MSc Project Specification

Jarvis Mirror

Submitted to

Dr. Othon Michail

Mr. Nikolaos Theofilos Sobrino

Miss Tansholpan Zhanabekova

Submitted by:

Mulla, Mehvish (201683918)

Devanga A, Vineela (201680186)

Bhatt, Balkrishna (201673048)

Mathew, Christeena (201631941)

Sara Sabu, Angeline (201684974)

Pinheiro, Avelin Mary (201648465)

1. INTRODUCTION

According to an online survey named WGNA, women spend an average of 6.4 hours/week while men spend 4.5 hours/week working on their appearances towards the mirror. Furthermore, for daily use, another survey was carried out and it was found that men spend 56 minutes every day in front of a mirror and women dedicate 55 minutes a day to their appearance. Thus, it is essential that the world improves time management and productivity, together with utilizing new, fast-evolving technologies. It is also critical to provide technology-savvy advantages to individuals with disabilities [1]. As a result, this Jarvis mirror idea was created to give access to the information that is provided by the user in a convenient and time-saving manner.

1.1 AIM AND OBJECTIVES

As a result of this project, we have designed a prototype to be able to detect the user's voice by speaking "Jarvis!" and create a mirror that can display the generated output, and therefore act as a "Jarvis Mirror". As a result of this device, customizable information can be displayed on the mirror. It is an electronic device that can act as both a mirror and a display device when given user queries. For example, if the user commands "tell me a joke", the mirror will display the joke on the mirror and the speaker will speak the joke while the text is displayed on the mirror.

1.2 KEY LITERATURE AND BACKGROUND READING

Several efforts have been made till date to develop a smart mirror in every field starting from household chores, retail use and health platform. Few such papers that we came across during our research phase gave us an idea of how we can explore the features of smart mirrors not only for the entertainment or shopping world but also for workplace health promotion as mentioned by Gomez-Carmona, O et al, in [2] and Alboaneen, D.A., et al [3]. Initially when we started with our project, we only had an idea to test the system with respect to text and audio but going ahead by referencing the strategy used by M. M. Yusri et al., in [4], we divided our test scenarios into unit and system testing which made our system more robust against any kind of testing. To integrate the NLP and google speech to text we have been going through multiple resources but a paper by Bisong, E.,[5] and Shakhovska, N.,[6] explicitly caught our attention due to its easy understanding on the topic. Managing open-source UI was not our first idea but Kredpattanakul, K et al.,[7] made the UI concept easier and more pocket friendly by explaining how we can use electron web app in our project that played a major role. We faced difficulty in searching ways for giving the “Jarvis” invoking command unless we came across an interesting paper by Burbach, L., et al [8]. We have also integrated with google maps api for navigation of routes and for weather we got a brief idea from Du, H., et al [9]. We dug in multiple resources to ensure our system has unique features along with all properties that are existing but in different systems, this makes our system a one-point product for all minor to major needs in this advancement of the tech savvy world.

2. JARVIS MIRROR

2.1 DEVELOPMENT AND IMPLEMENTATION SUMMARY

The JARVIS Mirror is a smart mirror concept that enables the user to utilize an interactive mirror that is hands-free and provides a one-station access to features like news, weather, stock news and entertainment like jokes. This section deals with the hardware and software aspects that are utilized to construct this system.

Hardware:

1. An LCD Screen
2. An acrylic sheet
3. Network adapter (for Wi-Fi) wood frame boards
4. HDMI-to-VGA adapter
5. Speaker
6. Microphone

The wooden frame is built first, with corner brackets to hold the screen in place. The plexiglass sheet which has reflective coating on either side is placed within the frame, followed by the LCD monitor in such a way that the sheet is sandwiched between the wooden frame and the monitor [10]. Lastly, the Raspberry Pi microcontroller is connected to the monitor through HDMI cable [11].

Software:

1. Speech-to-text and Text-to-Speech API
2. Electron
3. Wit.ai
4. Python

- *Speech-to-text and Text-to-Speech API*

The primal method of interaction with this device is through speech, which is achieved using the Google text-to-speech and speech-to-text API. Speech-to-Text enables easy integration of Google speech recognition technologies into developer applications. It helps to send audio and receive a text transcription from the Speech-to-Text API service [12].

