High-performance computing system

Conceptual document

Jelle Maas (479146)

Koen Janssen (451394)



# Contents

[**Contents**](#_4h6u2qo84hw8) **1**

[**Version History**](#_34h4zbc8dox0) **2**

[**1 Introduction**](#_ycsyzrd5c1g3) **3**

[**2 CPS**](#_cgn13e8q0wrm) **4**

[2.1 Clarify](#_1fob9te) 5

[2.1.1 Vision](#_wwrqi1khyfbr) 5

[2.1.2 Objective](#_r7uzwzb0aqy4) 5

[2.1.3 Goal](#_bbvq3msul6h7) 5

[2.1.4 Challenge](#_xy8r1467rlcz) 5

[2.1.5 Gather data](#_8blsdua6dxvh) 5

[2.2 Ideate](#_enjq7cgu69su) 6

[2.2.1 Web interface](#_qgfebd4qaqa4) 6

[2.2.1 Public web API](#_k24l5wfhvs6q) 6

[2.2.3 Command-Line access from web browser](#_6hif3gsnpanp) 6

[2.2.4 Network File System](#_15r0thmfzbu3) 6

[2.3 Develop/prototype](#_j38bcxpa9j4k) 7

[2.4 Implement](#_lbs22z5axtcy) 7

[**Conclusion**](#_avog5ixpl50u) **8**

[**Definitions, Acronyms, and Abbreviations**](#_pb2g3jtqn0c2) **9**

[**Bibliography**](#_45upjm7fi258) **10**

# Version History

| Date | Version | Author | Comment |
| --- | --- | --- | --- |
| 15.02.2022 | 0.1 | K. Janssen & J. Maas | Initial setup |
| 21.02.2022 | 0.2 | K. Janssen | Wrote Vision, Objective, Goal, Challenge |
| 21.02.2022 | 0.21 | J. Maas | Wrote data gathered |
| 22.02.2022 | 0.3 | K. Janssen & J. Maas | Wrote together the “Ideate” section and added the appropriate acronyms. |
| 22.02.2022 | 0.31 | J. Maas | Improved the introduction |
| 13.04.2022 | 0.4 | J. Maas | Processed some parts of the feedback from the stakeholder |
| 14.04.2022 | 0.41 | K. Janssen & J. Maas | Processed all the feedback from the stakeholder |

# 

# 1 Introduction

Soon the students from Delta will have their own high powered computer, also known as the number cruncher. The goal of the number cruncher system is to give all Delta students the ability to easily send their resource-intensive workload, e.g. AI model training and media rendering, to this powerful computer. This could potentially save lots of time for the Delta students, for the number cruncher can do resource-intensive tasks much more efficiently.

The general idea is thus, to create an easy-to-use system for Delta students so they can submit their workloads to the number cruncher and finally get back their results. This while having a system that automatically manages the workloads on the infrastructure and software side.

This document contains the creative problem-solving process and describes the concept, problems, solutions, prototypes and implementations.

# 

# 2 CPS

Creative problem solving (CPS for short) is a process that involves breaking down a problem to understand it, generating ideas to solve the problem and evaluating those ideas to find the most effective solutions.

“CPS is a proven method for approaching a problem or a challenge in an imaginative and innovative way. It helps you redefine the problems and opportunities you face, come up with new, innovative responses and solutions, and then take action.” (*What Is CPS? | Creative Education Foundation*, n.d.)

The CPS has four processes that are (refer to image 1):

1. Clarify
2. Ideate
3. Develop
4. Implement

These steps will be iterated on in the following chapters.

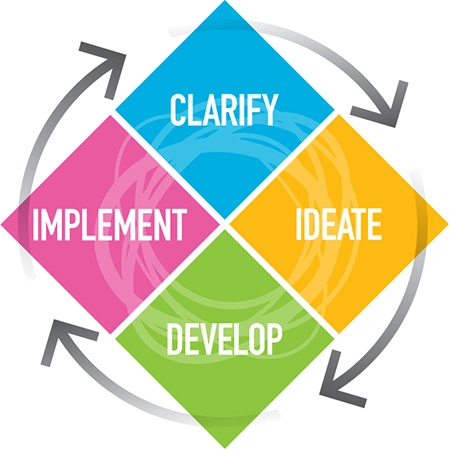


Image 1: CPS process

## 2.1 Clarify

### 2.1.1 Vision

Building a system to facilitate high-performance computing tasks which can be managed and executed from a centrally accessible platform.

