International Journal of Scientific Engineering and Science
Volume 8, Issue 11, pp. 99-103, 2024. ISSN (Online): 2456-7361

Building a Smart Mirror with Raspberry Pi

Kuo-pao Yang[1], Trevor Patorno[2], Camden McAllister[3], Brady Landry[4], Marco Figueroa[5]
[1, 2, 3, 4, 5]Computer Science Department, Southeastern Louisiana University, Hammond, Louisiana, USA

Abstract**—***This paper describes a project that creates a smart mirror with Raspberry Pi. Smart mirror technology is becoming increasingly accessible to the general public due to its affordability and widespread availability. Its appeal is not only rooted in ease of access and usability but also in its ability to streamline and centralize information in a minimalist and focused manner. This enhances convenience for users by presenting only essential information without overwhelming them with unnecessary details. The project developed a cost-effective approach to implementing smart mirror technology using Raspberry Pi.*

Keywords**—** *Smart mirror, Raspberry Pi, Google Assistant*.

I. Introduction

People often associate the role of smart mirrors in the modern household with the futuristic depictions seen in television shows and movies. These portrayals envision a future shaped by a century of technological progress, where advanced technologies seamlessly integrate into daily life, providing easy access to knowledge and entertainment [1]. However, smart mirrors are frequently perceived as requiring significant wealth and advanced technical expertise to become a reality. As a result, this technology is often seen as something distant and far in the future. In this mindset, we find ourselves imagining new technologies and speculating about their future direction. The future is always tomorrow, never today, so many fail to consider the possibility of creating these future inventions in the present, using the technologies that are already available to us.

Using these tools and this guidance, it is possible to come up with a design that implements smart mirror technology into a simple base mirror design that makes these inventions of the future possible within our homes. This can be accomplished by using the miniature, hand-sized computing machine, the Raspberry Pi [2][3]. In the 1970's to 1980's, computing simple mathematical functions would require an entire room of space for a few simple machines. Nowadays, millions of these mathematical functions are completed every second within miniature computers [4]. This description fits the Raspberry Pi quite well, but another such example would include the smart phones that we carry in our pockets.

Smartphone technology follows us everywhere, from the bathroom to the dining room, often becoming a root cause of our distraction from the environment and people around us. With a device that grants access to the entirety of human knowledge, it is unsurprising that its sheer accessibility can lead to overstimulation. Here, smart mirror technology emerges as a viable alternative to smartphone technology. With its minimalist interface, smart mirror technology offers users the ability to focus on only the most relevant information at a given moment. By limiting distractions and streamlining information, smart mirrors foster a more focused and purposeful approach to daily activities.

Using Raspberry Pi [5][6], this Smart Mirror project is able to implement smart mirror technology into compact devices, allowing for its usefulness and versatility in application [7]. Many different approaches have also used Raspberry Pi in such a way for a smart mirror to properly operate [8]. Such implementations also include the use of virtual assistants in different efficacious manners [9][10]. The usefulness of virtual assistant services has been challenged in the past, but it appears that their presence in the marketplace has stood the test of time, further improving the potential of this project [11]. The implementation of virtual assistants can also provide additional help, making them more accessible, and this accessibility will increase as time goes on [12].

This paper explores the evolution of smart mirrors over the years and examines how efficient production is crucial for their continued development [13]. The implementation of the Smart Mirror project is thoroughly discussed, along with the challenges encountered during its development. These challenges are addressed in the evaluation section. Finally, the conclusion provides a detailed discussion of potential improvements and directions for future works.

II. Related Work

Competing implementations of smart mirror technology for the general market often include products placed in the marketplace. While these smart mirrors are very intelligent, they often are not affordable to the average consumer. Furthermore, smart mirrors that exist in the general market are usually not configurable, limiting users' ability to customize features according to their specific needs and preferences.

There exists a need for affordable and configurable smart mirrors, and as such, the "one-size-fits-all" smart mirror simply does not cut it. Other solutions that are both cheap and highly customizable smart mirror models are available to the general public. However, these solutions fall into a new pitfall: the average consumer lacks the time and technical expertise necessary to produce such models from complete scratch. This reaffirms the need to balance affordability, customization, and ease of use, ensuring that smart mirror technology is accessible to a broader audience without requiring extensive technical expertise.

