# Candidate Test Task

Task: Job candidate hub API
Main tech stack: .NET

Publication Date: April 2024

NOTE: Don’t share this document with any 3rd party, don’t publish it on the Internet!

## 1 Description

Please develop a web application with API for storing information about job candidates. The application should contain just one API endpoint which in future will be used by other application(s) with UI. The endpoint should create, or update candidate contact information at the storage. In long term perspective itis supposed to store information of dozens of thousands of candidates in the system. It does not mean that the application should be able to store that volume of data right now, but it should have potential for such extension.

If something in this task is unclear just assume the answer and provide the assumption as the taskartifact.

## 2 Functional requirements

The application should provide REST API endpoint for adding or updating candidate information. Email of the candidate should be used as their unique identifier. If the candidate profile already exists in the system, it should be updated, if not - created. The endpoint should expect the following information:

- * First name
- * Last name
- Phone number
- * Email
- Time interval when it’s better to call (in case a call is needed)
- LinkedIn profile URL
- GitHub profile URL
- * Free text comment.

Required fields have asterisk (*) in the beginning.

All the data should be stored in a SQL DB of your choice, but in future the storage may be migrated to another type of DB, so please keep this in mind when designing the application. It should be easy to extend the application in future to store data in another storage without reworking the main business logic of the application.

## 3 Technical requirements & recommendations

- .NET stack should be used. The application should have just Rest API, no UI is needed.
- Application should be “self-deploying”. It should be possible just to open solution in Visual Studio, run the app and that’s it, all should work out of the box. It should not be needed to install any Visual Studio extension etc.
- Please write unit test for the code developed. Ensure that the coverage level is at that level that you consider reasonable in real life projects.
- Please think where cashing may be applied. If you have time and ideas, please apply them.
- Use Git as source control system. Try avoiding using one commit for all the changes, commit the changes by some logical pieces.

## 4 Time limits and assessment criteria

- All deliverables described in the next chapter must be provided, if something is missed then it affects the final assessment.
- Source code quality is important. Please concentrate more on quality rather than on functionality. It's better to provide a high-quality solution then cover a couple of additional edge cases.
- Everything that is NOT implemented should be documented and provided together with the task as a list of ways for improvement. Any format may be used for this.
- Appliance of any design pattern would be a big plus. But please do it deliberately, applying a wrong pattern in a wrong place may have negative impact on the task assessment.
- Compliance with task requirements.
- Source code should be provided either as a link to a public (Sigma should have access to it) GitHub project or as a local git repository sent in zip file. GitHub is preferable.
- Development time limit is 8 hours, but by calendar the task should not take more than 4 working days. So, for example, you can allocate to the task for example 2 hours per day for 4 days, or just 8 hours for one day. It's up to you and doesn't affect result of assessment once the task is done within 4 days. If you finish the task earlier, it would be a plus.

## 5 Deliverables (expected artifacts)

Once the task is done you should provide:

- Source code (link to public GitHub repo or zipped local git repo).
- List of ways for improvement if you have any (in any suitable format: a readme file, .docx, wiki page what ever.).
- List of assumptions if you have any (the same as regarding improvements in any suitable format).
- Total time spent for the task (in hours).