Design document for synoptic project

By Pavel Vjalicin

Contents

[Design 3](#_Toc14186816)

[Sequence diagrams 3](#_Toc14186817)

[Use case 1: using REST API with already registered card. 4](#_Toc14186818)

[Use case 2: using REST API with not registered card. 5](#_Toc14186819)

[Install / Configure 6](#_Toc14186820)

[Prerequisites 6](#_Toc14186821)

[Compiling 6](#_Toc14186822)

[Data Structures 7](#_Toc14186823)

[Project Structure 7](#_Toc14186824)

[REST API Endpoints 7](#_Toc14186825)

# Design

The following design was created based on the specifications provided in the SD Project E Membership System v1.2.pdf file.

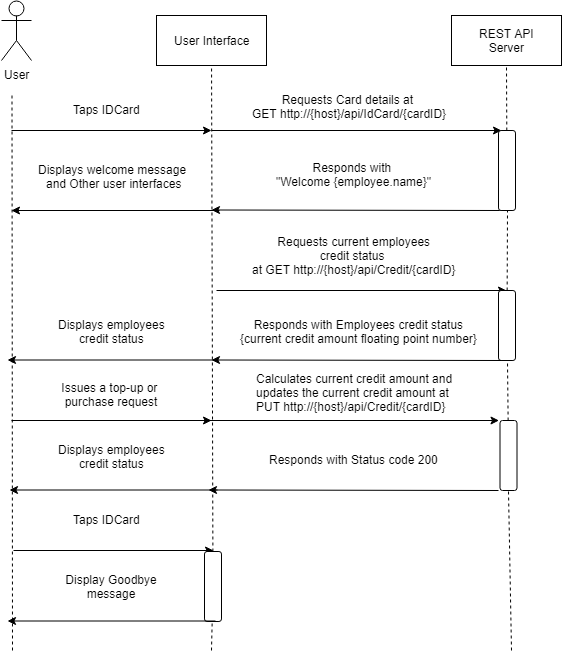
My task was to design a REST API for managing membership cards and here is what I came up with.

## Sequence diagrams

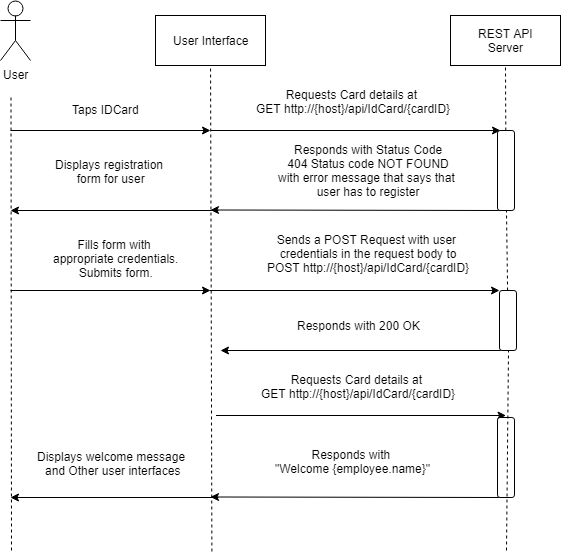
Based on the specifications provided I have created two UML sequence diagrams for different use cases.

Those diagrams display an abstract idea of how the REST API should be used in the context of the whole application.

### Use case 1: using REST API with already registered card.



### Use case 2: using REST API with not registered card.



After registration of a card is finished the application should proceed as usual (as displayed in Use case 1).

# Install / Configure

The REST API server was made by using ASP.NET Core 2.2 with C#.

This particular REST API server was design according to design principles of Microsoft.

The source code of the application is provided in the “qa\_synoptic\_project” folder or can be accessed from a github repository: <https://github.com/PavelVjalicin/qa_synoptic_project>

## Prerequisites

Supported OS.

Official Microsoft reference for OS support for ASP.NET CORE 2.2: <https://github.com/dotnet/core/blob/master/release-notes/2.2/2.2-supported-os.md>

Appropriate dotnet SDK for .NET Core 2.2 installed on the machine.

Link to SDK: <https://dotnet.microsoft.com/download>

## Compiling

This project was not designed to run in production environment.

The specifications did not provide a clear idea of a development environment setup. To build an executable version of the project some project configuration changes will have to be made.

Notes from Microsoft about different environment settings:

<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/environments?view=aspnetcore-2.2>

Notes from Microsoft explaining how to compile an executable file for different operating systems:

<https://docs.microsoft.com/en-us/dotnet/core/rid-catalog>

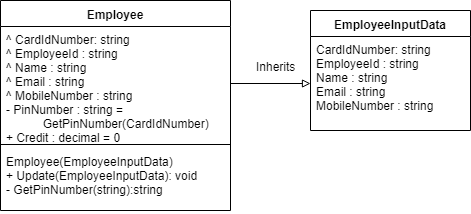
To run the project in development mode issue the following command in the command prompt from “qa\_synoptic\_project/FirstCateringLtd.BackService/” location: **dotnet run**

Here is a video by Microsoft employee on how to use and setup ASP.NET CORE REST API server and more in detail: <https://www.youtube.com/watch?v=--lYHxrsLsc>

After issuing the following command the server will be run on localhost with a random port assigned to it. The address of the server will be display in the console.

## Data Structures

The following data structures were created for this project:



C# implementations of those models can be found in “qa\_synoptic\_project/FirstCateringLtd.BackService/Models.Employee.cs”

We user the Employee class as our Model that we store in database and use EmployeeInputData to communicate with our database through REST API.

Database that is used has to have an appropriate table called Employee with all of the above fields in it. The current implementation has a built in database that is generated by ASP.NET Core framework. ASP.NET Core has multiple supported database types in this instance we use SQLite implementation.

The database generated is currently called CateringDatabase.db. The name of database can be changed from the Startup.cs in the FirstCateringLtd.BackService folder.

Currently database migrations are not supported. After changing any of the models to run project successfully the CateringDatabase.db file has to be deleted.

Here is a list of supported database types provided by Microsoft:

<https://docs.microsoft.com/en-us/ef/core/providers/>

## Project Structure

## REST API Endpoints