Reflect & Assist

Kacey Morris

CST-452 Capstone Project Proposal

Grand Canyon University

Instructor: Professor Mark Reha

Revision: 2.0

Date: April 24, 2022

**ABSTRACT**

The Reflect & Assist is a Raspberry Pi kit with all materials and software included to create a Smart Mirror display. This display is a mobile mirror that projects information contained in modules alongside its reflection. The Reflect & Assist kit will include the Raspberry Pi, a monitor, plexiglass, links to required GitHub modules, as well as instructions on how to set up the Raspberry Pi, install all required dependencies, and assemble the mirror set up. As a kit, those interesting in building a customizable smart mirror will find all essential resources in a centralized location, making learning programming and embedded systems simpler and more accessible.

The kit will include 14 different display modules options, including an alarm clock, weather display, a daily Bible quote, and many others, as well as access to a website for managing the individual display. To encourage creativity and expansion, access to an extended list of potential modules and a template for personalized module creation will be included as well.

|  |
| --- |
| History and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Revision Notes** |
| September, 2021 | Kacey Morris | Initial draft for review/discussion |
| April 24, 2022 | Kacey Morris | Added more detail to creation motivation, removed redundant objectives, updated how the management and mirror configuration are connected, updated physical supply requirements, updated connection diagram |
|  |  |  |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

**Integrated Instructor Feedback into Project Documentation**

Yes  No

**Project Approval**

Professor Mark Reha

**TABLE OF CONTENTS**

Project Overview and Project Objectives 4

Project Scope 5

Project Success Measures 6

Project High-Level Solution 7

Project Controls 8

Project Cost and Schedule 10

Appendix A – References 11

Appendix B – Copyright Compliance 12

Project Overview and Project Objectives

**State the Problem and Background**

Reflect & Assist is targeted towards individuals beginning to work with programming and embedded systems, although it could be integrated within non-programmers due to its ease of use and clear instructions. When purchasing a pre-assembled mirror with virtual display, the price can be quite daunting which may deter a great percentage of the population. If an individual takes the initiative to construct the mirror, they will need to research and purchase all materials separately without a guarantee that the materials and resources are compatible with each other, or produce the desired result.

With this product, the cost deterrent is reduced and more people gain access to the benefits of a smart mirror. A custom-built smart mirror has several benefits, particularly in regard to organization. Users have the option to choose from several organization modules such as an alarm clock to keep them on schedule, a Bible quote to motivate them for the day, and an overview of weather and news headlines that may impact their daily choices. As a kit, the users may also customize what they see when they look in the mirror to better fit their daily needs. Some more specific modules include sports statistics and Bible verse displays, allowing for a diverse experience specific to each user.

This solution came to fruition when one of the members of the development team began experimenting with Raspberry Pi projects. One of the common introductory projects for understanding embedded systems was the establishment of a smart mirror, but the tutorials consistently included solely the software needed to implement the modules. When learning a new skill, commitment to materials without a guarantee that they will perform as expected can significantly defer the induction of a project. Reflect & Assist bypasses the guessing required in most tutorials and allows for an easier introduction to the knowledge of embedded system. Any changes to the mirror surface also required alterations to the backend code manually, which took unnecessary time and effort. With the development of the management site, the mirror surface can be configured easily and seamlessly, providing a more enjoyable user experience.

**Project Objectives**

The success of Reflect & Assist depends greatly on the successful assembly of a customizable smart mirror. The mirror will be constructed with the following goals in mind.

* Maintain a collection of all materials required to create a smart mirror at the lowest possible cost.
* Display information modules on the surface of the mirror.
* A reflection is displayed on the face of the mirror alongside the information modules.
* The modules can be updated through the use of a website.
* The instructions are understood by non-programmers.
* A module not included in the original scope can be integrated within the mirror and website.

**Challenges**

The assembly of Reflect & Assist will be challenging for the development team. Although there may be some challenges, the end product will be beneficial and encourage more people to explore programming. Some of the challenges that are apparent in the creation of Reflect & Assist are as follows.

