Data Model

Plots Table

Id	Primary Key	ID auto generated sequence
Plot_name	String	UNIQUE , Not Nullable
Width	Integer	
Length	Integer	
xCoordinate	Integer	
yCoordinate	Integer	
lasTriggered	String	Timestamp of the last successful time a sensor irrigated the plot

- This Table is the main Plots table where a plot is added and configured
- With every plot added we need to verify the coordinates that it is not mapped to any other plot (not implemented in the code)
- With every successful irrigation, lastTriggered is updated with the Timestamp

Sensors Table

Id	Primary Key	ID auto generated sequence
name	String	
ipAddress	String	UNIQUE, Not Nullable
maxRetries	Integer	Default = 1
status	Enum	ACTIVE/INACTIVE
Plot_id	Integer	FK to Plots.id

- Every sensor is created with a unique IP Address where the sensor will be communicated with through IoT Network
- Every sensor is associated to a plot of land (later can be automated based on the sensor coordinates mapping the plot coordinate)
- Every sensor is having a maxRetries to decrement every time in case the sensor is not reachable and an alarm is sent if this value is reached to zero
- Every sensor is having a status to indicate ACTIVE/INACTIVE

Schedule Table

Id	Primary Key	ID is passed by the User to schedule a Task
cronExpression	String	CronJob Expression for the schedule, Not Nullable
waterAmount //in mL	Integer	Default = 1
Duration // in minutes	Integer	Degault = 1
Plot_id	Integer	FK to Plots.id

- Schedule Table is a wrapper over a cronjob scheduler Pattern
- Every Schedule is mapped to a plot of land indicating the frequency , amount of water , duration
    - All this information above is being sent to the sensor through an internal API when the cronjob of each schedule entry is triggered.

Provided APIs

Plot

GET /api/plot	Return all the available plots	
POST /api/plot	Add a new Plot	Body = Plot
DELETE /api/plot/{plotID}	Delete the Plot ID	Cascade ALL
PUT /api/plot/{plotID}	Update the Plots details	
PUT /api/plot/{plotID}/sensor/{sensorID}	Register Sensor to plot	

Sensor

GET /api/sensor	Return all the available sensors	
POST /api/sensor	Add a new Sensor	Body = Sensor
GET /api/sensor/irrigate/{plotID}/{waterAmount}/{duration}	Internal communication to trigger the irrigation	
DELETE /api/sensor/{sensorID}	Delete the Sensor	

Schedule

GET /api/scheduler	Return all the available schedules	
POST /api/scheduler/schedule	Add a new schedule to a certain plot	Body = Schedule
GET /api/scheduler/unscheduled/{jobId}	Remove the schedule	

Workflow

Initial Database Stage

Some data are pre-initialized to make the trials at ease and exist at *PlotConfig* Class.

- 2 plots are created
- 3 sensors are created and associated to plots

Code Structure

- Controller
    - Controller for all the services
- Repository
    - API layer to communicate with the Database
- Service
    - All Services
- Scheduler
    - Scheduler specific files
- Sensor
    - Sensor specific files
- Plot
    - Plot specific files