To incorporate Google API features into the project, the API configuration needs to be set up initially. The Speech-to-text feature is to be enabled on the Google Cloud Platform (GCP) project. Followed by setting up authentication variables for the environment that it is to be used in. As the project can require storing records like audio files and such, optionally we can set up Google Cloud Storage bucket as a storage functionality.

- *Construction of a speech-to-text request*

The input audio consisting of the user's command will go through as a recognition request to the API [13].

There are several kinds of config fields, namely -

Mandatory fields: Encoding, sampleRateHertz, languageCode,

Optional fields: maxAlternatives, profanityFilter, speechContext etc

1. Encoding: Used to specify the encoding of supplied audio such as FLAC or LINEAR16.
2. sampleRateHertz: Mentions the sample frequency rate of audio input. Allowed values are between 8000Hz to 48000Hz, although here 16000Hz is used as a moderate value.
3. languageCode: Used to indicate language and region. e.g., 'en-US'
4. maxAlternatives: Number value representing number of alternate transcriptions of audio.
5. profanityFilter: Boolean value to indicate whether to filter out certain words.

The audio file is sent through the audio parameter, which can consist of one of the below fields:

1. Content: Contains the audio sample file.
2. URI: Indicates the location of the audio sample in the Google Cloud storage.

- *Speech Recognition model:*

The machine learning model that will be used to analyze the user's speech and convert it to text is chosen as a parameter within the Recognition request as a member of RecognitionConfig object [14]. Google possesses trained machine learning models which can be used to transcribe different types of audio files from various audio sources. By specifying the audio source in the recognition request, the transcription can be improved. This machine learning is to be specified in the RecognitionConfig object for request. Examples: Latest Long, Video, Telephone call. For the case of this smart mirror, we intend to use 'command_and_search' model, as this is suitable for short audio clips which contains commands or search phrases.

- *Text-to-Speech audio response:*

The smart mirror replies back to the user's commands using a synthetic voice that imitates natural human speech. The input for this conversion will be the text file generated by the Speech-to-text convertor and its output will an audio file of type MP3 or LINEAR16. To create an audio file the 'synthesize' component of the API is to be invoked.

- *Electron as a User Interface Platform*

In this system, the user interface is built using the Electron software. The interface will be always displayed on the mirror while the product is performing some activity, such as when the user asks it some question like "How's the weather?" the screen displays that the system is actively listening to the command.

- *Wit.ai*

Wit.ai is an open-source framework with natural language processing capabilities that help developers to get structured data from chat or voice. For example, when a user asks the mirror some question like "Tell me a joke", Wit.ai will analyze the text and find the keyword 'joke', which will then be used to fetch the joke from the python library.

- *Python*

Python is a high-level, general-purpose programming language. Python is frequently used for creating websites and applications, automating repetitive tasks, and analyzing and displaying data [13]. Since Python is a relatively easy-to-learn programming language, we can easily create a backend server and REST APIs that allow frontend and backend users to communicate with one another.

2.2 MODULES

The modules which we have implemented are:

i. Joke

When the user asks the mirror "Tell Me A Joke" the mirror will fetch the joke feeds that will appear on the top list and the joke will be displayed on the mirror as well as the speaker will broadcast the joke and the user will be able to hear it. Google speech-to-text convertor is used to generate text which is then passed to Wit.ai. Wit.ai will analyze the text and find the keyword 'joke', which will then be used to fetch the joke from the python library.

ii. Jarvis

This is the module that is used for activating the mirror. When the user stands in front of the mirror and says the hotkey "JARVIS!", the mirror is triggered and responds to user by greeting them like "Good Morning Tony".

iii. Stock price

In this module we have used an open-source stock price API - EODHD API to fetch the stock price whenever the user needs it. Different case scenarios like "High", "low", "open" and "close" will also be displayed.

iv. Weather

In this module the user will be able to fetch the weather. The weather will be available for every specific city [15]. The user will also be able to fetch the forecast for the next three to seven days [16]. Here we are using meteomatics open source api key to get weather data. Meteomatics offers sample code and open-source Python modules for the easy retrieval of any type of weather data. According to the REST standard, data can be queried using HTTP-GET or HTTP-POST requests from api.meteomatics.com. A request can be constructed using the format:

https://api.meteomatics.com/<validdatetime>/<parameters>/<location>/<format>?<optionals>