* The development team does not have experience with the hardware or software programming language utilized in the project, so a learning curve may extend the development timeline.
* Several third-party software modules are included in the function of the mirror and there is not a guarantee that they are functional, so manipulation of the existing modules may be required during development.
* Different programming languages will be utilized for different section of the software so establishing communication between the essential pieces of software may pose a challenge.
* Writing instructions that make sense to beginning programmers may prove as a challenge.

**Benefits and Opportunities**

The target audience of Reflect & Assist includes Computer Programming or Software Development students interested in developing skills related to embedded systems and Internet of Things technology. A smart mirror can be integrated into the daily life of its users and improve organization and overall daily satisfaction. This low-cost solution to an otherwise lucrative end product will encourage those not yet involved in programming to begin their development experience with Reflect & Assist. This product will make programming fun, easy, and accessible to those interested. The spread of knowledge surrounding Computer Programming in a simple format will decrease the stigma of extreme difficulty that may discourage people from trying to program. In a classroom setting, Reflect & Assist also proves extremely beneficial. This product provides an easy-to-understand introduction to embedded systems while maintaining a low-cost within the range of a typical student. Widespread knowledge of this type of technology will drive technological advances and improve the quality of the world.

Project Scope

The completion of a Reflect & Assist kit will provide the customer with a smart mirror powered by a Raspberry Pi that displays modules on the mirror surface and is managed by a responsive web application. When completed, users will have the ability to do the following in scope functions.

* Display four sections of information on the mirror
* Choose the mirror display modules
* Save a display state
* View their reflection (unless otherwise incapable)
* Turn on/off the module display
* View the current time
* View daily updating compliments
* View daily updating insults
* View the current weather for their location
* View the weather forecast for their location
* View a news feed
* Configure visual and auditory alarms
* View the current moon phase
* View an image of the World
* View a daily updating Bible verse
* Configure and view sports game statuses

Although the project will include those features, it can be expanded in the future. Out of scope features are as follows.

* View a daily horoscope
* View a picture
* View a video with sound
* Alternate saved displays with hand gestures
* View a daily countdown
* View a daily joke
* View a daily comic
* View a daily Pokémon character
* Create multiple accounts
* Save multiple displays
* Control frame lights
* Remote module control

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Work Breakdown Structure | | | | | | | | | | |
| ID | Task | Dependencies | Status | Effort Hours | Cost | Start Date | Planned Completion | Estimate to Completion | Actual Completion | Resource |
| 1 | Construct the Project Proposal | N/A | Ongoing | 8 Hours | N/A | 9/7 | 9/26 | 9/26 | 9/26 |  |
| 2 | Outline the technologies used in the project | N/A | Ongoing | 3 Hours | N/A | 9/7 | 9/26 | 9/25 | 9/25 |  |
| 3 | Outline the project functionality | N/A | Ongoing | 2 Hours | N/A | 9/7 | 9/26 | 9/24 | 9/24 |  |
| 4 | Outline the steps required to complete the project | N/A | Ongoing | 2 Hours | N/A | 9/7 | 9/26 | 9/26 | 9/26 |  |
| 5 | Detail the Project Requirements Analysis | N/A | Future | 8 Hours | N/A | 10/4 | 10/24 | 10/24 | 10/24 |  |
| 6 | Construct diagrams of back-end design | N/A | Ongoing | 1 Hour | N/A | 9/7 | 9/26 | 9/25 | 9/25 |  |
| 7 | Construct wireframes of front-end design | N/A | Future | 1 Hour | N/A | 10/4 | 10/24 | 10/18 | 10/20 |  |
| 8 | Detail User Cases | N/A | Future | 3 Hours | N/A | 10/4 | 10/24 | 10/20 | 10/21 |  |
| 9 | Construct Architectural Requirements | N/A | Future | 3 Hour | N/A | 11/15 | 11/21 | 11/21 | 11/21 |  |
| 10 | Finalize Technical Requirements | N/A | Future | 1 Hour | N/A | 11/15 | 11/21 | 11/20 | 11/20 |  |
| 11 | Finalize Design Plan and Concepts | N/A | Future | 2 Hours | N/A | 11/15 | 11/21 | 11/19 | 11/18 |  |
| 12 | Construct Test Cases | N/A | Future | 2 Hours | N/A | 12/6 | 12/19 | 12/16 | 04/24 |  |
| 13 | Complete Unit Tests for all Modules | GitHub code blocks of 10 external modules (MIT Licenses) | Future | 4 Hours | Free Modules under MIT Licenses | 12/6 | 12/19 | 12/19 | 04/24 | See References for modules |
| 14 | Complete Component Testing for the front-end and back-end | GitHub modules | Future | 4 Hours | Free Modules under MIT Licenses | 12/6 | 12/19 | 12/17 | 04/24 |  |
| 15 | Assemble the hardware | Raspberry Pi OS | Future | 3 Hours | $130-$180 Overall  $35 – Plexiglass  $20 – Frame  $50-$100 – Monitor  $30 – Pi | 12/6 | 12/19 | 12/7 | 04/10 |  |
| 16 | Establish the development environment | Geany or other IDE | Future | 1 Hour | N/A | 12/6 | 12/19 | 12/7 | 11/18 |  |

