Remote Drip Irrigation Management Project Documentation

Project Name	:	Remote Monitoring and Management of Drip Irrigation using Digital Twin Technology.
Faculty Incharge Name	:	C. Pavan Kumar
Student Name	:	Sai Likhita Rage
Roll number	:	20BD1A6642
Class	:	3rd year CSE(AIML)-A

Faculty Signature

Student Signature

Problem Statement: Drip Irrigation is a widely adopted system for delivery of water and other nutrients to crops efficiently. We control the drip irrigation model containing different components such as pump station; control valves, mains and sub-mains etc. in the physical world through a 3D model of the irrigation system in virtual world using Unreal and IOT.

Water is an extremely important part of our lives. Water is a unique and non-substitutable resource. As the foundation of life, water carries multiple values and benefits. It is one of the most important inputs essential for the production of crops. The misuse of water leads to the problems thus rendering agricultural lands unproductive. Hence a proper and economic utilization of water resources for maximum crop production is required.

Enhanced irrigation efficiency is vital to the preservation of water resources, particularly in arid and semi-arid regions that can benefit the most from investment in agricultural water technology. Drip is a micro-irrigation system that has the potential to save water by allowing water to drip slowly and directly to the plant's roots zone, in the right amounts, at the right time, so each plant gets exactly what it needs, when it needs it, to grow optimally.

Need of monitoring and managing drip Irrigation using Digital Twin Technology

It is time consuming and requires manual work and effort for managing the drip system in large farms. Farmers need to manually monitor the condition of the crops, valves, soil moisture content and then manage them accordingly. Instead we implement an application using Unreal and IOT technologies where a user can monitor and manage the condition of the farm efficiently from any place of the world and the process is automated. It is also cost efficient.

Technical Description:

In my application, a 3D model of the drip irrigation system containing valves, mains is created in the Unreal Engine to control the physical model. WebSocket API integrated with Lambda functions is used for two-way communication between physical and virtual model. IOT programming is done which continuously monitors the soil moisture content, state of valves and sends signals to Unreal Engine through WebSocket API, through which the user can control the components.

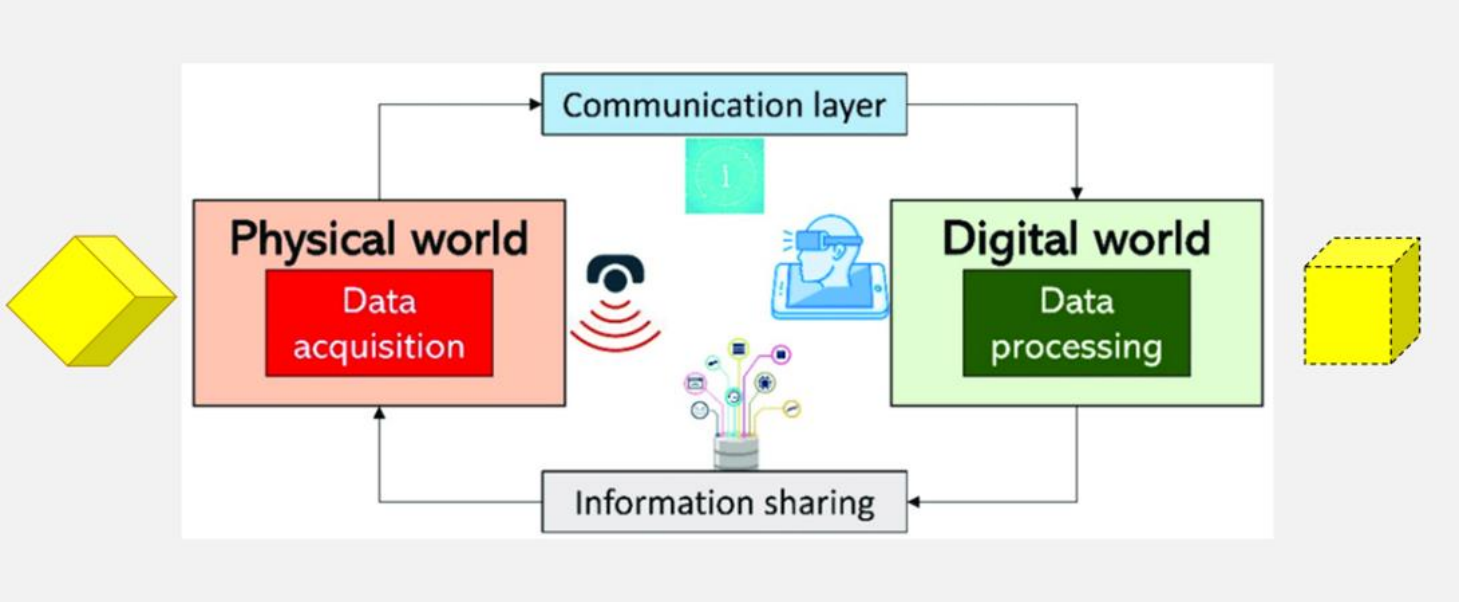
Tech Stack:

- Amazon Web Services

- Unreal Engine

Hardware Components used in the project:

- Microcontroller (ESP32)



Why Unreal for my application?

