IIHT

Time To Complete: 10 to 12 hr

Project Management tracker Application

Contents

[1 Problem Statement 2](#_Toc103000532)

[2 PROPOSED APPLICATON WIREFRAME 2](#_Toc103000533)

[3 Tool Chain 3](#_Toc103000534)

[4 Development Flow. 4](#_Toc103000535)

[5 Business Requirements: 4](#_Toc103000536)

[6 Proposed Rest Endpoints to be exposed 7](#_Toc103000537)

[6.1 Rest APIs: 7](#_Toc103000538)

[7 Rubrics/Expected Deliverables 7](#_Toc103000539)

[8 Implementation/Functional Requirements 8](#_Toc103000540)

[8.1 Product and Frameworks: 8](#_Toc103000541)

[8.2 Governance and Tooling: 8](#_Toc103000542)

[8.3 Code Quality/Optimizations 9](#_Toc103000543)

[9 Frontend 9](#_Toc103000544)

[10 Platform 9](#_Toc103000545)

[10.1 Cloud Specific Design 9](#_Toc103000546)

[10.2 Design Specification - 1 9](#_Toc103000547)

[10.3 Design specification - 2 10](#_Toc103000548)

[10.4 Design specification – 3 10](#_Toc103000549)

[10.5 FrontEnd Deployment 10](#_Toc103000550)

[11 Methodology 10](#_Toc103000551)

[11.1 Agile 10](#_Toc103000552)

# Problem Statement

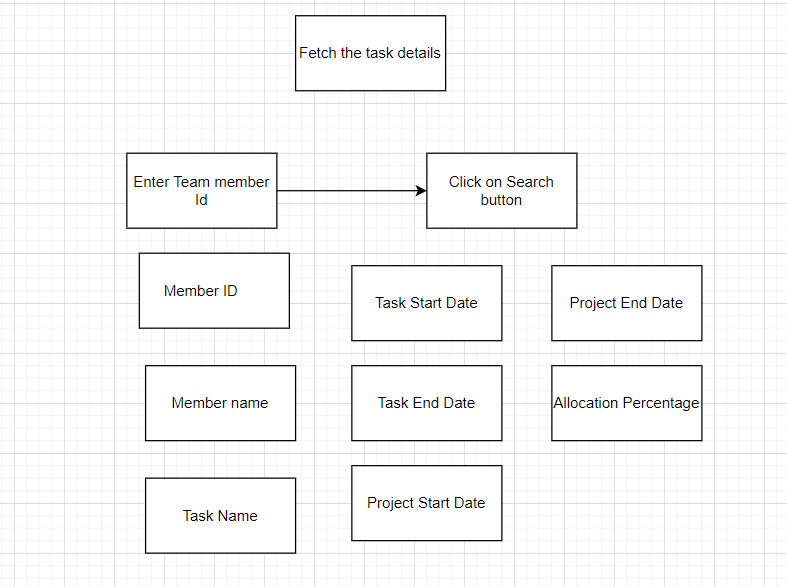
**Project Management Tracker Application** is microservice based Cloud Native Application. The Main purpose of **Project Management Tracker** is to provide the ability to add team members with skill set, assign a task and allocate them to the project:

* Put a new member to the team.
* Assign the task to each member.
* Update the allocation percentage.
* Enlisting of assigned task for each member
* Enlisting of all members from a team along with complete profile details.

The scope includes developing the application using tool chain mentioned below.

# PROPOSED APPLICATON WIREFRAME

Below is the wireframe for reference.



# Tool Chain

|  |  |  |
| --- | --- | --- |
| Competency | Skill | Skill Detail |
| Engineering Mindset | Networking and Content Delivery |  |
|  | Ways of Working |  |
|  | Consulting Mindset |  |
|  | DevOps |  |
|  | Secure Coding |  |
|  | Code Quality |  |
| Programming Languages | Application Language | C# .NET |
| Products & Frameworks | Presentation | Angular/React |
|  |  | Javascript/Typescript |
|  |  | Bootstrap |
|  | Compute & Integration | Asp .NET Core |
|  |  | ELK Stack |
|  |  | Kafka/Rabbit MQ/ Active MQ |
|  |  |  |
|  | Governance & Tooling | Git |
|  |  | Mockito |
|  |  | Jasmine/Jest/Nunit |
|  |  | Protractor/Cypress |
| Engineering Quality |  |  |
| Platform | Cloud Tools | Azure ACI |
|  |  | Azure CosmosDB/SQL DB |
|  |  | Azure Redis Cache/Storage |
|  |  | Azure DevOps/Pipeline |
|  |  | Azure API Gateway |
|  |  | Azure Load Balancer |
|  |  | Azure Notification Hubs |
|  |  | Azure Functions |
|  |  | Azure ACI |
|  |  | Azure CosmosDB/SQL DB |
|  |  | Azure Redis Cache/Storage |
|  |  | Azure ACI |
|  |  | Azure CosmosDB/SQL DB |

