**Design Specification**

**User Story**

In agile we write user stories to describe a feature that should be implemented by the team.

**For example**, as a **user**, I want to **change the password** of this website when I forgot the old password so that I can access this website.

**Acceptance Criteria**

By definition, acceptance criteria are “Conditions that a software product must satisfy to be accepted by a user, customer or other stakeholders.”(Microsoft Press) That means a set of statements that describes the user’s requirement or features and functionalities of an application.

**For example**, when changing the password, the password must be no less than 8 and no greater than 12 characters, contain at least one uppercase letter, one lower case letter, and at least one number.

[Software Design Document (arxiv.org)](https://arxiv.org/ftp/arxiv/papers/1005/1005.0595.pdf)

**User interface**

\*Explain major components of \*\*user interface\*\* and their breakdowns, should be easy to follow with no prior knowledge of the project. This doesn't need to be (and shouldn't be) the finished design of your project (that is, for example, a bulleted list of views and their features for an app, NOT a finished wireframe). Use graphics only when needed.\*

**Inputs and Outputs**

\*Explain the inputs and outputs of the program. Generally, the information \*\*provided\*\* by the user is considered the input, and the information \*\*returned\*\* to the user is the output.\*

**For example**,

**Functional decomposition into modules**

\*What are all the modules required for the full project? Start big, and break down what you come with into modules that you think would be straightforward to tackle. If something still seems complicated when you're done, break it down further!\*

**Pseudo code for logic/algorithmic flow**

\*What are the main processes that need to happen in the modules you outlined? Explain, in plain English (not code!), how you imagine these processes will work. These processes don't need to be broken down to the extent they will be in your code, just enough that logic is easy to follow.\*

**Dataflow through modules**

\*How will modules within your project interact? What information will be used in which modules, and how will they communicate with other modules to serve their purpose? If you have a login module in an app, for example, what information does it need to convey to the rest of the modules for them to properly function? This could include credentials, the user's name, etc.\*

**Major Data Structures**

\*How do you plan to hold the information you need for this project? Outline the main data structures you will need to do this. If one dataset needs to be used by several different modules for different purposes, how can you store it in a way that is usable by all of them? This should tie in closely with the \*\*Dataflow through modules\*\* section!