Unreal Engine is a new age technology that provides 3D computer graphics game engine, which enables to create real time 3D models of the physical world. In my application, I used Unreal Engine to create a virtual model of the drip irrigation system containing valves and soil moisture content indicator. I made use of the WebSockets module provided by Unreal Engine for communicating with the physical world. This socket allows me to send and receive messages from the physical drip model.

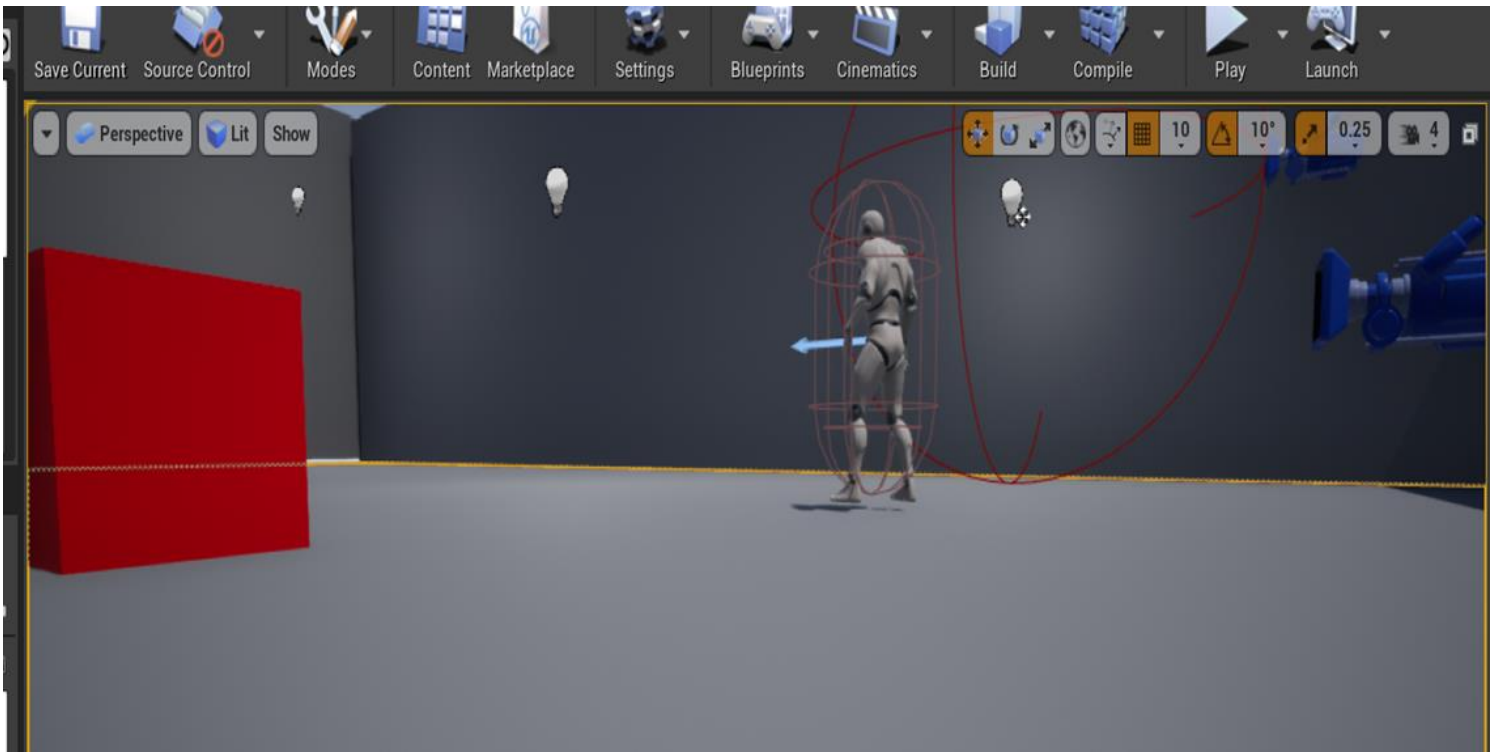
Events used along with the socket in my application are