III. Implementation

A. Software Approach and Implementation

The first place to venture when trying to capture the "magic" of smart mirror technology is the Magic Mirror library, an open-source platform that integrates a series of

International Journal of Scientific Engineering and Science
Volume 8, Issue 11, pp. 99-103, 2024. ISSN (Online): 2456-7361

software modules in order to implement individual features within the mirror itself [14]. These modules can be used interchangeably within an interface to serve the user with any amount of information they request. At this junction, we can already see how high performance and customizability trampled the common market smart mirrors on their software implementation alone.

In this way, the accessibility that Magic Mirror creates with its module system is a great implementation of the component software architectural pattern, in which code is broken into smaller components in order to create the whole of a software program. Users can develop modules themselves and also access a marketplace of free-to-use modules developed by other community members, designed for easy exploration and future implementation.

Magic Mirror has a variety of methods that concern the initiation of the interface when first activating the mirror itself. Some implementations use modules themselves in combination with the interface itself. But if the user does not want such an intricate setup for any number of unspecified reasons—for risk of lower performance, simplicity of setup, or otherwise—a simple script will suffice. Fig. 1 shows a simple script that can open the Magic Mirror interface with ease.

```
import os

exit_code = os.system("cd ~/MagicMirror && DISPLAY=:0 npm start")

if exit_code == 1:
    print("ERROR: Failed to open Magic Mirror. exit_code: %d" %
          exit_code)
elif exit_code == 0:
    print("Now opening Magic Mirror. exit_code: %d" % exit_code)
```

Fig. 1. Software Implementation to Launch the Magic Mirror Interface in Python

Importing of the ‘os’ Python library is only of use when attempting to reach and return the exit code to the user within the Raspberry Pi console [15]. To make a codebase more accessible to users with limited technical knowledge, it is important to provide thorough and detailed explanations for every component. With this considered, the opening of the Magic Mirror interface itself can be straight-forward without a wealth of technical knowledge. This is particularly relevant when applied to the internal infrastructure of Magic Mirror, which utilizes a catalogue type system in order to handle modules. While the technology has been simplified in this way, the configuration file which handles most modules via a JavaScript interface can still be quite daunting.

Managing the configuration file within a large JavaScript file might initially be challenging for some users. However, with the previously mentioned detailed explanation, users should be able to decipher the program, enabling them to implement their own modules after a brief adjustment period.

Fig. 2 demonstrates that the opening of the configuration file written in JavaScript handles a variety of Hypertext Transfer Protocol Secure (HTTPS) callouts for the internet interactions required by the platform and its modules.

```
let config = {
        address: "localhost",
        electronOptions: {
                webPreferences: {
                webViewTag: true
                }
        },
        port: 8080,
        basePath: "/",
        ipWhitelist: [ // IPs to whitelist!]
        …
}
```

Fig. 2. HTTPS Callouts for Assigning and Implementing Data in Modules

Users can decide where modules are stored, how they are stored, how they are programmed, and what positions they take up in the user interface as shown in Fig. 3. Users can implement this information using JavaScript Object Notation (JSON), which simplifies the process and removes the need for prior JavaScript knowledge. It is worth noting that the overall structure imitates a JSON database’s structure almost one-to-one.

```
modules: [
        {
                module: "alert",
        },
        {
                module: "updatenotification",
                position: "top_bar"
        },
        {
                module: "clock",
                position: "top_left"
        }
        …
]
```

Fig. 3. An Example of a Code Snippet for a Module Using the JSON Database Structure

Every user will have their own unique needs. While research has gone into the effectiveness of using a virtual assistant to improve efficiency, letting the adoption of a virtual assistant be non-compulsory is a great way to retain user autonomy to tailor to their needs in any program [16].

This implementation includes a virtual assistant with voice-activation and response to elevate ease of use. Being able to speak to one’s mirror to get a response, either contextually or otherwise, helps improve accessibility by enabling hands-free interaction, offering auditory feedback and simplifying tasks for users. When implementing a virtual assistant, this project chose to work with Google Assistant [17][18]. Furthermore, this project has modules to communicate with one another. For example, the microphone takes in voice input and commands and can change its display as needed.