Example:
https://api.meteomatics.com/2022-11-09T00:00:00ZP1D:PT1H/t_2m:C,relative_humidity_2m:p/47.4245,9.3767/html?model=mix

v. Reminders

One can ask the mirror to display reminders for any day, following which the reminders would be displayed on the mirror and the same would be conveyed through the speaker. Google Calendar API will be integrated to Electron user interface, and this will enable users to view reminders [17]. When the user asks "Show me reminder" - the Google speech to text convertor and wit.ai parsing will be used to detect and analyze the user request.

vi. Maps

This is the module in which, when the user asks for the navigation, the mirror will display the maps of specific types like "Hybrid", "Terrain" and "Satellite". The mirror will also display the map of a specific city [18]. We use google maps API to attain this feature.

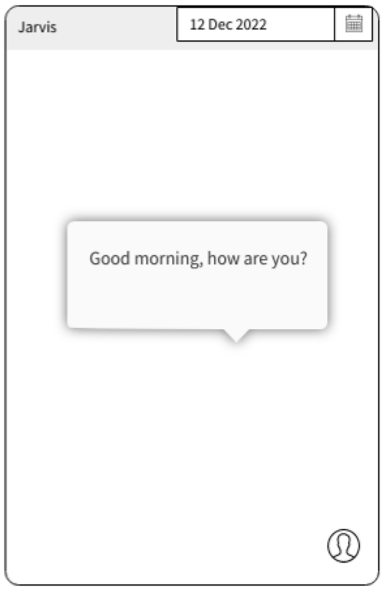
vii. Currency Converter

An open-source API called fixer API is used for currency conversion. We will be sending an API request containing the base currency and the desired currencies. This request is then processed by the fixer API and then a JSON output will be generated which is then converted to the desired output via python.

2.3 USER INTERFACES

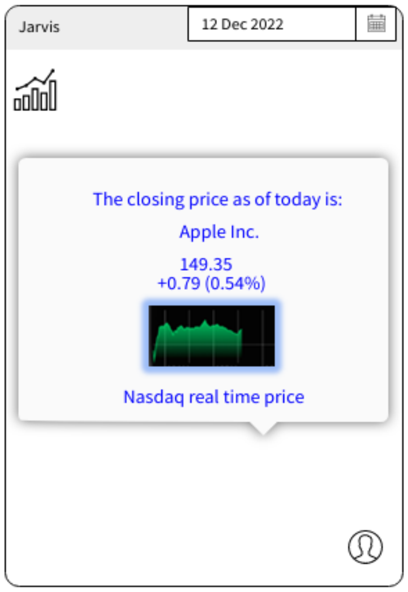
User interfaces for few interactions with the mirror are shown below.

i. Start interface:



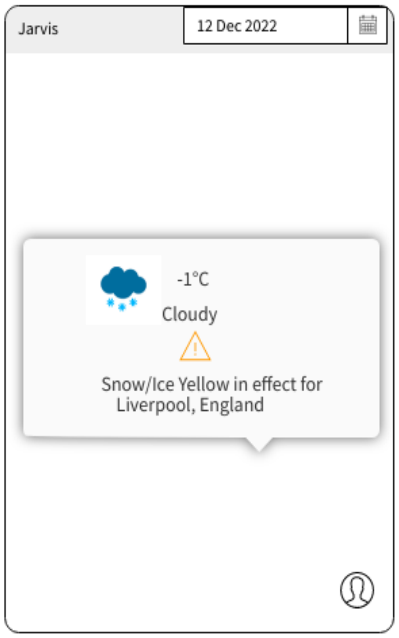
Good morning, how are you?

ii. "What was today's closing price for Apple stock?"



