"start_date":"2023-04-19T18:30:00.000Z	{"Buzzer":[],"Humidity":[],"Lamp":["957e19	H-205

1. The end user visits the platform and is presented with a list of applications available on the platform. The user then chooses an application and decides to run it. The user must first choose the location where they wish to run the application (there will be a geographical constraint - IIITH campus). The user then chooses if they want to schedule the application to run at a specific time or launch it now.
2. When the user selects the location, the sensor binding stage comes into play. When the application is launched by the deployer in the further steps and an instance of the application is created, the sensor instances are also initialized corresponding to the application instance.

3. The scheduling information is then sent to the platform manager and the same procedure is followed as the upload of an application. The platform manager forwards the scheduling json file to the scheduler.
4. The scheduler parses the json file and sends the app ID with the start flag to the deployer.
5. On receiving the app ID from the scheduler via kafka, the deployer requests the platform manager for the code corresponding to the ID. An instance of the application is created and stored in the Deployer DB and the state of the instance is set to init. The flow is then transferred to the load balancer, who is responsible for monitoring the worker VMs, where the applications are supposed to run. The DB is also updated with a pending state. The applications are containerized and run on the worker VM and on completion, the status is updated to completed.
6. In case the code accesses any sensor data through an API call, the request is routed to the sensor manager, who then fetches the relevant data from the OneM2M interface and provides it to the respective VM. This communication will take place via a kafka topic.
7. The application can also make use of the trigger manager to take action on their binded sensor. The trigger manager picks up this command and fetches the action file of the specific sensor. A request is then generated to OneM2M for a trigger response on a sensor to be taken place.