To trigger the smart mirror for the “Jarvis” module, users can speak to the smart mirror, “Hey Jarvis!” and then follow

International Journal of Scientific Engineering and Science
Volume 8, Issue 11, pp. 99-103, 2024. ISSN (Online): 2456-7361

with a command. This keyword can easily be changed within the module itself, as seen in Fig. 4.

```
module: "EXT-Detector",
position: "top_left",
configDeepMerge: true,
config: {
        debug: false,
        micConfig: {
            recorder: "insignia",
            device: "plughw:1",
        },
        useIcon: true,
        touchOnly: false,
        detectors: [        {
            detector: "Snowboy",
             Model: "jarvis",
             Sensitivity: null
            },
         ]
    }
}
```

Fig. 4. Working with Hotwords / Keywords for Starting Google Assistant

The implementation of the detection module called Snowboy is a piece of software that receives and interprets microphone input and gives an output where necessary. Many other modules exist and are implemented in this nature, such as Google Assistant, daily stock displays for cryptocurrency, relevant or upcoming dates and holidays, meeting times, appointment times, news headlines, and more. There even exist modules whose purpose is to cycle through other modules. In the current implementation, this project uses the "Carousel" module, which cycles through relevant news headlines from the same day.

These are only a small number of modules and features that can be conceived, developed, configured, and implemented in Smart Mirror project. With a variety of tools, including the Raspberry Pi, this project is able to create a working smart mirror.

B. Hardware Approach, Equipment, and Programming

When implementing a module that uses voice-activated Google Assistant, it is important to have a tool that is able to supply the interface with your voice. One such implementation uses Snowboy, as previously mentioned.

In our hardware implementation, the Raspberry Pi is used to compute and run the Magic Mirror interface and store the database of modules. The Raspberry Pi comes with a variety of Universal Serial Bus (USB) ports that can be utilized with different tool inputs and outputs.

The Raspberry Pi also makes use of a High-Definition Multimedia Interface (HDMI) port to take full advantage of the multi-media facets available from the Raspberry Pi itself. The Magic Mirror software can output video and audio as well. This would be necessary for some functionalities, some of which we have implemented, including YouTube streaming and virtual assistant voice activated response.

The USB ports available in the Raspberry Pi are very helpful in making the project as modular as possible. With as many moving parts as there are involved in the making of a smart mirror, smart mirror technology can be simplified when the user has more control over what parts are used in what ways. With some initial setup defined in the module configuration file, the user can define what devices and indicate which port inputs are used by the Magic Mirror interface.

In this way, making the system as modular as possible allows the use of interchangeable parts through USB. The Raspberry Pi features USB 2.0, which will make only a subtle difference in speed when working with USB 3.0 devices. In any case, the user can specify what devices they would like to utilize, meaning less parts are hardwired.

If user's implementation does not require a camera, or a microphone, such things can be left off the list. No feature is built in, expected, or necessary when adapting this system to user's needs. Likewise, the microphone or other device used in implementation can always be switched out with another device if it proves superior to the user's current setup.

With the advent of print-on-demand circuit board services, even a breadboard can be replaced with a single chip, simplifying the internal infrastructure of the smart mirror and streamlining the overall design process. By utilizing a pre-made circuit board or similar service, the need for complex technical setups and time-consuming manual wiring can be eliminated entirely, enabling a more compact and efficient system. For this project, however, a breadboard was chosen, as using a circuit board would involve a potentially long production turnaround.

Fig. 5 shows a simple mirror configuration using an LCD screen display for the Smart Mirror project. This screen casing features a two-way mirror with a black/blue border to house the internal components. The figure illustrates how the software works in conjunction with the physical frame. The frame is constructed from several wooden pieces: two for the arms and one as a prop to ensure the display reaches the proper height for the two-way mirror. The two-way mirror allows display light to pass through while reflecting the user's face back towards them by blocking other light from passing through.