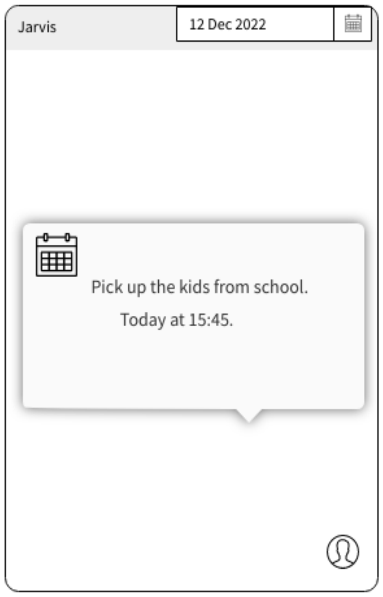
The closing price as of today is:
Apple Inc.
149.35
+0.79 (0.54%)
Nasdaq real time price

iii. “How’s the weather looking today?”



-1°C
Cloudy
Snow/Ice Yellow in effect for Liverpool, England

iv. “Remind me what I need to do today? ”



Pick up the kids from school.
Today at 15:45.

2.4 TESTING

Jarvis Mirror is not only a software but a product and we have put our thoughts together to design test methodology and test cases to test our system. We have used Unit testing to assess the integration points of the user interface of the open source web app: Electron [4]. We have chosen google chrome to perform this unit testing as our compatible browser. Below is the table that has 3 columns namely, modules - This is the category under which there are a variety of cases to be tested. Test cases- These are the number of cases according to which we need to test our module. Expected results - This is how our final developed system must behave.

Table 2.4.1 shows the unit test cases to be performed:

Module	**Test Case**	**Expected Result**
Electron	User friendly design of web page,	Content on Electron webpage is

	easy to understand	clear and in correct order and the user can understand things like the mirror is connected, etc.
Compatibility	Check Mirror Pi Config compatibility with Electron	Mirror connects to Electron and user can see it active and running, hence compatible

Table 2.4.2 shows System testing, this is used by testers to evaluate the functional criteria of Jarvis Mirror:

Module	**Test Case**	**Expected Result**
Running Jarvis	System is in running mode when invoked by “Jarvis”	System is running and we can see light on the edges of the mirror screen.
Audio Input	Mirror takes audio input to process further	Audio input like “Tell me a joke” is taken by mirror and forwarded to wit-ai and then speech to text API respectively
Audio Output	Mirror responds to the audio input.	Mirror processes the input as mentioned in the above step and then responds by fetching processed data to give audio output.
Display Text	Mirror displays text of basic information.	System fetches the processed data and displays it on the mirror.
Correctness of data	Display Information Correctness	Information displayed on the screen such as date, time, day, temperature, etc is correct.

2.5 EVALUATION

The tests performed were divided into Unit testing and System Testing. This evaluation is done by the testers in our group. In unit testing the system integration with the UI platform is found clean and understandable. In system testing the evaluation criteria includes running the system, checking audio input and output for Correctness/Reliability of the system by giving confusing commands repeatedly to check whether it understands the command and responds to correct keywords. System is also evaluated for stability by testing the display information only when invoked. We also evaluated that there are still grounds to improve in terms of adding more features such as AI for gestures so that it can be used for industrial retail purposes as well.

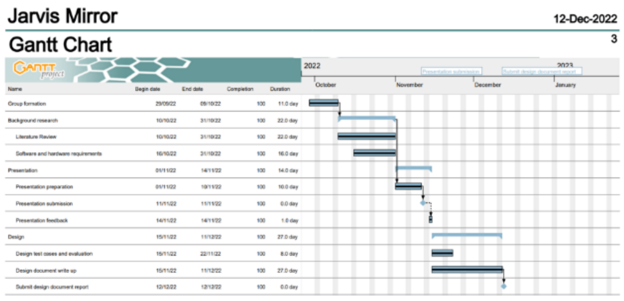
3. ETHICAL CONSIDERATIONS

We consent that we have read and understood the ethical guidelines and followed them during the project this includes the use of data in the public domain, the generation of new data, and the results of testing and evaluation.

