IOT BASED SMART MIRROR

Project report submitted in partial fulfillment of the requirement of the degree

Of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

Shikhar Srivastava (161082)

Deepak Kumar (161090)

Somya Chaturvedi (161099)

UNDER THE GUIDANCE OF

Dr. Rajiv Kumar

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT

May 2020

TABLE OF CONTENTS

DECLARATION

We hereby declare that the work reported in the B.Tech Project Report entitled "**IOT BASED SMART MIRROR**" submitted at **Jaypee University of Information Technology, Waknaghat, India** is an authentic record of our work carried out under the supervision of **Dr. Rajiv Kumar**. We have not submitted this work elsewhere for any other degree or diploma.

Shikhar	Deepak Kumar	Somya
Shikhar Srivastava	**Deepak Kumar**	**Somya Chaturvedi**
(161082)	**(161090)**	**(161099)**

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Dr. Rajiv Kumar

Date:

Head of the Department/Project Coordinator

ACKNOWLEDGEMENT

We would like to thank **Prof. Dr. M.J Nigam**, Head of Dept. Electronics and Communication, for providing us the opportunity of working on a project.

We are highly indebted to **Dr. Rajiv Kumar** for motivating and enlightening us for our project work. We thank you for being a constant support throughout, without whose valuable guidance and insights, this project would not be a complete one. We offer you our sincere gratitude to you for instructing and directing us through thick and thin.

We would also like to acknowledge **Mr. Abhishek Ray** for helping us in the labs. Thank you for being there and helping us in and out.

ABSTRACT

According to survey conducted by TODAYAOL, women spend an average of 6.4 hours in a week while men spend 4.5 hours/week working on their appearances. While spending some of the minutes of those 4.5 hours we came up with an idea, what if mirror can talk to us? Like fairy tales, it can tell what's going on in the world? What if it can recognize who I am?

Smart mirror is an idea to bring technology and imaginary world into real world, to save time and to meet the technological advancements.

Main motive is to make livelihood easier and smooth by the help of technology.

The "Smart Mirror" consists of an electronic display along with a one-way mirror for the informational display. It uses a camera to track movement and provide facial recognition with the use of an embedded computer. Users can create their customized display through a smartphone interface. Some informational displays contain information aggregated from third party APIs, such as social media feeds, news feeds, and weather updates.

Solutions proposed: -

1. Time saving for corporate people
2. Check notifications by just standing in front of mirror
3. Can be used in official receptions.

You can join your android and iOS phones

CHAPTER 1

1.1 INTRODUCTION

The aim of this project was to design and make a device that acted as a "Smart Mirror" by "displaying the user's image and providing customizable information on the display". A "Smart Mirror" is a smart device in the smart world that not just work as a traditional mirror but also superimpose informational data, the user can customize which. The mirror also gives experience of touch free user interaction in addition of some data displays. Users can make a profile and change the visual interface of the display about the specific data feeds they want.

The "Smart Mirror" comprises an informatics display along with a one-way mirror for the electronic display. It uses a camera to track the user and feature the facial recognition with the use of an embedded computer. Users can create their customized display through a Mobile interface. The informational display comprises the third party API used for social media, weather and other uses.

1.2 MARKET ANALYSIS

- With the growing demand of technology in the market and between people there is a high demand of artificial intelligence and life easing products. These types of products usually cost people a good pocket losing amount with less features.
- We have a more reliable and cost-effective product then other companies.
- Face recognition is there in the market, but they are quite expensive plus they do not give you direct linkage from your mobile phones.
- We will be providing more technology in fewer prices.

For example, Samsung has produced a smart mirror called Samsung Smart Mirror, which uses Intel's technology. This technology is groundbreaking and is not easily procurable or affordable for this implementation. Designs offer informational widgets, however, they lack the multiple informational screens, gesture support, and multiple user configurations and new applications like WhatsApp and LinkedIn aren't directly connected.

1.3 REQUIREMENT AND SPECIFICATION

Marketing Requirements

1. The system can display widgets.
2. The system can detect the user.
3. The system can navigate the UI based on motion of user.
4. The system has lower costing than the existing designs.
5. The system can work as a normal mirror.
6. The system looks charmingly appealing.
7. The system has a re-programmable UI.
8. The system can make a store for user accounts and preferred applications.
9. The system can begin usage only when the user needs it.
10. The system can access social media, which attracts user.

Concept Selection

All products inside the smart mirror market are too high priced or confined to wellknown hobbyists. For instance, Samsung has produced a smart replicate referred to as Samsung Smart Mirror, which uses Intel's RealSense era. This era is groundbreaking and is not without difficulty procurable or inexpensive for this implementation. Designs provide informational widgets; however, they lack the multiple informational screens, gesture assist, and more than one consumer configurations. The following desk shows numerous clever reflect implementations that were researched.