### 2.1.2 Objective

Enable all Delta students to perform high-performance computing tasks, as not every student has access to the necessary level of computing power.

### 2.1.3 Goal

Developing a platform that interfaces with a high-performance computing system that can be used by delta students for tasks such as machine learning, code compilation, media rendering and fluid dynamics.

### 2.1.4 Challenge

The developed platform is accessible for everyone within Delta, including other possible granted students, from a technical and usability standpoint.

### 2.1.5 Gather data

To clarify the needs of the users, a survey was created via a form. In this form, questions were asked about the desires of the features of the system, including the use-cases and opinions about its characteristics.

The result of this form showed that the number cruncher computer is going to be mainly used as an AI model training computer. However, there is also a desire to use the computer for code compilation and media rendering (e.g. video and 3D). The computer must be able to connect to the internet outside of the Fontys network. Additionally, users would like to use a GUI to interface with the system. Last but not least, there is some desire for an API, which can be used for automation. However, since most people are not sure they will use it, this is currently not a priority.

## 

## 2.2 Ideate

### 2.2.1 Web interface

With the online web interface, users can get access to the HPC system. For this, users have to identify themselves with the ADFS. This will ensure that only the permitted users can get access, which in this case is currently the Delta student group. When identified, users can submit their public key(s) to the HPC system, which grants them isolated access to the system over an SSH connection.

When the user is new to the HPC system, a fresh UNIX account will be created on the HPC system by using the user's unique Fontys identifier, fetched from the ADFS login. This prevents users from creating multiple accounts and allows system administrators to link users of the system with their real identities.

### 2.2.1 Public web API

Users who know how to use web APIs can choose to interface with the HPC system from a publicly available web API. The API allows them to automate their resource-intensive computing tasks on the HPC system, by for example including it in their CI/CD pipeline(s).

To authenticate the API requests, API clients can create an API token using the web interface mentioned before, which is then included in the API request headers.

### 2.2.3 Command-Line access from web browser

Giving users the possibility to access the command-line interface of their isolated environment on the HPC system. After being authenticated with their Fontys account on a publicly available web interface.

### 2.2.4 Network File System

Providing users with easy access to the files in their own isolated environment. To which they can easily connect their own machine. For example, they can attach their own home folder on the HPC system as a network drive, for example with SMB, CIFS, NFS etc.

