### MANDATORY REQUIREMENTS

The code of the completed **task should be**:

* Well structured
* Easy to read
* Contain the necessary comments

The program you have written **must be a complete program product**, i.e. should be easy to install, provide for the handling of non-standard situations, be resistant to incorrect user actions, etc.

### TECHNOLOGY REQUIREMENTS

Tasks should be completed:

- On .Net or .Net Core (EntityFramework / EntityFramework Core, Web API)

- With any DBMS (MS SQL Server, PostgreSQL)

- With Swagger for automated API documentation (<https://swagger.io>)

- Uploaded to the GitHub

- Using English to write comments and descriptions of classes, fields, etc

Non-functional requirements:

- Three-level project architecture (data access level, logic level, representation)

- When using third-party frameworks and packages - they must be publicly available

- It is recommended to cover the logic level with unit tests

### REQUIREMENTS FOR PRESENTATION FORMAT

The completed task should be uploaded to GitHub and include:

1. **Mandatory:** provide all source files along with the project files

2. **Mandatory:** provide a description of the system configuration, startup process in the explanatory note

### WILL BE A PLUS

- README.md added

- Applying design patterns

- Deploy and provide public access to the application

- Using Docker

- Textual description of the product, technologies and templates used, instructions for use

# Task: Implement Web API for entering project data into the database (task tracker)

You need to implement task storage by projects. “Task” – is an instance which contains at least 3 fields listed below:

1. Id

2. Task name

3. Task description

Solution should provide an ability to easily add new fields to the Task entity.

Each task should be a part of only one project. Project – is an entity which contains name, id (and also keeps Tasks entities).

The program must be a Web API.

Functional requirements:

- Ability to create / view / edit / delete information about projects

- Ability to create / view / edit / delete task information

- Ability to add and remove tasks from a project (one project can contain several tasks)

- Ability to view all tasks in the project

- WIll be a plus to have an ability to filter and sort projects with various methods (start at, end at, range, exact value, etc.) and by various fields (start date, priority, etc.)

Project information that should be stored:

- the name of the project

- project start date

- project completion date

- the current status of the project (enum: NotStarted, Active, Completed)

- priority (int)

Task information that should be stored:

- task name

- task status (enum: ToDo / InProgress / Done)

- description

- priority (int)