Project Success Measures

The success of Reflect & Assist will depend greatly on the functionality of the smart mirror product. The success will be measure by durability, user acceptance, database efficiency, visual aesthetic, functionality, and diversity. Several specific metrics can measure project success including the following.

|  |
| --- |
| Project Completion Criteria |
| 1 - *At least 70% of Software Development students report a greater inclination to create a smart mirror with access to the Reflect & Assist.* |
| 2 - *The database is able to support 100 different mirror owners* |
| 3 - *The website content is contained on a cellular device screen size.* |
| 4 - *The mirror fits four modules comfortably without obstructing the user’s reflection.* |
| 5 - *All 14 modules display the correct information on the mirror surface* |
| 6 - *The display is visible in artificial and natural light.* |
| 7 - *The mirror is mobile and can be set up in a different location within five minutes.* |
| 8 - *Stability of the display withstands being carried.* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Assumptions and Constraints | | | | | |
| ID | Description | Comments | Type | Status | Date Entered |
| 1 | It is assumed that the selected third party GitHub modules function how they are described in the module descriptions. | There are nine third-party modules included in the construction of this project and it is likely that at least one will not function correctly. | Assumption | Ongoing | 9/23 |
| 2 | It is assumed that the Raspberry Pi has enough memory to perform the required tasks efficiently. |  | Assumption | Ongoing | 9/24 |
| 3 | It is assumed that the Raspberry Pi will be safe to use and will not cause me physical harm or delete my existing work. |  | Assumption | Ongoing | 9/20 |
| 4 | It is assumed that the MIT Licenses associated with the third-party software will not change within the lifecycle of the project. |  | Assumption | Ongoing | 9/25 |
| 5 | It is assumed that the supplies are accessible and within the budget of the project. |  | Assumption | Ongoing | 9/19 |
| 6 | It is assumed that Python and PHP have the capability to support the software requirements. |  | Assumption | Ongoing | 9/25 |
| 7 | It is assumed that the display will show through the plexiglass overlay. |  | Assumption | Ongoing | 9/19 |
| 8 | It is assumed that the third-party modules can work together within an application. |  | Assumption | Ongoing | 9/26 |
| 9 | It is assumed that Python can connect to a MySQL database. |  | Assumption | Ongoing | 9/26 |
| 10 | The development team does not have unlimited resources and the cost may exceed the delegated budget. |  | Constraint | Ongoing | 9/25 |

Project High-Level Solution

**Introduction**

The construction of the smart mirror will require several pieces of hardware as well as the installation and development of software. To begin, the Raspberry Pi will need to be connected, turned on, and the Raspbian Operating System will need to be installed and configured. It is assumed that the Pi contains enough physical RAM to support the operating system as well as all modules because the Raspbian OS was made specifically for Raspberry Pi’s, and the modules are simple programs constructed to work with the Pi. Once the OS is configured, the development environment Geany will also need to be installed and Python should be downloaded on the system. From there, the GitHub modules will be downloaded using the instructions located in the module descriptions.