## 

## 2.3 Develop/prototype

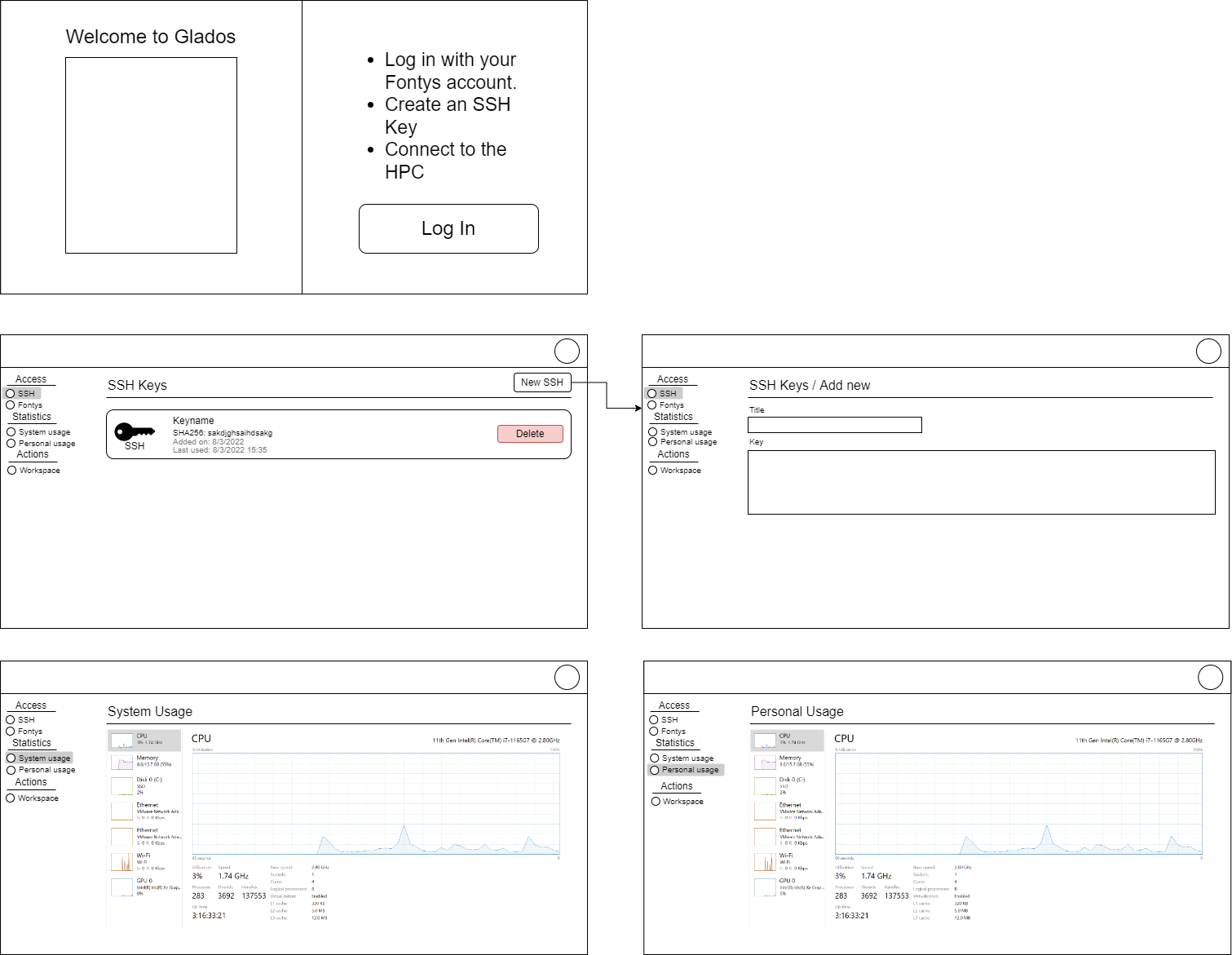
During the development/prototyping phase, the following tasks will be iterated on:

1. Setup of the general software architecture
2. Creation of wireframes
3. Implementation of small prototypes.

### 2.3.1 Software architecture

As a first step of the development/prototyping phase, a C4 model was created, which defines the potential architecture for the final implementation phase. For more information regarding the C4 model, refer to the supplemental document along with the C4 model.

### 2.3.2 Wireframe



### 

### 2.3.3 Prototype

After designing the wireframes, a small prototype was created, which allowed the team to get familiarized with the chosen framework (Next.js). This prototype included the setup of a proper development environment with safeguarding of the code quality by using techniques such as testing and linting. In addition to that, the prototype included the first iteration of the wireframe designs.

## 2.4 Implement

Halfway through the project, a meeting was arranged with the stakeholder, which indicated that the scope was too big. This resulted in the scope shrinking during the development/prototyping phase, which caused the implementation phase to be skipped. The deliverables that were made are described in the development/prototyping phase.

# Conclusion

After a meeting with the stakeholder, it became clear that the defined scope during the conceptual phase was too stretched. The stakeholder indicated that the first iteration should meet the current needs of the end-user. In future iterations, the functionality could be expanded upon when there is a clear need.

# Definitions, Acronyms, and Abbreviations

| **Term** | **Description** |
| --- | --- |
| HPC | High-performance computing |
| CPS | Creative problem-solving |
| LXC | Linux Containers |
| ADFS | Active Directory Federation Services |
| CIFS | Common Internet File System |
| NFS | Network File System |
| SMB | Samba |
| SSH | Secure Shell |
| CLI | Command-line interface |
| API | Application Programming Interface |
| GUI | Graphical User Interface |
| AI | Artificial Intelligence |
| CI/CD | Continuous Integration/Continuous Deployment |

# 

# Bibliography

*What is CPS? | Creative Education Foundation*. (n.d.). Creative Education Foundation.

Retrieved April 14, 2022, from https://www.creativeeducationfoundation.org/what-is-cps/

## 