- OnConnected() – The code in this event runs once the connection is established between Unreal and ESP32. We display the “Successfully Connected” message.
- OnConnectionError() - The code to be run if the connection fails is placed in this event.
- OnClosed() – When the connection to the API is terminated, code in this event runs. ”Connection Closed” is displayed in my application.
- OnMessage() – In my application the websocket receives the message a string. Whenever the unreal Engine receives the string message from physical world, the code in this event gets executed. We display “Received message” and the message on unreal interface.
- OnMessageSent() – This code is executed when the message is sent to the physical world. The sent message is displayed along with “Message Sent”.

In my application,Connect() method is used with websocket object to connect to the server, Send() method is used to send messages from Unreal Engine through websockets. Connection is closed using the close() method.

The physical drip model can be controlled through unreal and also by giving messages in command prompt by connecting to local server using wscat.

Unreal user interface of my application:



In my application, the three bulbs in unreal indicate the three valves in the physical drip model. Red color cube indicates the soil moisture content. Whenever the character in the interface moves near to a bulb and presses a specific key the bulb turns on or off based on its state. As the bulb changes its state of on or off message is sent to the physical world through websocket and the corresponding valve is turned on or off according to the received message.

Amazon Web Services in my application

I used the websocket API of Amazon Web Services for two-way communication between physical drip model and virtual model. I

integrated the predefined websocket routes with lambda functions to handle the requests.

