**Automizy dev-test tasks**

First of all I spend one day with the analyzing of the project structure what I got. Furthermore I have to obtain some knowledge about grpc, so I am going to read some documentation about it.

**Regular Tasks**

*In chronological order:*

* Install all of the packages.
* Create docker environment for the database.
* Test whether all of the provided codes works well.

**Backend**

*In chronological order:*

* Create project and management data models and add the relevant queries to the migration file. Also create missing .proto files.
* Create grpc services based on the structure what I already have in student grpc service file.
* Create project endpoint urls for the project section.

Management tasks in detail:

* create new logic for the GET methods by student\_id and by project\_id in the service file
* create logic for the PUT method in the service file
* also modify the DELETE method, it will happen by the id
* Write the controller file based on the visioned structure
* Create management endpoint urls according to the points above
* Also create the .proto file based on the future structure

**Frontend tasks**

*In chronological order:*

* Create Project page based on the Student page
* Create management page skeleton in one component
* Create ByProjects BysSudents ProjectsForOneStudent StudentsForProject functions
* Separate Management page into 4 files
* Create AssociateModal component. It will be used for associate projects to user and also for associate students for project.
* Add create associations and delete association logic

**Back to the backend**

* Handle duplicates in associations, projects, students
* Handle student-project deletion – delete the related associations too
* Add search possibilities to the controllers, and the services.. also .proto files should be updated

**Back to the frontend**

* Create axios interceptor
* Restructure the project
* Create search functionalities

**Others in the end**

* refactor the backend: project and student services have similar functionalities..
* Create Postman collections
* Create the endpoint documentations
* Add comments to the code (yes at the end of the project, because on this way I can review all of the lines, and can remove the obsolete things as well)

**Future plans**

* refactor the backend rest api, move some functions to middlewares
* Create one associatemodal component on the frontend with the proper parameters which can be used to build the modal html with option list using checkboxes