Full Stack Developer / Back end Developer- .Net

Background

ABC company need a ticket-based support system that can be assigned tickets to each project. They have different customers with many projects. Initially they need to design a web API for accessing web-based system.
You need to design and implement a Web API for achieving following requirements.

Tech Stack

BACKEND/API

- .NET Core 6
- REST-API
- GitHub
- SQL Server

Note: You can use SQL Server Stored Procedures if required. (Or Entity Framework core)

API documentation in Swagger

Note: This assignment contains both Front end and Back-end assignment work. You can select it according to your preference and complete the assignment.

If you prefer full stack complete the full assignment. (Complete within 3 days)

If you prefer back-end, you can skip the front-end part. (Complete within 2 days)

Security

The application consists of a backend that accesses input and output data and a frontend that should be secured by JWT-authentication towards users that are setup in the database.

JWT Authentication is preferred in this implementation.

There are 3 types of roles as below,

One role for one user as below,

SUPERADMIN – Create customers and projects. And all access.

ADMIN – Can access all projects.

Normal – Only have explicit rights within projects.

Note: You need to validate authentication according to the above roles.

Overview of the basic concepts

CUSTOMER

A customer can have multiple users and projects. Customer directly connects with the project.

USER

A user can be Admin, Superadmin or Normal.
At least one project needs to be assigned to a **Normal** user. But one "**Normal**" user can be assigned to many projects.
Admin users can access all projects.
Superadmin can setup new customers and projects.

PROJECT

A project must be assigned to "**Normal**" users. (You need to validate assigning other users)
Projects can have multiple tickets.

TICKET

A ticket contains information about a single issue. A ticket can have a status (open/pending/complete).
Ticket can be assigned to any user if required.

End Points

CUSTOMER

- GET /customer – Returns all customers
- GET /customer/{id} – Returns one customer by id
- POST /customer – Add new customer
- PUT /customer – Update existing customer
- DELETE /customer/{id} – Delete customer by id

(In this test you can manually add customer into database table)

Auth

• POST /Login – Login user
• POST /Logout – Logout user

USER

• GET /user – Returns all users ->Filters: customerId, projectId
• GET /user/{id} – Returns one user by id
• POST /user – Add new user
• PUT /user – Update existing user
• DELETE /user/{id} – Delete user by id

PROJECT

• GET /project – Returns all projects -> Filters: customerId, userId
By default it should be return all projects i.e. . {yoururl}/api/v1/Project
But if you add filters, you need to filter it.
(e.g. {yoururl}/api/v1/Project?CustomerId=1&UserId=2)

• GET /project/{id} – Returns one project by id
• POST /project – Add new project
• PUT /project – Update existing project
• DELETE /project/{id} – Delete project by id
• POST /project/user – Add normal user to project
• DELETE /project/user/{userId} – Remove user from project

TICKET

• GET /ticket – Returns all tickets →Filters: customerId, projectId, userId, status
• GET /ticket/{id} – Returns one ticket by id
• POST /ticket – Add new ticket
• PUT /ticket – Update existing ticket
• DELETE /ticket/{id} – Delete ticket by id

FRONTEND

You need to implement basic client application from one of below framework.

• React / React Native

We expect basic functions of the following screens.

- Login
- User (List/Add)
- Project (List with filters /Add/Edit)
- Ticket (List with filters/ Add/Edit)

- Add **normal** users to Project.

You need to implement the above highlighted API end points.
But you can share your completed API end points and frontend code within 3 or 2 days. (No need to complete everything)

The following outputs need to be required.

1. Database diagram and database backup (Or Data script)
2. API and frontend source code in github (public access) location. (You need to add a readme file to how to set up it. (E.g., connectionString)
3. If you have a problem on github, you can share any other public location. (e.g. Google drive)
4. Brief explanation about your architecture if required (e.g., diagram)
5. You can add all the document in one folder in the source code. (name= document)

Final output needs to be sent within **3 or 2 days** for the email below.
Email = careers@suntechit.in

Emai Subject = **Assignment – Full Stack .Net / Back End .Net**

Please mention your name and source code location in the email body.