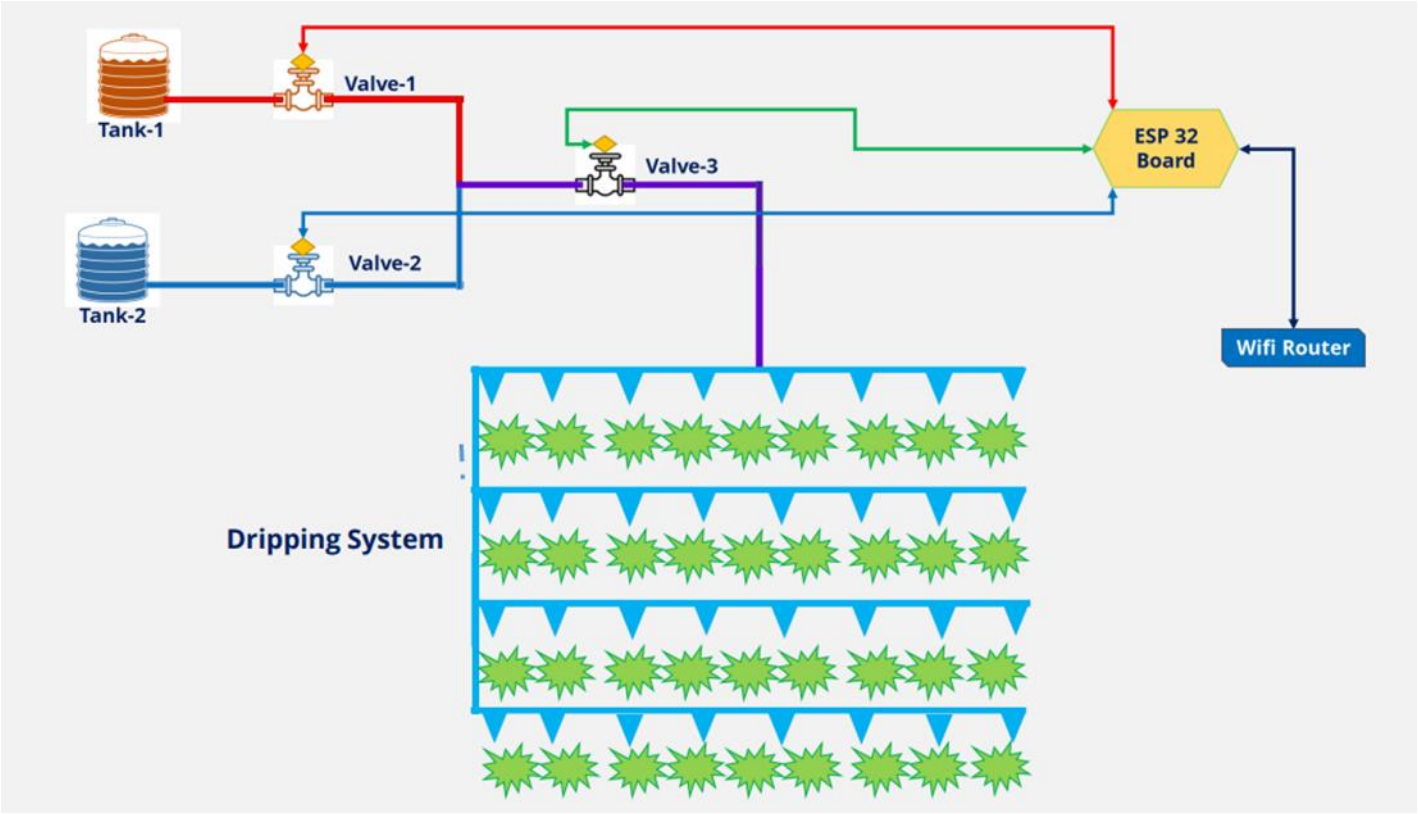
Routes and Lambda functions

- **$connect** — when this route is called, a Lambda function will add the connection id of Unreal Engine, ESP32 to DynamoDB.
- **$disconnect** — when this route is called, a Lambda function will delete the connection id of the connected devices (Unreal Engine, ESP32) from DynamoDB.
- **onMessage** — when this route is called, the message body will be sent to unreal and ESP32.
- **Sendmessage –** I selected the Sendmessage route as a custom route to handle the transfer of messages**.**

In my application my API's route selection expression is `$request.body.action`. So whenever I type the following message {"action": "sendmessage", "message": message to be sent} Sendmessage route is invoked and the corresponding lambda function collects the ids of unreal Engine and ESP32 and sends messages to them.

IOT Programming:

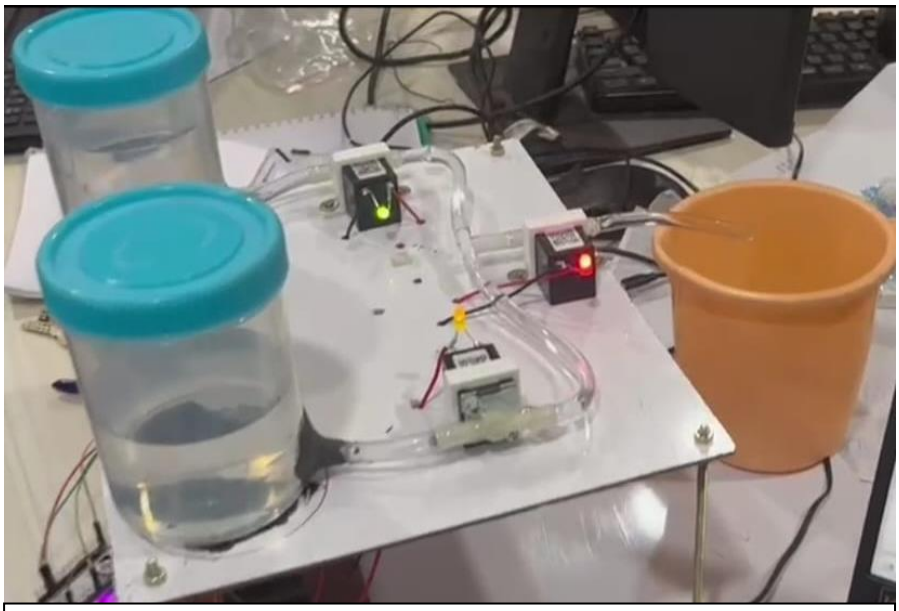
I used ESP32 microcontroller for IOT programming to monitor the physical drip model and condition of the farm and crops.



In my application, there are 3 valves connected to two tanks. Two valves are directly connected to the tanks and the third valve would only allow the flow of water if any of the first and second valves are On. The ESP32 board is connected to wifi and monitors the condition of farm and messages are sent to unreal, according to the requirement valves are turned on. Corresponding LED's glow on ESP32 board, message is sent to unreal and corresponding point lights glow continuously indicating that the valves are on. Soil moisture content is monitored using the soil moisture sensor.

Whenever the value is less than the threshold value, a red LED glows on ESP32 indicating low moisture content value and message is sent to unreal and the colour of the cube changes if the value is below threshold value.

Images of working of application:



Flowing of water through valves in drip model

Measuring soil moisture content, LED glows indicating low moisture value.

Using Digital twin technology simplifies human effort and helps in automation of tasks. Farmers with large acres of farms can easily adopt this method. Instead of going to the farm each time to monitor its condition and manage its requirements farmer can simply remotely monitor and manage it from any part of the world. This application can further be expanded by adding many other components used to manage the farms. Digital twin technology can also further be extended to other domains and applications to automate tasks.