The third module will handle the sorting and reverse engineering part of the project.

The projects received from the second module will be sorted based on the following criteria:

- Project popularity - number of stars
- Releases - number of releases and the most recent one
- Contributor popularity - number of followers
- Average - an average computed on all the criteria mentioned above

In the sorting process we will use the APIs provided by GitHub. They will help extract the necessary data.

We will receive a criteria, from the ones listed above, from the first module.

Using https://developer.github.com/v3/activity/starring/, we will receive a list of users that have 'starred' the project and we will sort the list of projects according to the number of users that watch a certain project.

With https://developer.github.com/v3/repos/releases/ we will get a list of releases for each project that will enable us to sort the projects based on the number of releases and the date of the most recent one.

Using https://developer.github.com/v3/projects/collaborators/ and https://developer.github.com/v3/users/followers/ , we will receive the list of collaborators and then, we will get the number of followers for each contributor. We will sort the list of projects according to the total number of followers of each contributor.

For the RE part, we will create class diagrams from the projects received from the crawling component of the project.

Actors:
-Github API for sorting : we will use these APIs in order to sort the projects received from M2
-User : will request for the sorted data or for the reversed data
-DataBase: will be used to obtain and store information both about the projects and about the users (authentication information)

Use cases:

Case 1: Request sorted data

1. The user logs in the application
2. The user requests some data according to some specifications
3. The user requests to sort the projects based on the number of releases/project popularity/collaborator popularity

4. The user gets the requested sorting (realised with the statistics provided by the API)

Case 2: Request class/package diagrams

1. The user is logged in the application and requests reversed data of a project
2. The data is queried
3. The diagrams are generated
4. The reversed data is provided to the user

Members:

- Chiorcea Andra
- Drumea Alexandru-Daniel
- Gospodaru Bogdan
- Hriscu Oana
- Pescaru Victor
- Piclea Cristina
- Poenaru Cosmin-Adrian
- Topciu Liliana-Ilinca