The user facing management web application will be constructed using PHP, HTML, CSS, and Bootstrap. Geany will also be used for the PHP file development and the program will follow an N-Layer design and include MVC architecture. The database layer will utilize SQL to connect to and manipulate a MySQL database hosted through Azure. Information about the registered user and saved states of the mirror display will be documented in separate tables. To interact with the JavaScript modules, an API containing JSON data will send information about what should be displayed on the mirror from the PHP application. As an example, when a user logs into the management application, a session variable will be saved detailing the users ID. The user ID allows the database to connect that specific user with the details of other tables. When the user turns on the mirror, the most recent entry in the mirror state table will be compiled into a JavaScript configuration file, replace the current configuration file for the mirror, and display the desired information on the mirror surface. If the user wanted to display the weather, moon phases, alarm, and Bible verse, they would select the positioning in the management app, that positioning would be saved in the database, then as the mirror turned on the management app would access the database and relay the information through the configuration file in the mirror display directory.

The physical construction of the display requires a monitor, reflective acrylic, tape, and an HDMI cord. When working with the acrylic, the piece will be large enough to cover the surface of the monitor and can be trimmed using a razor blade with the edges taped for preservation and to prevent splintering. The tape will hold the monitor and plexiglass together as well as provide a border for the mirror. An HDMI port on the Raspberry Pi is required to connect the Pi with the monitor. It is also assumed that the mirror will be placed in a location near an outlet, as the mirror will need to be plugged in for power.