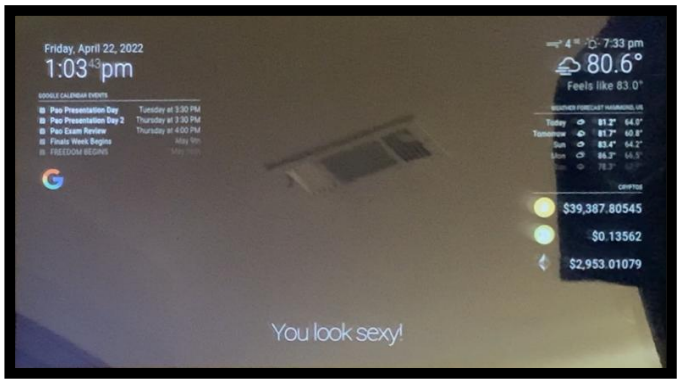

Fig. 5. LCD Screen Display for the Smart Mirror Project

The overall cost of this Smart Mirror project is extremely affordable. In the marketplace, a highly sophisticated smart mirror can cost hundreds of dollars. Additionally, since the parts used are interchangeable, there is an opportunity to reuse components from other technical projects or spare USB parts found around the house.

IV. EVALUATION

When testing our solution, this Smart Mirror project made several advancements that included testing individual components, then testing them in combination with their functionality counterpart in Magic Mirror and the Raspberry Pi hardware itself.

One issue that this Smart Mirror project encountered during its production was failing to initiate the mirror using Jarvis. As such, we drew our attention to the implementation. Snowboy was able to detect and decipher exactly what we were saying to the very last syllable. Clearly it was not an issue of interference, nor interpretation of input data. The issue must have lied elsewhere.

Thus, our attention was drawn to the voice-activated virtual assistant. When using the hot word "Jarvis," the program was not responding. But upon further examination of the modules, the hot word is always prefixed with "Hey," meaning the full key term to activate Jarvis was, in full, "Hey Jarvis." Upon saying this, instead of simply "Jarvis," we were able to show that the microphone was properly detecting input, receiving it in the Magic Mirror interface, porting it to Google Assistant, and properly processing the input data.

V. CONCLUSION

Despite some initial challenges, this Smart Mirror project is able to create a product that is affordable, accessible, and customizable. It serves as a testament that pursuing similar endeavors is entirely feasible even with minimal technical training. It is worth appreciating that creating a working smart mirror from scratch is no small feat. By outlining a clear series of steps, we hope to show that anyone willing to learn has the ability to take on this project and succeed.

While the project has achieved its goals, there are still areas that can be refined. One way to improve is by addressing the poor technical performance observed during extended use. In future research, we plan to explore ways to enhance performance by testing various module combinations and addressing software issues. Overall, we are pleased with the progress made on this project. In the future, we look forward to advancing this work, exploring new areas of research, and building upon the original concept with novel implementations designed to further enhance user convenience.

REFERENCES

[1] S. Mazumder, B. Bishnoi, and D. Patel, "News Headlines: What They Can Tell Us?" in *Proceedings of the 6th IBM Collaborative Academia Research Exchange Conference (I-CARE) on I-CARE 2014 (I-CARE 2014)*, Banglore, India, pp. 1–4, October 2014, DOI: 10.1145/2662117.2662121.

[2] K. P. Yang, P. McDowell, R. Demourelle, T. Parker, and E. Langstonirst, "3D Printing: A Custom-Built 3D Printer with Wireless Connectivity," *SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE)*, ISSN 2348 – 8387, vol. 7, issue 10, pp. 1-5, October 2020, DOI: 10.14445/23488387/IJCSE-V7I10P101.

[3] A. Mohamed, M. Ab Wahab, S. Suhaily, and D. Arasu, "Smart Mirror Design Powered by Raspberry PI," in *Proceedings of the 2018 Artificial Intelligence and Cloud Computing Conference (AICCC '18)*, Tokyo, Japan, pp. 166–173, December 2018. DOI: 10.1145/3299819.3299840.

[4] K. P. Yang, A. Burningham, K. Champagne, D. Dahal, and J. Hernandez, "A Venture to Build a CNC Machine," *International Journal of Computer & Organization Trends (IJCOT)*, ISSN 2249-2593, vol. 9, issue 5, pp. 5-9, October 2019, DOI: 10.14445/22492593/IJCOT-V9I5P302.

[5] J. Brock, R. Bruce, and M. Cameron, "Changing the World with a Raspberry Pi," *Journal of Computing Sciences in Colleges*, vol. 29, issue 2, pp. 151–153, December 2013.