1.4 RESEARCH ON EXISTING SMART MIRRORS IN MARKET

Device Name	Features	Price	Usage & Notes	Website
Magic Mirror	Informational	Open-Source	Hobbyist project	http://michaelteeuw.nl/tagged/magic mirror
“Samsung Smart Mirror”	3-D camera, gesture controls, voice control	Not listed yet; “very costly” according to interviews	Expensive luxury device, out of price range for typical households	http://www.businessinsider.com/Samsung-smart-mirror-photos-2015-6
TecH20 Televisions	Displays that also function as mirrors	$1000-10000	Only the mirror display component; no UI is involved with the device	http://www.tech2o.tv/

Options to the "Smart Mirror" can be repeat utilizing a Mobile Phone or PC. These gadgets can show educational applications and make the client to connect through a touchscreen. The cameras, on these gadgets take into consideration facial acknowledgment however won't have signal acknowledgment in light of the accessibility of a profundity camera. The advantage of the "Smart Mirror" signal control is that the client need not contact the screen, henceforth taking out different contacts in the pandemic occasions of corona virus. Cell phones and Laptops have little screen estimate and can not mirror that great as a completely utilitarian mirror. In spite of the fact that, these gadgets can utilize voice acknowledgment for the quest demand process for the client without even truly collaborating with the gadget.

CHAPTER 2

2.1 DESIGNING

The level 0 design of the Smart Mirror can be seen Figure 2.

Module	"Smart Mirror"
Inputs	"Wireless Internet Kinect Camera/Microphone Output" 220V AC, 50Hz
Outputs	Reflectives widgets display.
Functionality	Informational widget sent to the display allow the user to interact with through the camera and microphone

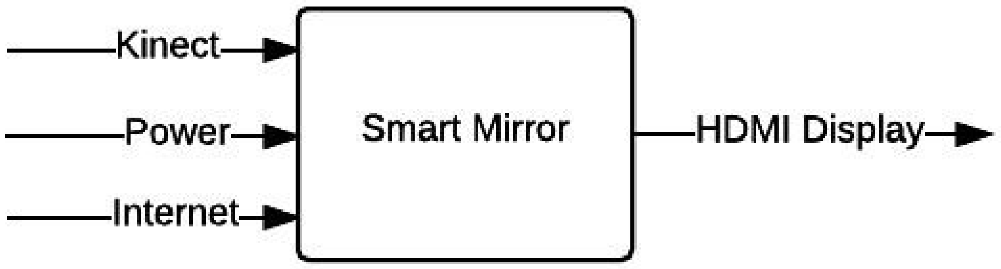


Figure 2 – "Level 0 Design of Smart Mirror"

The level 1 designed "Smart Mirror" see in Figure 3.

Module	“Smart Mirror”
Inputs	Wireless Internet 220V AC, 50Hz
Outputs	Visual Display
Functionality	The embedded computer’s camera and microphone data from, Kinect to process user interactions while informational widgets are displayed

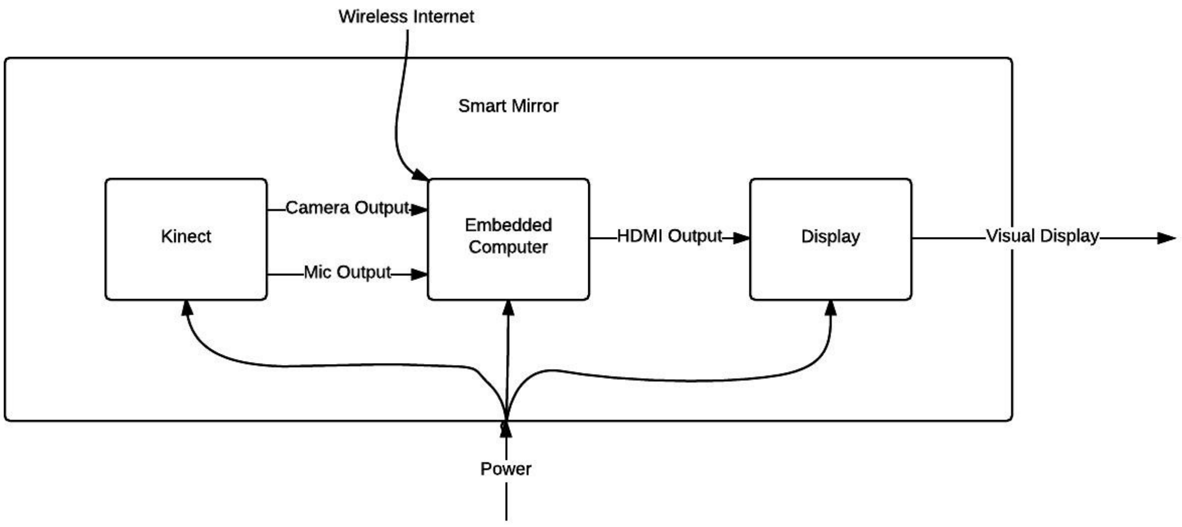


Figure 3 – “Level 1 Design of Smart Mirror”

