Software project delivery  
Generic template

# **How to use the document** *(for students)*

At the end, the team is required to provide a list of functionalities of the project. It needs to connect on the initial Software project description. The project might be steered during its execution. User stories / (non-)functional requirements might change as the team is gaining new knowledge during the implementation as well as be specified more thoroughly. Similar to the “Software project description” document, it must be approved by your supervisor as well as a member of KSVI Project Board and also an opponent. The opponent is appointed by the KSVI Project Board. Completed document (including signatures) should be in the end scanned and sent to the [KSVI Project Board](https://docs.google.com/document/d/1PsY5-6JFFqXjjbUD24gSTWDT-ciCvwS2eLwR4lf3rIY/edit?usp=sharing) <[projekty@ksvi.mff.cuni.cz](mailto:projekty@ksvi.mff.cuni.cz)>; keep your copy while leaving the original with your supervisor.

# Software project details

Name of the project:

<fill in the name here>

Initial Software project description document:

<fill in the link here>

SCM repository of the project:

<fill in the link here, mark whether it is public or private>

Team members submitting the project:

| **Full name** | **Field of study** | **Email** | **Signature** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Main contact

*Appoint a (student) team member who is the main contact of the team; fill in their full name.*

Full name:

Does the team differ from its initial state?

Yes / No

If the team has changed, describe why:

<fill in the reasoning or delete this section>

Date of submission:

<fill in the date you are submitting your project>

# List of user stories supported and corresponding (non-)functional requirements.

*Provide a list of user-stories your project supports along with their (non-)functional requirements. Honor the user-story numbering from the original Software project description document. For each user-story, indicate whether it is original or new one and in case of original, state whether its requirements has been altered or not. Indicate the changes. Provide reasoning for your changes. As a rule of thumb, begin with copy-pasting user stories and requirements from the “Software project description” document and work on it incrementally as the project goes and changes pop out. The example follows.*

1. *(User story)* I want to be able to log into my account.

Type: Original

State: Altered

* 1. ~~Users can log into their account by entering their email and password.~~
  2. ~~Users can log in with their Google accounts.~~
  3. ~~Users can reset their password by clicking on "I forgot my password" and receiving a link to their verified email address.~~
  4. NEW: Users can log in using CUNI CAS credentials.

Reasoning: The project was steered toward the authentication infrastructure used at MFF UK instead of the original Google one as it was decided it will be used by CUNI students only and without the need to do a custom registration into the system.

1. *(User story)* I can register and log in.

Type: New

* 1. The system allows external user to register using their email and password.
  2. The system must allow external users to log into their account by entering their email and password.
  3. The system must allow users to reset their password by clicking on "I forgot my password" and receiving a link to their verified email address.

Reasoning: As the definition of “existing user” changed into “existing CUNI CAS” user, we wanted to provide a way for registration also for external users.

# Documentation

User documentation form:

*You are required to provide a way how to use the software project. The complexity and the form of this document will vary depending on the type of your project. For a game, it can be an in-game tutorial or 1 page explaining the main controls of the game. For a complex information system, the user documentation would include detail steps how to install and run it.*

Programmer documentation form:

*You are required to provide a description of the project architecture. Keep it high-level and provide references to the code that ought to be documented. Again, depending on the project type, this might need to be more detailed, e.g., if you are developing a library, you should document it well.*

# 

# Approval

**Supervisor**:

Place:

Date:

-------------------------------------------

SIGNATURE

**Opponent:**

Place:

Date:

-------------------------------------------

SIGNATURE

**KSVI Project Board Member:**

Place:

Date:

-------------------------------------------

SIGNATURE