# Metacognition

1. **Describe the scope of the project that this prototype relates to**

As per the plan that was developed in Project 1 for this task, there are a series of requirements and business rules that the development of this app relates to. The project scope is a simple 4 page front end, 3 page back end web app that tracks the location of cosplayers (from now on referred to as users) in a virtual queue for a physical photography service. It also stores the information supplied by the users to assist the photographer (from now on referred to as admin) to process and attribute the images that they have taken to the correct user.

While there is a provision for additional field variables such as a check box for model release forms, any additional pages or extra topics could be classified as scope creep and outside of the initial scope and approved plan for The Cosplay Queue.

**2. Describe why prototyping tool chosen was the best for the outlined task**

When I started using Materialize as a prototyping tool it was a new experience and left me wishing I had chosen a tool that I am used to like Bootstrap. However, as development went on some of the internal components and code were indispensable when it came to creating the finished prototype.

While I am sure the same effects and components could be replicated with a different tool or program, for this project, after taking some time to adjust to the difference in coding languages it proved itself to be the best tool for the job.

**3. Write three paragraphs each for how you think the tool has handled the rendering of all CSS/HTML and JavaScript code.**

HTML – Materialize has an interesting HTML code structure that varies from other coding methods I have utilised before. While Bootstrap and plain HTML share common structures, Materialize takes the common structures and adds onto them, which can add an element of difficulty when you are attempting to code what used to be universal elements such as checkboxes and form input fields. One key example of this was attempting to create the checkbox needed to enable local storage, what can be achieved in a single line of code in straight HTML requires 8 lines to do the same thing in Materialise.

CSS- When it came to rendering the CSS code for this prototype it was incredibly easy to utilise and alter the existing Materialize CSS code to create the look for the prototype. While other prototyping tools may require less CSS to achieve the same outcome Materialize created a stylish look while maintaining ease of use and flexibility in customisation.

JavaScript – While great for rendering HTML and CSS, when it comes to JavaScript code Materialize isn’t perfect in its execution. There was an issue that arose when I was attempting to implement the local storage. Due to the naming code structure that Materialize requires in order to enable input verification it is in impossible to implement local storage with validation on the same div due to duplicate ID’s. Once the ID duplication was removed local storage was able to function normally.

Overall all though the inbuilt JS stylesheet is incredibly functional and creates animation and effects effortlessly with little to no input from the developer.

**4. Conclusion as to how you might be able to use some of the components the tool has rendered going forward to UX2**

While building out this prototype I ended up taking my construction a step further and building out the entirety of the front end page structure. When I move into UX2 I will be able to utilize the code that I already have and the components that I utilised in the initial build as a starting point for the further development and the functionality of the overall app.

**5. Document the licensing and/or costs associated with your chosen tool**

Overall the development of this prototype was completely free as Materialize is a free to use prototyping system. There are no costs or licencing fees included so there overall cost is $0.