4.0 Results and Discussions

The prototype our group developed for a smart mirror project it's not perfect, but it contains %70 of all the features we initially conceptualize at the beginning of this project. The reason being that we had a range of issues from "what is achievable" to project management skills to not always being consistent, but more importantly, we learned a whole lot of skills from each aspect of the project.

Without a doubt, we learned and improved our techniques skills from software programming to hardware troubleshooting. For example, we learned how to work with 3rd party software such as Google assistant and work with open-source software from GitHub. In more detail, we learned how to configure and install the necessary software packages to enable google assistants and MagicMirror2. During the configuration process, we faced tremendous issues related to incompatible packages, software error, incompatible hardware, most of these issues were resolved by consulting with documentation of google assistant and MagicMirror2. Reading the documentation and configuring 3rd software packages, whether it's open-source or proprietary software is very useful skills for software engineers.

Another technical skill we learned how to come up with alternative software solutions based on our current skill level because some solutions we conceptualize we couldn't skill-fully implement due lack of experience with languages and understanding code-base of open-source software. For example, we wanted to add a feature to Magic-mirror software such as creating alarm widgets to alert the user, however, the task was too very difficult because we had to first understand the Magic-mirror complex Node.js, learn Node.js syntax and functions, importantly finish the task before the deadline.
These software difficulties pushed me to find an alternate solution, which is Linux service, which is an application that can be run back around to do something. This concurrent behaviour can be used with Magic-mirror software. In another word, we implemented each feature as a Linux service to run concurrently with MagicMirror2. This helped us to learn more about Linux services, Linux commands, and where else it can be used.

In terms of the hardware and embedded system, we learned how to integrated various input and out sensors into our raspberry pi. In more detail, we learned each pin in raspberry-pi had a different purpose, how to read and write to pin. We learned how to test the PCB board for connection between the traces by using a multi-meter, better soldering techniques.

In equal value to technical skills, we learned in the hardware and software part of the project, we learned a lot of software skills. For example, in the case of project management, we tried to follow an agile project management method because of its most common project method, and it makes sense than waterfall. We implemented scrum for each feature we wanted to prototype to have at a minimum, and we created a list of the backlog that contain all features. Each sprint was one week, and during that sprint, we picked the highest priority feature from the backlog and worked on it. When finished each feature, we review it and check if it meets all requirement or if want to change modify if we didn't like it. There are a lot more skills such as being consistent, debugging, writing emails, technical writing, and a lot more. Needless to say, the skills we learned will be in big use in our careers or even in other areas of life.

5.0 Conclusions

Creating a large number of smart mirrors for the rest of the Canadian population that can bring a lot of interesting challenges that would have n't risen if only we produced a dozen of them. When it comes to hardware there is not much of a challenge, it's just plug in a microphone, infrared sensor, and build a nice case that fits everything. The true challenge comes from the software side, install all necessary software, and creating necessary software infrastructure to connect all smart mirror. It was easy to install open-source and proprietary software in one raspberry-pi with all necessary support packages but it would be time-consuming to install all necessary software to thousands mirror. There are two solutions to this time-consuming problem. The first to create an agreement with owners of the raspberry pi to pre-install of this software just after manufacturing for sale percentage. The second solution its to create our hardware similar to raspberry pi minus all stuff that is not needed.

Most likely we will continually update, improve the user experience, connect all users, help all user issues without coming to there door, only if we connect all our smart mirrors into a single cloud platform. This platform will gather data from all smart mirrors and enable our team to resolve the issue before the customer give us a call.