4. PROJECT PLAN

Time	Hierarchy	Activities
Weeks 1–3	1	Group formation
Weeks 3–6 Weeks 3–6 Weeks 4–6	2 2.1 2.2	Background research Literature Review Software and hardware requirements
Weeks 6–8 Weeks 6–7 11 November Weeks 8	3 3.1 M 3.3	Presentation Presentation preparation Presentation submission Presentation feedback
Weeks 8–11 Weeks 8–9 Weeks 8–11 12 December	6 6.1 6.2 M	Design Design test cases and evaluation Design document write up Submit design document report

- Here 'M' indicates a milestone.
- 1.1 and 1.2 are subtasks of 1, similarly, 2.1–2.4 are subtasks of 2 and 3.1–3.4 are subtasks of 3.
- Tasks 2, and 3 depend on tasks 1, and 2 being completed, respectively.
- Milestone "Presentation submission" and Submit design document report" depends on subtask 3.1 and subtask 6.2 being completed.


Jarvis Mirror 12-Dec-2022
Gantt Chart 3

Name	Begin date	End date	Completion	Duration
Group formation	29/09/22	09/10/22	100	11.0 day
Background research	10/10/22	31/10/22	100	22.0 day
Literature Review	10/10/22	31/10/22	100	22.0 day
Software and hardware requirements	16/10/22	31/10/22	100	16.0 day
Presentation	01/11/22	14/11/22	100	14.0 day
Presentation preparation	01/11/22	10/11/22	100	10.0 day
Presentation submission	11/11/22	11/11/22	100	0.0 day
Presentation feedback	14/11/22	14/11/22	100	1.0 day
Design	15/11/22	11/12/22	100	27.0 day
Design test cases and evaluation	15/11/22	22/11/22	100	8.0 day
Design document write up	15/11/22	11/12/22	100	27.0 day
Submit design document report	12/12/22	12/12/22	100	0.0 day

5. RISKS AND CONTINGENCY PLANS

Risk	Contingencies	Likelihood	impact
Software failure	ensuring every function is tested properly	low	results might not be exact or results might not be out
Programming fault	Checking if the software is written correctly	medium	wrong results is be displayed
mic & speaker issues	installing high definition	low	user's information maynot be considered exactly
Acrylic sheet	ensuring every function is tested properly	low	Display of the results would be affected
Wi-Fi error	making sure that there is no disruption in Wi-Fi connection during the interaction	medium	User might have to restart his his/her request

6. REFERENCES

[1] Chaparro, J.D. et al. (2021) ‘The SHAPES Smart Mirror Approach for Independent Living, Healthy and Active Ageing’, Sensors (Basel, Switzerland), 21(23), p. 7938–. Available at: https://doi.org/10.3390/s21237938

[2] Gomez-Carmona, O. and Casado-Mansilla, D., 2017, July. SmiWork: An interactive smart mirror platform for workplace health promotion. In 2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech) (pp. 1-6). IEEE.

[3] Alboaneen, D.A., Alsaffar, D., Alateeq, A., Alqahtani, A., Alfahhad, A., Alqahtani, B., Alamri, R. and Alamri, L., 2020, March. Internet of things based smart mirrors: a literature review. In 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS) (pp. 1-6). IEEE.

[4] M. M. Yusri et al., "Smart mirror for smart life," 2017 6th ICT International Student Project Conference (ICT-ISPC), 2017, pp. 1-5, doi: 10.1109/ICT-ISPC.2017.8075339.

[5] Kredpattanakul, K. and Limpiyakorn, Y., 2018, June. Transforming javascript-based web application to cross-platform desktop with electron. In International Conference on Information Science and Applications (pp. 571-579). Springer, Singapore.

[6] Burbach, L., Halbach, P., Plettenberg, N., Nakayama, J., Ziefle, M. and Valdez, A.C., 2019, July. " Hey, Siri"," Ok, Google"," Alexa". Acceptance-Relevant Factors of Virtual Voice-Assistants. In 2019 IEEE International Professional Communication Conference (ProComm) (pp. 101-111). IEEE.

[7] Du, H., Jones, P., Segarra, E.L. and Bandera, C.F., 2018. Development of a REST API for obtaining site-specific historical and near-future weather data in EPW format.

[8] Lee, S.-H. et al. (2022) ‘Accuracy of Cloud-Based Speech Recognition Open Application Programming Interface for Medical Terms of Korean’, Journal of Korean medical science, 37(18), pp. e144–e144. Available at: https://doi.org/10.3346/jkms.2022.37.e144.