**Solution**

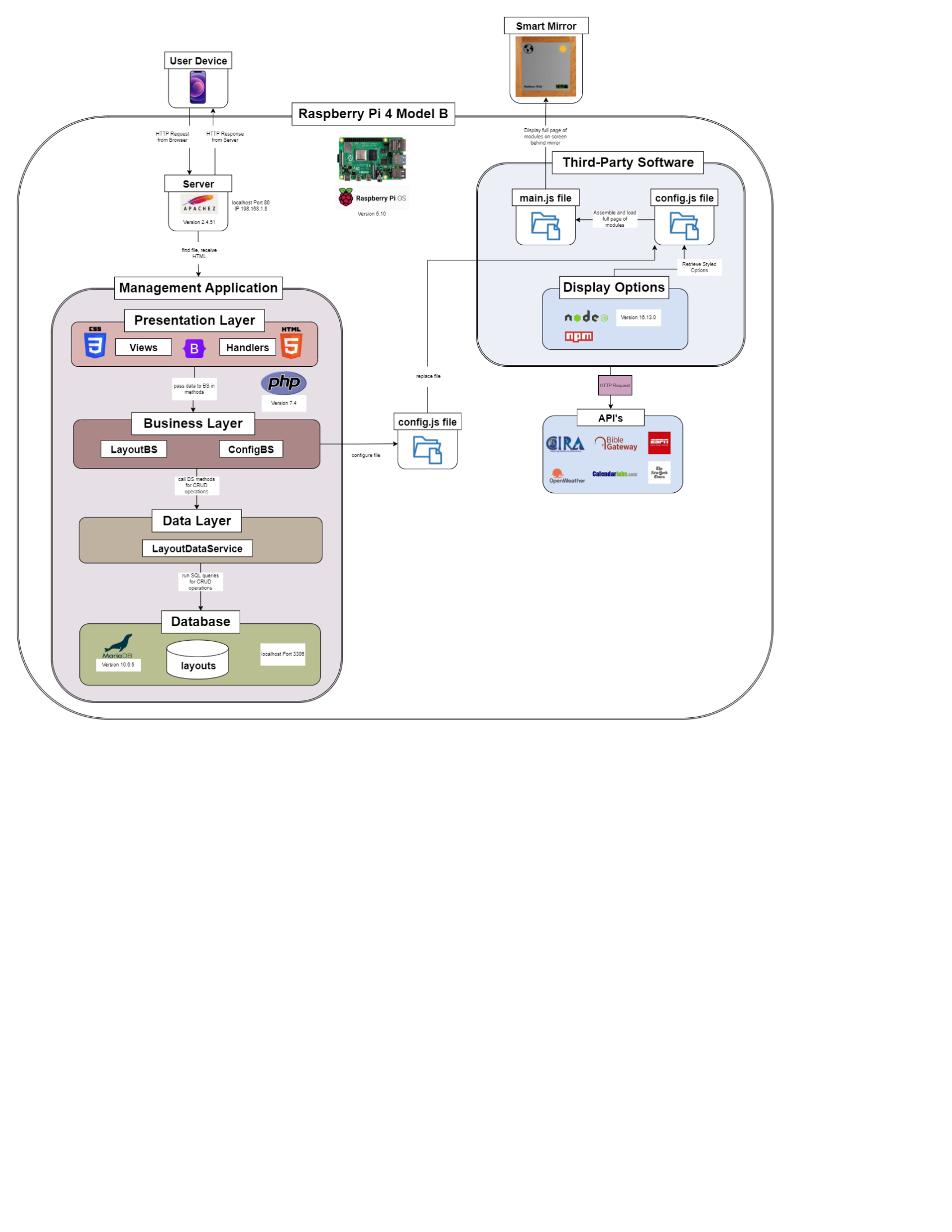
Because the development team is currently unfamiliar with many of the technologies, several tutorials will be utilized, as well as extensive research into the interactions of each piece of hardware. As an end product, the user will need to register an account through the management web application. The management app will use a form constructed in HTML, CSS, and Bootstrap. The form will collect a username, password, and email address from the user, which will be properties of the user model. When the user selects the register button, the controller will direct the model properties to the business layer. The business layer will then relay the information to the database layer, which will then perform SQL queries to create an entry in the database under the user table. With that entry, a user ID is associated with the specific user’s information and will connect that user with all future interactions. A default preferences database entry will also be created and associated with this user. Within the PHP code, a session variable will store the user ID. The user will then have access to the management site where they can update their user information through a similar process or begin setting up their display. To set up the display, the user will choose from a dropdown list of existing information modules relating to the desired orientation on the mirror. For example, there will be dropdowns for the top-left, top-right, bottom-left, and bottom-right positions on the mirror. None of the locations will be required, so if the user does not want information at all places, they do not need to choose an option for each. Zero to four information modules can be displayed on the mirror at a time and there will be 14 module options to choose from. Once the desired modules are selected, the user will select the save button. The save button will relay the chosen module names and the user ID down the layers and an entry will be made in the database mirror state table. This table will contain information on the user ID as an integer and saved state for each position as a string.

When loading the display, the database access layer in the management application will access the active display state from the database associated with the current session user ID. This state will then be passed up to the business layer and converted to a JavaScript configuration file. That file will replace the configuration file in the NodeJS application, which will indicate which information modules should be displayed on the mirror and in what location. The display will then turn on and display the information on the screen, which will show through the acrylic while also maintaining a reflection of the user on the surface.

The initial set up of the modules will follow instructions connected to each GitHub module which usually includes running a command in the command prompt on the Raspberry Pi to download the project. A list of chosen default modules for the mirror are listed in the References, as well as a link to a website displaying common modules to extend the project. All modules selected for integration in the mirror have MIT Licenses, which means they are free to use as long as reference is made to the source within the project. UML diagrams will be constructed for the user and mirror state models. Wireframe diagrams will plan the views required by the management application, as well as the desired mirror display.

All tutorials and module instructions are listed in the References section. For the construction of the frame and mirror setup, the YouTube video by Davis and Aaron will provide guidance in the process (Davis & Aaron, 2016). Instructions on how to download the required GitHub modules can be found in the overview of each module. The article by Bulat provides insight on how to connect a PHP application to a Python application (Bulat, 2018).

