Requirements

Customer's requirements - product requirements document:

- **Presentation** (15 pts). This factor evaluates the team presentation, including how well-organized the presentation is, whether it covers the the important points of the project, the use of visuals, and the effectiveness of any product demonstration.
- **Repository** (15 pts). This factor evaluates the repository history, including whether the history of the development is reflected in the repository history, how branches were used, and the use of pull requests.
- **Functionality** (15 pts). This factor evaluates the functionality implemented by the product. It considers how well the functionality provided matches the requirements documented in the product requirements specification. It also considers whether the implementation of the functionality is adequate to support the intended use of the product. Finally, it considers whether the feature set provided seems reasonable for the size of the project team.
- **Organization** (15 pts). This factor evaluates the organization of the repository, including the structure of the repository and its contents, and whether the repository directories provide descriptions of the contents (e.g. with "README.md" files). The evaluation will consider how easy it is for someone who is not familiar with the project to explore the repository and understand its contents.
- **Methodology** (15 pts). This factor evaluates evidence of applying the chosen methodology during the project. This includes communications, documentation of processes (e.g. backlog decisions, user stories, milestone tracking, issues), communications, and use of tools (e.g. GitHub project boards, issues, etc.).
- **Code** (10 pts). This factor evaluates the project code to ensure that it follows good engineering practices, including the quality of the code, whether the code includes adequate documentation, whether any tests are provided for the code, whether there are scripts for building or packaging the code, and whether there is any external documentation describing the code, its organization, and how to build or deploy it.
- **Documents** (15 pts). This factor evaluates the external project documentation. This includes project organization, requirements, features, functionality interfaces or APIs, design, communications plans, and test plans.

Software requirements specification:

1. Introduction
    a. Purpose: a school project.
    b. Project Scope: a small project developed by a team of 4 NEU students.
2. Overall Description
    a. Product Perspective: an web application which gets your life together and you can organize your daily/weekly/monthly routine.
    b. Product Features: a home page for people to sign-up and login, a "blocks" page to show events, a "create" page to create event, an "edit" page to edit event.
    c. Operating Environment: browser.
    d. User Documentation: project organization, requirements, features, functionality interfaces or APIs, design, communication plans, and test plans.
3. System Features
    a. Document of features.
4. Other Nonfunctional Requirements
    a. Performance Requirements: good performance.
    b. Software Quality Attribute: good performance quality, good feature quality, reliability, conformance, durability, serviceability, aesthetics, perception.
5. Other Requirements
    a. Presentation

This project uses Agile methodology, Agile interleave requirements engineering with design activities in and iterative product development process.

Inception: course need.

Elicitation: a team with size four, an application.

Elaboration: an web application which gets your life together and you can organize your daily/weekly/monthly routine.

Negotiation: doesn't have to be a very complicated application, the point is to understand software engineering.

Specification: design UI, design backend, implement/test, deploy, documentation, present.

Validation: all requirements are clear and doable.

Management: optimized for changes.