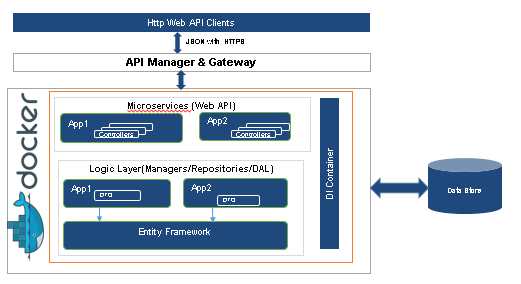
**Clock Calculator Acceptance Criteria**

1. **Deployable code base –**

An Asp.net core web api micro service is created with C# as language. All the code is inside the ClockAngleCalculatorService folder. It has two projects one is service and other is Unit Test project. The project can be opened on VS Code or Visual Studio. Aplication can be debugged both inside the Docker Container or on IIS Web Express.

**Service Architecture –**



**Additional Components –** Added Repository Layer and Base Code for Entity Framework so that data can be persisted in the database.

URL to Test –

The API can be tested using the **Swagger** interface, which has been embedded in the API. Open below url, in the page expand the GET section and then click on Try it out button. Provide the value of hour and min and execute. You will get back the angle as response.

<https://clockanglecalculatorservice.azurewebsites.net/swagger/index.html>

Direct API Test –

[https://clockanglecalculatorservice.azurewebsites.net/api/ClockAngleCalculator /**{hour}/{min}**](https://clockanglecalculatorservice.azurewebsites.net/api/ClockAngleCalculator%20/%7bhour%7d/%7bmin%7d)

<https://clockanglecalculatorservice.azurewebsites.net/api/v1/ClockAngleCalculator/12/15>

<https://clockanglecalculatorservice.azurewebsites.net/api/v1/ClockAngleCalculator/04/13>

1. **Solution deployed to a cloud environment**

In the interest of time, the solution is currently deployed to Azure Web Apps Service. Urls have been shared above.

**Ideal Solution –** Creating an azure app fabric and application to be deployed as a micro service in a docker container. A docker file is already created as part of service solution

1. How will you deploy this solution (in code or as a todo list if time is limited). i.e. how and where will this run? Details of CI/CD pieplines etc.

**Created the YAML for Azure Pipeline. Azure Devops pipeline has been setup and integrated with GIT Repo.**

Below are the Various Tasks configured in CI/CD Pipeline –

**Continuous Integration Pipeline – To be Triggered on code checkin in GIT repo.**

* Nuget Package Restore
* Build Solution
* Run Unit Test
* Run Static Code Analysis/Security Scans [SonarQube, Fortify, WhiteSpace]
* Create Release Artifacts

**Continuous Deployment Pipeline – A Release is created on successful run of CI Pipeline.**

* Various environment to be setup like DEV,TEST STAGING, PROD.
* Perform Integration testing on TEST Environment.
* Pre and Post Approval Gates
* Deploy the code to various environment.

1. How will you manage any infrastructure needed? **ARM Templates**
2. Delivered as a feature branch in the repo fork. **Yes**
3. Any DevOps/Cicd components that would support this feature in a production setting

**Blue Green Deployment**

**Security Scan of code using Fortify and WhiteSource Tools.**

**Use of Kubernetes for container orchestration or use azure service fabric.**

**Use of Istio/Linkered Service Mesh to manage the microservices.**

**Integration of Application Insights.**

**Integration of API Gateway.**

**Using Azure Monitor Service for various telemetry and analysis.**