This organizational approach will allow the end product to meet all desired objectives of the project including the physical display, python modules, and management web application.

******

*The interaction between the management web application, the database, and the mirror display.*

Project Controls

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Management | | | | |
|  | **Risk Probability** | **Risk Impact** |  |  |
| **Event Risk** | **(high, medium, low)** | **Risk Mitigation** | **Contingency Plan** |
| Computer Failure | Low | Restarting the project from scratch | Ensure the computer has reliable specifications | Back up all code and documentation on GitHub |
| Limited experience with Python | Low | May not be able to run the desired modules | Conduct extensive research into Python and create unit tests for small code snippets | Research different tutorials for interacting with Python |
| No experience with Raspberry Pi’s | Medium | May need to change the project if this cannot be resolved | Conduct extensive research into Raspberry Pi’s and create unit tests for small code snippets | Establish clear communication with the project mentor |
| No experience connecting Python and PHP | High | The project will not be able to interact with all required sections | Conduct extensive research into Python and PHP and create component tests for small code snippets | Maintain clear separation of concerns in case the management application must be written in a different language |
| May not be able to extend the project to other locations | High | Cannot market the project | Host all reasonable information on the cloud | Provide detailed instructions on how to locally host all required information |
| Potential Extensive Cost | Low | Project cannot look as professional as envisioned | Research low-cost solutions | Provide alternative options in the documentation for future recreation |
| GitHub Modules do not Function Correctly | High | User information display options will be reduced | Unit testing of modules and ensuring published dates are recent | Choose different information modules to display |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Issues Log | | | | | | | | |
| **ID** | **Description** | **Project Impact** | **Action Plan/Resolution** | **Owner** | **Importance** | **Date Entered** | **Date to Review** | **Date Resolved** |
| 2 |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Change Control Log | | | | | | | | | |
| **ID** | **Change Description** | **Priority** | **Originator** | **Date Entered** | **Date Assigned** | **Evaluator** | **Status** | **Date of Decision** | **Included in Rev. #** |
| 1 |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |

Project Cost and Schedule

|  |  |
| --- | --- |
| Material | Cost |
| Raspberry Pi | $100 |
| Washi Tape | $5 |
| Reflective Acrylic | $35 |
| HDMI Cord | $5 |
| Monitor | $130 |

Appendix A – References

Bethge, J. (2021, September 1). *Jopyth/MMM-Remote-control*. GitHub. Retrieved September 26, 2021, from https://github.com/Jopyth/MMM-Remote-Control.

Bulat, R. (2018, October 24). *Python to PHP Communication — How to Connect to PHP Services using Python*. Medium. Retrieved September 26, 2021, from https://rossbulat.medium.com/python-to-php-communication-how-to-connect-to-php-services-using-python-f48893a2c98e.

Clarke, J. (2021, March 19). *JCLARKE0000/MMM-myscoreboard*. GitHub. Retrieved September 26, 2021, from https://github.com/jclarke0000/MMM-MyScoreboard.

Davis, & Aaron. (2016, September 5). *How to Make a Raspberry Pi Smart Mirror*. YouTube. Retrieved September 26, 2021, from https://www.youtube.com/watch?v=fkVBAcvbrjU.

Fewieden. (2020, April 28). *Fewieden/MMM-AlarmClock*. GitHub. Retrieved September 26, 2021, from https://github.com/fewieden/MMM-AlarmClock.

Gartenberg, C. (2017, August 17). *Building your own smart mirror is surprisingly easy*. The Verge. Retrieved September 26, 2021, from https://www.theverge.com/circuitbreaker/2017/8/17/16158104/smart-mirror-diy-raspberry-pi-commute-weather-time-gadget.

Garza, A. (2020, April 14). *Arthurgarzajr/MMM-dailybibleverse*. GitHub. Retrieved September 26, 2021, from https://github.com/arthurgarzajr/MMM-DailyBibleVerse.

HackerShackOfficial. (2017, February 12). *Hackershackofficial/Smart-Mirror*. GitHub. Retrieved September 26, 2021, from https://github.com/HackerShackOfficial/Smart-Mirror.