2.2 MORPHOLOGY: SMART MIRROR

Based on the research, there is no competition in the market that matches the functionality that our Smart Mirror has, for the same cost. To reach the desired functionality within the limitation of our budget, we divided the Smart Mirror into five principal components that were investigated and selected.

One of the fundamental components of the Smart Mirror is the relevant tool with which we control the display and digital camera interface. Central devices encompass an embedded computer, tablet, laptop, or a disassembled pc. There are quite some factors that govern if the device is really worth using: three-D processing skills, costing, and simplicity in improvement, speedy computational energy, and tool interface skills. We in particular focused on the overall value of the device. Making a computing device would use a majority of the finances and consequently lessen the investment, could be used on making a higher show or using a doubtlessly higher satisfactory camera. Taking in account a disassembled laptop also might make it hard to duplicate the project. Based on these, we deemed the embedded laptop the nice device to be used.

So once we determined the outer hardware idea, we looked for five embedded computers in depth. “The Raspberry Pi, Banana Pi, ODROID C1, UX4, and BeagleBone Black” were all taken into account for the “Smart Mirror’s” embedded computer, but we ultimately chose the “Raspberry pi 3B” ultimately. The “Odroid XU4” was initially chosen for embedded computer, but after initial tests with the ‘Kinect’, it was determined it was less efficient in power. The other embedded we compared computers to the Raspberry Pi as the point of reference. The Banana Pi is comparatively more expensive, and one of the few differentiating features is the built-in Wi-Fi, which is a crucial component for the Smart Mirror’s functionality. Taking account to the BeagleBone, Pi 3B was more powerful graphics processing power, which is important for the Smart Mirror’s display. The C1 has better performance compared to the Pi 3B, but a little increase in performance did not prove to be a large enough upgrade.

The UX4 is much more expensive even if it was much more powerful than the other systems. The increased cost was worth it because of the device’s overall performance interfacing with the Kinect ballooned. We chose ubuntu as the embedded computer OS because of the ease of development on the platform. Refer to Figure 4 which depicts the total number of components that have to be connected to the embedded computer.

.

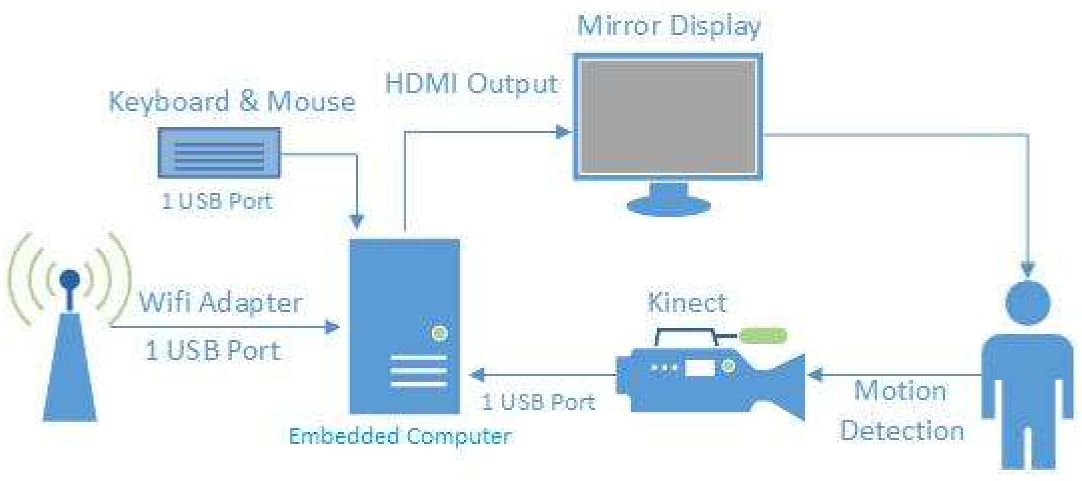


Figure 4 – "Devices Interfacing with Embedded Computer"

Based on the illustration, at least it requires two USB ports to get the "Smart Mirror" usable for a user without a keyboard and mouse. The mirror's display is connected through HDMI cable to prevent having to buy an HDMI to VGA/DVI adapter. It also requires an outrageous display resolution is also, it requires two USB ports meaning graphics processing capabilities are of high importance.

