Vaccination Registration

Introduction

You agreed to help a local doctor by creating a website for people who want to register for a **Covid-19 vaccination**.

The doctor has sent out letters to interested patients. Each letter contained a **personal PIN code** for the patient. Patients who would like to arrange a vaccination date should go the website you have to implement, enter their **Social Security Number** (short *SSN*) and their PIN.

If SSN and PIN are correct, they can **pick a date**. The doctor will offer vaccinations **between December 1st and 20th 2021**.

Once they picked a date, they can select a **time slot**. The doctor offers time slots based on the following rules:

- The doctor starts at 8am and has to leave at 11am.
- The doctor wants to have enough time for each patient, so he **reserves 15 minutes for each patient** registered for a vaccination. As a result, **12 time slots** should be available on each day:
  - 8:00am
  - 8:15am
  - 8:30am
  - ...
  - 10:30am
  - 10:45am
- Once a patient has registered for a time slot, no other patient can take the time slot.

Data Model

Your software must store data about **registrations** and **vaccinations**.

> You can find the skeleton project in the attached ZIP-File.

- **Registration** means that the patient has shown interest in the vaccination. Therefore, the doctor has sent the patient a PIN code. For each registration, you have to store the following properties:
  - Social Security Number
  - PIN code
  - First name
  - Last name

  - Optional reference to the vaccination appointment, if the patient has already created one.

- *Vaccinations* are appointments for a vaccination with date and time slot. For each vaccination, you have to store the following properties:

  - Vaccination appointment's date and time
  - Mandatory reference to the registration that the appointment is based on. It must not be possible to create a vaccination appointment without a registration.

- Make sure to remove the hard coded location of the `vaccinations.db` database file and put the location in `appsettings.json`.

Backend

> Your code **must** compile without errors.

- Import registrations from a JSON file (`registrations.json`) (see also How to serialize and deserialize JSON). You can trigger the import by starting the ASP.NET Core application with the command-line arguments `import registrations.json`.

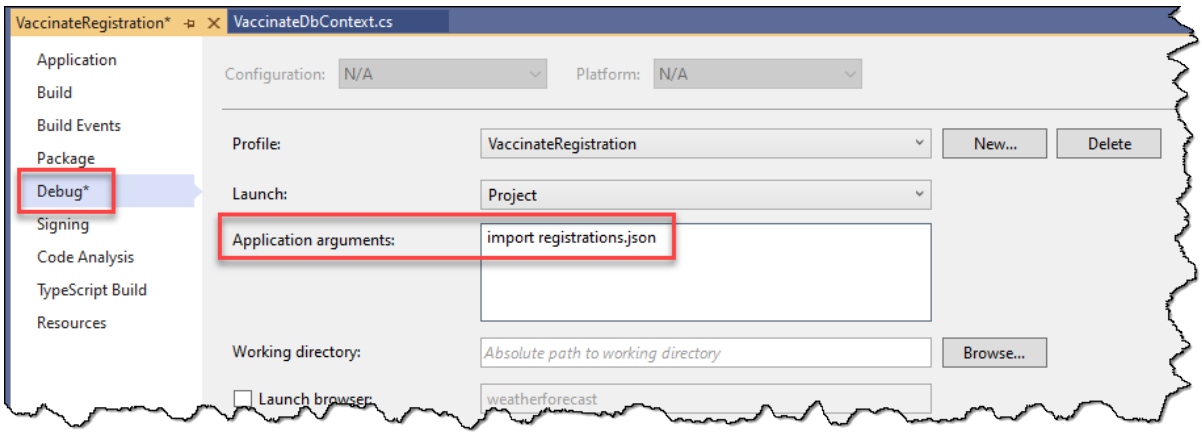

Figure 1: Debug with arguments

Add the required code `Program.cs`. Before importing data, make sure to wipe any existing data from the database.

> **HINT**: You can map model fields to JSON properties like this (`Registrations.cs`):

```
public class Registration
{
    ...
```

```
    [JsonPropertyName("ssn")]
    public long SocialSecurityNumber { get; set; }
    ...
}
```

- Get all available time slots for a given day. You have to calculate all available time slots as described above and remove all time slots that have already been taken.