Kingdon, N. (2020, November 5). *Nolankingdon/MMM-moonphase*. GitHub. Retrieved September 26, 2021, from https://github.com/NolanKingdon/MMM-MoonPhase.

LukeSkywalker92. (2021, January 17). *Lukeskywalker92/MMM-globe*. GitHub. Retrieved September 26, 2021, from https://github.com/LukeSkywalker92/MMM-Globe.

MatthiPi. (2020, August 23). *Matthipi/MMM-defaultweatheranimated*. GitHub. Retrieved September 26, 2021, from https://github.com/MatthiPi/MMM-DefaultWeatherAnimated.

Mykle. (2020, August 17). *Mykle1/MMM-Insults*. GitHub. Retrieved September 26, 2021, from https://github.com/mykle1/MMM-Insults.

*Raspberry Pi Smart Mirror*. Hackaday.io. (n.d.). Retrieved September 26, 2021, from https://hackaday.io/project/13466-raspberry-pi-smart-mirror.

Teeuw, M. (2021, July 1). *Michmich/MagicMirror*. GitHub. Retrieved September 26, 2021, from https://github.com/MichMich/MagicMirror#manual-installation.

Teeuw, M. (2021, September 21). *3rd party modules · Michmich/MagicMirror Wiki*. GitHub. Retrieved September 26, 2021, from https://github.com/MichMich/MagicMirror/wiki/3rd-Party-Modules.

Werth, D. (2021, June 21). *Werthdavid/MMM-Burnin*. GitHub. Retrieved September 26, 2021, from https://github.com/werthdavid/MMM-BurnIn.

Appendix B – Copyright Compliance

Several GitHub project modules are included in the creation of Reflect & Assist. All links to GitHub projects in the References section contain a sub link leading to the License of the third-party software. Every piece of third-party software incorporated into the project falls under an MIT License, which is as follows.

“Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

The software is provided “as is”, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software,” (Teeuw, 2021 *MichMich/MagicMirror*).

The third-party modules will be incorporated as created to improve user experience. Descriptions of each module are provided below.

Clock – This module displays a digital clock on the surface of the mirror (Teeuw, 2021 *MichMich/MagicMirror*).

Compliments – Displays daily updating compliments (Teeuw, 2021 *MichMich/MagicMirror*).

Hello World – Displays the phrase “Hello World” on the mirror display (Teeuw, 2021 *MichMich/MagicMirror*).

Calendar – Displays a calendar view (Teeuw, 2021 *MichMich/MagicMirror*).

Current Weather – Displays the current weather for a specified location (Teeuw, 2021 *MichMich/MagicMirror*).

Weather Forecast – Displays the weather forecast for a specified location (Teeuw, 2021 *MichMich/MagicMirror*).

News Feed – Displays a news feed (Teeuw, 2021 *MichMich/MagicMirror*).

Alert – Displays an alert on the surface of the mirror (Teeuw, 2021 *MichMich/MagicMirror*).

Moon Phases – Displays an image of the current phase of the moon as well as a text description (Kingdon, 2020).

Globe View – Displays an image of the world as seen from space, updating throughout the day (LukeSkywalker92, 2021).

Daily Bible Verse – Displays a daily Bible verse (Garza, 2020).

Alarm Clock – Configures a visual and auditory alarm (Fewieden, 2020).

Insults – Displays a daily insult (Mykle, 2020).

Sports – Displays the statistics of a desired sports game (Clarke, 2021).

Prevent Burn In – Periodically inverts the screen colors to prevent burn in (Werth, 2021).

Pretty Weather – Formats the weather display to include more visually appealing graphics (MatthiPi, 2020).

Turn on and Off Display – Assists in safely turning the display on and off (Bethge, 2021).

Although several instances of third-party software are included in Reflect & Assist, the management software will be assembled by the development team. The use of this outside software is to increase the end user experience with the mirror by provided more options for the display. Because these additions are solely for visual enhancement, it would be unwise to develop individualized options for all functionality.