[9] Shein, E., 2015. Python for beginners. Communications of the ACM, 58(3), pp.19-21.

[10] Colantonio, S. et al. (2015) ‘A smart mirror to promote a healthy lifestyle’, Biosystems engineering, 138, pp. 33–43. Available at: https://doi.org/10.1016/j.biosystemseng.2015.06.008.

[11] Bhadoria, S. and Oliva Ramos, R. (2017) Raspberry Pi 3 home automation projects: bringing your home to life using Raspberry Pi 3, Arduino, and ESP8266 / Shantanu Bhadoria, Ruben Oliva Ramos. 1st edition. Birmingham, England: Packt Publishing.

[12] Bianco, S. et al. (2021) ‘A Smart Mirror for Emotion Monitoring in Home Environments’, Sensors (Basel, Switzerland), 21(22), p. 7453–. Available at: https://doi.org/10.3390/s21227453.

[13] Hillar, G.C., 2018. Django RESTful Web Services: The Easiest Way to Build Python RESTful APIsand Web Services with Django. Packt Publishing Ltd.

[14] Schwarz, N., Johnson, K.A. and Babel, M. (2021) ‘Exploring the variable efficacy of Google speech-to-text with spontaneous bilingual speech in Cantonese and English’, The Journal of the Acoustical Society of America, 150(4), pp. A357–A357. Available at: https://doi.org/10.1121/10.0008580.

[15] M, R. et al. (2021) ‘IOT Smart Mirror with News & Temperature’, International Journal of Computer Science and Engineering, 6(1), pp. 19–25. Available at: https://doi.org/10.14445/23488387/IJCSE-V6I1P104.

[16] S. Athira, F. Francis, R. Raphel, N. S. Sachin, S. Porinchu and S. Francis, "Smart mirror: A novel framework for interactive display," 2016 International Conference on Circuit, Power, and Computing Technologies (ICCPCT), 2016, pp. 1-6, doi: 10.1109/ICCPCT.2016.7530197.

[17] Chaitanya, M.U. and Sunayana, M.K.V. (2020) ‘Voice Assistant and Security based Smart Mirror’, International journal of recent technology and engineering, 8(6), pp. 4054–4060. Available at: https://doi.org/10.35940/ijrte.F9464. 038620.

[18] ‘Smart Mirror’ (2019) International journal of recent technology and engineering, 8(2S11), pp. 925–929. Available at: https://doi.org/10.35940/ijrte.B1152.0982S1119.

[19] Bisong, E., 2019. Google automl: Cloud natural language processing. In Building Machine Learning and Deep Learning Models on Google Cloud Platform (pp. 599-612). Apress, Berkeley, CA.

[20] Shakhovska, N., Basystiuk, O. and Shakhovska, K., 2019. Development of the Speech-to-Text Chatbot Interface Based on Google API. In MoMLeT (pp. 212-221).

[21] Iancu, B., 2019. Evaluating google speech-to-text API's performance for Romanian e-learning resources. Informatica Economica, 23(1), pp.17-25.

[22] Chawathe, S., Dhakad, S., Sharma, R. and Ambadekar, S., 2019. Interactive Smart mirror. Int. J. Recent Technol. Eng., 6(6), pp.2083-2087.

[23] Ogunjimi, A. et al. (2021) ‘Smart mirror fashion technology for the retail chain transformation’, Technological forecasting & social change, 173, p. 121118–. Available at: https://doi.org/10.1016/j.techfore.2021.121118.

[24] Belinda, M.J.C. and Rupavathy, N. (2021) ‘Smart Mirror Using Rasberry Pi’, Turkish journal of computer and mathematics education, 12(9), pp. 190–194.

[25] Nadaf, R.A. et al. (2019) ‘Smart Mirror Using Raspberry Pi for Human Monitoring and Home Security’, in Advanced Informatics for Computing Research. Singapore: Springer Singapore, pp. 96–106. Available at: https://doi.org/10.1007/978-981-15-0111-1_10.

[26] Sophia Jasmine, G. et al. (2021) ‘IoT Based Voice Controlled Raspberry PI Smart Mirror’, in 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS). IEEE, pp. 1170–1173. Available at: https://doi.org/10.1109/ICACCS51430.2021.9441945.