[6] K. P. Yang, N. Moran, I. Bendana, S. Champagne, and T. Becker, "LOPEZ: A Bilingual Robotic Car," *International Journal of Research in Advent Technology (IJRAT)*, E-ISSN 2321-9637, vol. 4, issue 12, pp. 51-55, December 2016.

[7] K. Jin, X. Deng, Z. Huang and S. Chen, "Design of the Smart Mirror Based on Raspberry PI," *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Xi'an, China, pp. 1919-1923, May 2018, DOI: 10.1109/IMCEC.2018.8469570.

[8] S. Sahana, M. Shraddha, M. Phalguni, R. Shashank, C. Aditya, and M. Lavanya, "Smart Mirror using Raspberry Pi: A Survey," *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, pp. 634-637, April 2021, DOI: 10.1109/ICCMC51019.2021.9418408.

[9] A. Olowolayemo, S. Alenazi, and F. A. Syairah Seri, "Mirror that Talks: A Self-Motivating Personal Vision Assistant," in *Proceedings of the 2018 International Conference on Image and Graphics Processing (ICIGP 2018)*, Hong Kong, pp. 157–161, February 2018, DOI: 10.1145/3191442.3191476.

[10] S. Hussain, S. Deepalakshmi, R. Benilla, and V. Nivetha, "Automation of Smart Home using Smart Phone via Google Assistant," *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, pp. 499-505, January 2023, DOI: 10.1109/ICSSIT55814.2023.10060979.

[11] K. Sengupta, S. Sarcar, A. Pradhan, R. McNaney, S. Sayago, D. Chattopadhyay, and A. Joshi, "Challenges and Opportunities of Leveraging Intelligent Conversational Assistant to Improve the Well-being of Older Adults," In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*, pp. 1–4, April 2020, DOI: 10.1145/3334480.3381057.

[12] A. Hippocrate, E. Luhanga, T. Masashi, K. Watanabe, and K. Yasumoto, "Smart Gyms Need Smart Mirrors: Design of a Smart Gym Concept through Contextual Inquiry," in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers (UbiComp '17)*, Maui, HI, USA, pp. 658–661, September 2017, DOI: 10.1145/3123024.3124427.

[13] A. Johri, S. Jafri, R. Wahi, and D. Pandey, "Smart Mirror: A Time-Saving and Affordable Assistant," *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, Greater Noida, India, pp. 1-4, December 2018, DOI: 10.1109/CCAA.2018.8777554.

[14] M. Seraphina Astriani, A. Kurniawan, and N. Qomariyah "COVID-19 Self-Detection Magic Mirror with IoT-based Heart Rate and Temperature Sensors," *2021 2nd International Conference on Innovative and Creative Information Technology (ICITech)*, Salatiga, Indonesia, pp. 212-215, September 2021, DOI: 10.1109/ICITech50181.2021.9590150

[15] K. P. Yang, P. McDowell, P. Dolan, C. Otts, G. Chenevert, and C. Tunstall, "SelfieBot: A Robot to Detect, Photograph, and Tweet Smiles," *International Journal of Engineering Research & Technology (IJERT)*, ISSN 2278-0181, vol. 8, issue 10, pp. 387-390, October 2019, DOI: 10.17577/IJERTV8IS100241.

[16] L. Cheng, C. Wilson, S. Liao, J. Young, D. Dong, and H. Hu. "Dangerous Skills Got Certified: Measuring the Trustworthiness of Skill Certification in Voice Personal Assistant Platforms," in "*Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*," pp. 1699–1716, November 2020, DOI: 10.1145/3372297.3423339.

[17] D. Sheppard, N. Felker, and J. Schmalze, "Development of Voice Commands in Digital Signage for Improved Indoor Navigation Using Google Assistant SDK," *2019 IEEE Sensors Applications Symposium (SAS)*, Sophia Antipolis, France, pp. 1-5, March 2019, DOI: 10.1109/SAS.2019.8706120.

[18] K. P. Yang, A. Jee, D. Leblanc, J. Weaver, and Z. Armand, "Experimenting with Hotword Detection: The Pao-Pal," *European Journal of Electrical Engineering and Computer Science (EJECE)*, ISSN 2506-9853, vol. 4, issue 5, pp. 1-5, September 2020, DOI: 10.24018/ejece.2020.4.5.246.