# Development Flow.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **MC** | **Competency** | **Section** | **Indicative**  **Mechanism for Evaluation ( Passing score of 60% in each MC)** | **Points to Note** |
| [**Business Requirement**](#_Business-Requirement:) | | | | |  |
| **1** | **Backend** | Rest API, Database, Messaging, Log/Monitoring, Non-functional considerations | [Click here](#_Proposed_Rest_Endpoints) | **Code Submission and Evaluation, Panel Presentation** | It is mandatory to complete this MC with 60% to proceed with the next Frontend MC |
| **2** | **Front End** | Angular/React | [Click here](#_Frontend) | **Code Submission and Evaluation, Panel Presentation** | It is mandatory to complete each MC with 60% to proceed with the next Cloud MC |
| **3** | **Cloud** | Compute, Identity, Compliance, Security and Content Delivery | [Click here](#_Platform) | **Code Submission and Evaluation, Panel Presentation** | It is mandatory to complete this Cloud MC with 60% |

# Business Requirements:

Below are the user stories for the given problem statement

|  |  |  |  |
| --- | --- | --- | --- |
| **User**  **Story #** | **User Story Name** | **User Story** | **Development** |
| US\_01 | Adding new member to the project | As a manager I can add a new member to my project team  *While adding the member, following information is required.*   * Team Member Name * MemberID * Number of years of experience * Skillset * Additional description about current profile * Project start date * Project end date * Allocation percentage   Constraints:   1. Only if the number of experiences is greater than 4, the member can be part of this project 2. Member should possess at least 3 skillsets   Validations:   1. All the fields are mandatory 2. Project end date should be greater than project start date 3. Allocation should be provided as percentage | Only API to be developed |
| US\_02 | Fetching all the team member details | As a manager, I should be able to fetch all my team member details  Considerations:   1. Fetch all the complete profile details of the team members 2. Sort the list in the descending order of number of experiences | Only API to be developed |
| US\_03 | Assigning a task | As a manager, I should be able to assign a new task to each team member  When creating a new task, following information is required:   1. Member Name 2. Member Id 3. Task Name 4. Deliverables 5. Task Start Date 6. Task End Date   Pre-condition: Member Id and Member name should be retrieved from the database  Constraints:   1. If task end date is greater than Project End date, then custom exception should be thrown 2. Task End date should be greater than task start date | Only API to be developed |
| US\_04 | Viewing the task | As a team member, I should be able to view the assigned task  Considerations:   * While fetching the task details the API must return the project details also (Project Start Date, Project End date, Allocation percentage) | API and Frontend to be developed |
| US\_05 | Updating the allocation percentage | As a manager, I should be able to update the allocation percentage  Constraints:   1. If project end date is lesser than the current date, then the allocation percentage must be updated as 0.   If the project end date is greater than the current date, then the allocation percentage must be 100% |  |

# Proposed Rest Endpoints to be exposed

## Rest APIs:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **URL Exposed** | **Purpose** | | /projectmgmt/api/v1/manager/add-member | Adds a new member | | /projectmgmt /api/v1/ manager /list/{memberDetails} | Fetches details of all the team members | |
| |  |  | | --- | --- | | /projectmgmt /api/v1/manager/assign-task | Assigns a task to each member of the team | | /projectmgmt/api/v1/member/list/{memberId}/{taskDetails} | View the assigned task | | /projectmgmt/api/v1/manager /update/{allocationPercentage} | Updates the allocation percentage | |

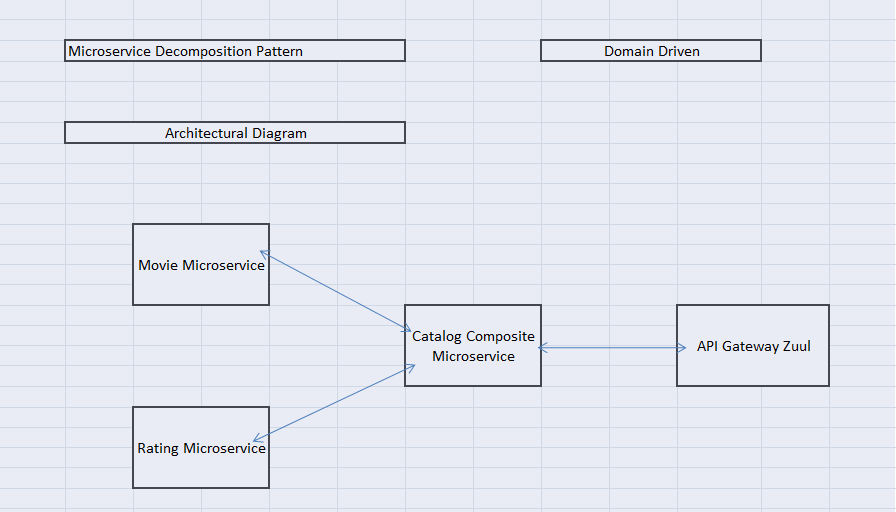
# Rubrics/Expected Deliverables

**Architecture definition**

You are expected to share a design document describing the below. It may contain diagram, flowcharts etc. You may use a presentation tool like PowerPoint or Word Document or simple text file.

