**Introduction**

This Project Use Case is for the mid-term project use case display for Publicis Sapient wherein the participants inculcate their learning and IT skills in developing quick proof of concepts. The candidates are freshers passed out of various engineering branches. In this project plan, they would be introduced to a problem statement in which they are expected to work on problem-solving skills, integrate different toolchains, along with working on observability and monitoring.

**Problem Statement**

You would be working on creating a 2-Tier Application Stack setup involving Postgres, Tomcat, and deploying the application in a containerized environment.

The project utilizes Maven to build the artifact, which is then deployed to the Tomcat server.

In this project, PostgreSQL serves as the chosen database management system.

You would be using Github for code management, keep a tab of the Github practices for push and pull. You can use a single branch or utilize multi-branch deployment.

Service Provisioning includes:

* Apache Tomcat v9 or v10
* PostgreSQL Database

Ensure the firewalls/security groups are configured in a proper manner to allow required ports to be enabled.

The application logic comprises of:

1. Developing a service that will read weather API data from: <https://openweathermap.org/api>
2. URL mapping needs to be done to fetch the following :
3. Display the current weather data for a location

How to get started with Weather API is available at:

<https://openweathermap.org/appid>

You would be writing the logic using Java + SpringBoot + PostgreSQL

Deploy the JAR in a Docker environment and scale the app using 2 nodes of Docker-compose or Kubernetes.

* Deploying the Datadog Agent and instrumenting the application
* For the monitoring of the application, you would be using:
* DataDog Free Account
* Infrastructure monitoring using Infrastructure List [depending on the deployment environment]
* Create dashboard to display the metrics and use the Metrics -> Explorer option to fetch Metrics and time difference
* Monitor CPU and Memory usage
* Monitor Java app using Java APM in DataDog
* Monitor API [if time permits, brownie points 😊]

**Demo:**

You would need to explain the working of the application, any bugs/challenges you came across in the existing code base, how the integration worked. Also, explain the dashboard and the metrics being monitored.

**Outcome**

On working on this Project Case, you will get the following exposure:

1. Working with an API

2. Integrating and monitoring an existing service

3. Demonstrate ability to resolve issues

4. Monitoring

5. Working with 3rd party packages

6. Integration of different DevOps tool chain

7. Code collaboration and management using Github

8. Observability Patterns

9. Development/Update of Code

10. Automation

**Assumptions:**

1. Has access to existing API

2. You have been trained on the technology stack you are using

3. Application deployment using Docker and scaling using Docker-Compose

4. Basic understanding of configuration files

5. Understand the core concepts of Observability, can do fundamental monitoring with DataDog