  You **must not** hard-code all 12 available daily time slots in C#. Use any kind of C# logic (e.g. loop, Linq) to generate the time slots algorithmically.

- Use **services** to access the database layer. Use these services in the controllers.
- Use **DTOs** to exchange information with the frontend.
- Implement a **RESTful API** as required by the frontend.

Frontend

Your final application could look like this:

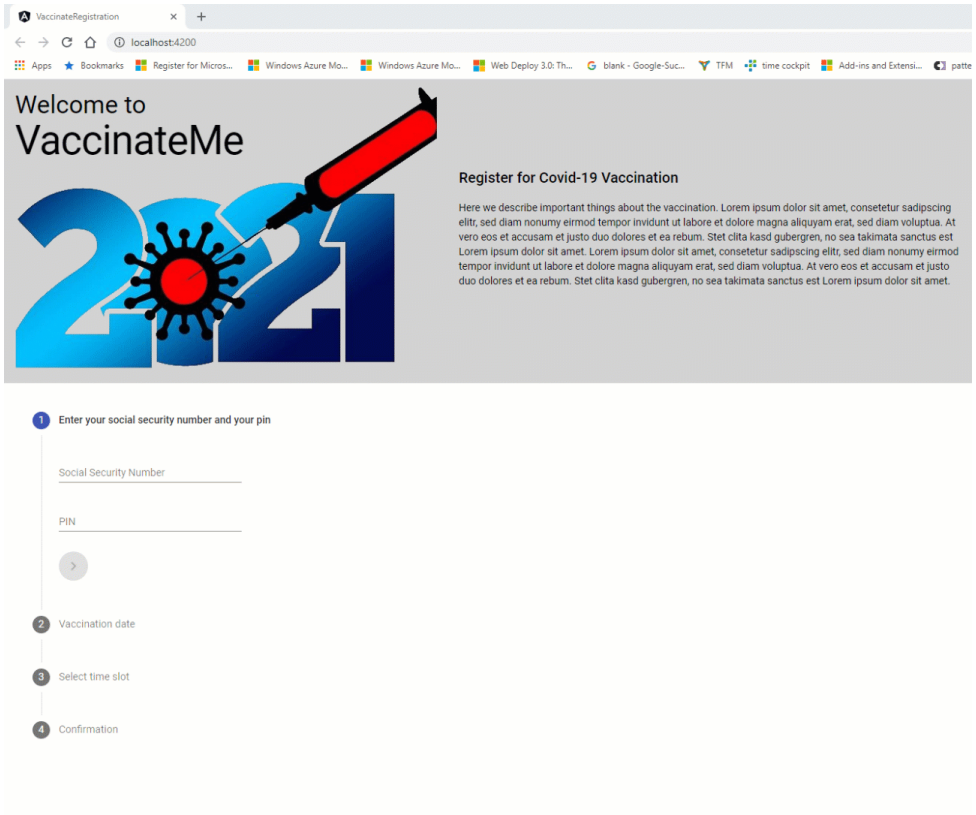

Figure 2: Angular UI

- Implement the frontend using `Angular`.
- Use **DTOs** to exchange information with the backend.
- Use **services** to consume the WEB-APIs.
- Use Angular Reactive Forms to input the SSN and PIN.
- Use Form Validation to check the SSN and PIN.
  - Make sure both fields are **required**.
  - The SSN has **exactly 10 digits**. The PIN has **exactly 6 digits**. Implement Custom Validators to check those conditions.
- Implement some kind of **wizard** functionality with the following steps:
  - Let the user enter SSN and PIN.
  - If they match, **display the first- and lastname of the patient** and the patient can select a date.
  - If a valid date ist selected (between December 1st and 20th), let the user select a timeslot. Use **Angular Form Validation** to check the date.
  - If everything is OK, show a confirmation message.

Technical Tips

- C# struct for handling date and time: `DateTime`
- How to create a `DateTime` instance
- `DateTime` has various methods to add hours (`AddHours`), minutes (`AddMinutes`), etc.