* The Design approach should follow SOLID principles.
* Identify the design patterns and application of the same should be clearly stated in the document.
* Architectural assumptions, dependencies, and risks should be clearly stated with mitigation for the same.
* Capture how the Non-Functional requirements like Security, Performance, Availability, Scalability and Resilient are achieved.

**A Reference representation can be as follows:**

****

# Implementation/Functional Requirements

## Product and Frameworks:

1. **Compute and Integration**

Develop the backend application as a microservice architecture. (Implementation as follows)

* + 1. Identify the best decomposition pattern and create microservice based on that (mention the architecture of same in design document)
    2. Identify the best Database Deployment pattern for use case and implement it (mention the same in design document)
    3. Integrate a message broker in your microservice (Kafka, RabbitMQ or ActiveMQ) to implement CQRS pattern. Implement it for assigning a task (User-Story 3: Command Part) and getting the assigned task (User-Story-4: Query part)
    4. Use any one of the Creational Design patterns for composing the model object to be sent back as response on following endpoint:

/projectmgmt/api/v1/member/list/{memberId}/{taskDetails}

(Fetches details of all task assigned for the respective team member. Mention the patterns used in design document and also specify the reason for selecting the one)

* + 1. Identify and implement the best possible use of .NET Core collections and stream APIs
    2. Optimize you REST endpoints to allow filtering, sorting, and pagination.
    3. Document REST endpoints with OpenAPI or Swagger
    4. Expose all rest Endpoints using a common API Gateway.
    5. If any inter microservice call is required, use feign client.
    6. Implement service discovery and circuit breaker pattern in microservice architecture.
    7. Implement at least 2 Security OWASP recommendations in your .NET Core applications.

## Governance and Tooling:

* + - 1. Follow the practise of Creating Testable Component
      2. Configure your frontend application to implement End-To-End Testing using either Protractor or Cypress.
      3. Test suites must contain exception situation testing.
      4. Generate the Code Coverage report of the same.
      5. Implement logging in your end-to-end testing and report the same using either ELK stack.
      6. Perform Regression testing and develop performance test suite
      7. Perform unit and integration testing of your application and do proper CI/CD

## Code Quality/Optimizations

1. Associates should have written clean code that is readable.
2. Associates need to follow SOLID programming principles.

# Frontend

Develop the frontend for User Story 4. (Implementation as follows)

1. Implement using either Angular or React.
2. Implement any one of the Gang of four Patterns to compose data using typescript before presenting the same on UI.
3. Implement at least one approach for UI performance consideration.
4. Identify and Implement client-side Optimization Techniques for Bootstrap.
5. Implement the prevention of XSS cross-site security threats for frontend application.
6. Implement using proper SOLID design principles
7. Perform unit and integration testing for the front-end application

# Platform

## Compute

Use User Story-4 from the Business Requirements to implement the below.

* + - 1. Use ACI CLI (as an alternative to Azure Management Console) for container management and deployment of microservices. You should be able to explain and demonstrate the same in interview.
      2. Use NoSQL instance of Azure CosmosDB/(SQL) as a database for the Market Application

## Compute, Identity & Compliance, Security& Content Delivery

1. Use Azure Functions and Azure CosmosDB to build a backend process for handling requests for Market App.
2. Orchestrate containers with ACI to run the Microservices
3. Use Azure API Gateway to expose the built Azure functions to be accessible on public internet.
4. Use Azure LB to configure the load balancing of the instances.
5. Configure Azure Notifications to issue messages whenever Auto Scaling is launching or terminating the Container Instances in the Auto Scaling group.
6. Use Azure Security Centre for authentication.
7. Use Azure AD to Sign Up the user for secured operations.
8. Configure Azure Monitor to receive, maintain and expose metrics of all the write operations (Adding company and new stock price details)

Note : Minimum 3 API’s to be hosted in cloud

# Methodology

## Agile

1. As an application developer, use project management tool along to update progress as you start implementing solution.
2. As an application developer, the scope of discussion with mentor is limited to:
   1. Q/A
   2. New Ideas, New feature implementations and estimation.
   3. Any development related challenges
   4. Skill Gaps
   5. Any other pointers key to UI/UX and Middleware Development

# Methodology

## Agile

1. As an application developer, use project management tool along to update progress as you start implementing solution.
2. As an application developer, the scope of discussion with mentor is limited to:
   1. Q/A
   2. New Ideas, New feature implementations and estimation.
   3. Any development related challenges
   4. Skill Gaps
   5. Any other pointers key to UI/UX and Middleware Development