Camera

We chose the Microsoft Kinect for the camera aspect of the Smart Mirror because motion detection using infrared technology seemed like the practical choice instead of trying to work with a webcam. Between the two depth sensors, we chose the 'Kinect' over the Asus Xtion PRO LIVE. Even though the ASUS sensor is lighter and does not require an external power supply, the Kinect is more powerful and has a larger capture range. We tested the cameras in the following Pugh chart.lighter.

We did typical motion detection using a camera with a depth sensor. The Microsoft Kinect used an RGB camera with a depth and motion sensor and an infrared projector with a CMOS sensor, which can create a 3D environment. We sent the images captured from the RGB camera and depth sensor to the embedded computer through the USB 2.0 and interpreted by using a software library to perform facial recognition or to detect motion control. Such image processing requires about 50% of the USB 2.0 throughout, around 20 MB/s to send the frames to our embedded computer at resolving 640x480 and 30 frames per second. This camera, with depth capabilities, allows the user to control the UI seamlessly without the use of external components such as a keyboard or mouse. We can see the block diagram of the camera output to in Figure 5, and we can track the level 0 design of the camera in Figure 6.

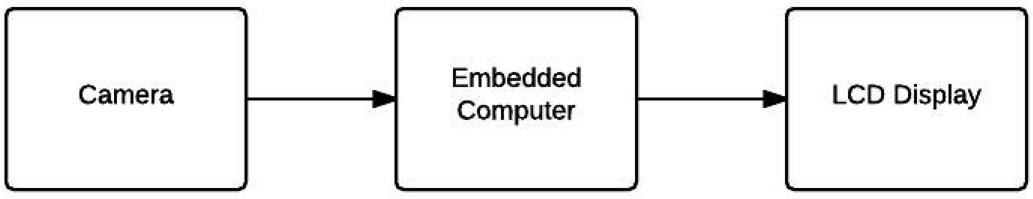


Figure 5 - Camera Interaction Explained

Module	Camera
Inputs	RGB Camera Input Depth Camera Input 120V AC, 60Hz
Outputs	Infrared Light USB to send camera frames
Functionality	Collects RGB and Depth image to send over USB, to be interpreted by another Device.

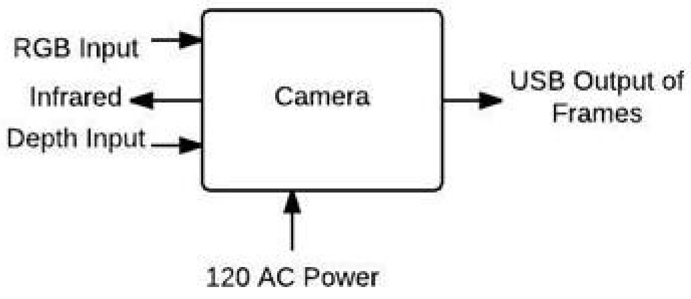


Figure 6 – "Level 0 Design of Camera"

2.3 WEB APP

A major component of the "Smart Mirror" system is the web application that let the user interaction possible. "Ruby on Rails, Django, and Tapestry5" were taken in account as web application frameworks. We worked on each web application framework on its license terms, customer base size, customizability, and availability of testing and security frameworks built in the web application framework.

"Ruby on Rails" is a wide range of choice in web app development. It can make use of the model view controller (MVC) design pattern. We base this web app. framework on the convention and configuration paradigm, which creates a well-planned layout and take down the number of decisions the developer can make regarding the code. The lack in the design flexibility makes this framework not ideal.

"Django" is another well-known web development framework, written in Python. It is based on the model template view pattern, very similar to MVC. An intriguing feature of this framework is that it let the developer more control in choosing the layout, a configuration in the design, which is important for the "Smart Mirror" system.

A "Pugh analysis" was used to choose the web application framework. Ultimately, the Django framework was chosen after the comparison. See the following table for the Pugh chart.

	Ruby on Rails	**Django**	**Tapestry5**
Freeware Software	1	1	1
User Base	0.85	0.85	0.3
Customizable - 0.25	0.4	0.75	0.3
Testing Frameworks - 0.15	0.7	0.8	0
Security Frameworks - 0.15	0.6	0.6	0.5
Total	0.7075	0.81	0.425
Continue	No	Yes	No

We base the web app on a server and client model. We hosted the server component on the embedded computer, and the client component is on the mobile device. The purpose of using this model is to designate the functionality for creating a user account and adjusting the display preferences of the widgets to a separate component. Given this functionality separation, the major component's purpose will log into an existing account and API usage. See Figure 7 for the server-client model diagram.

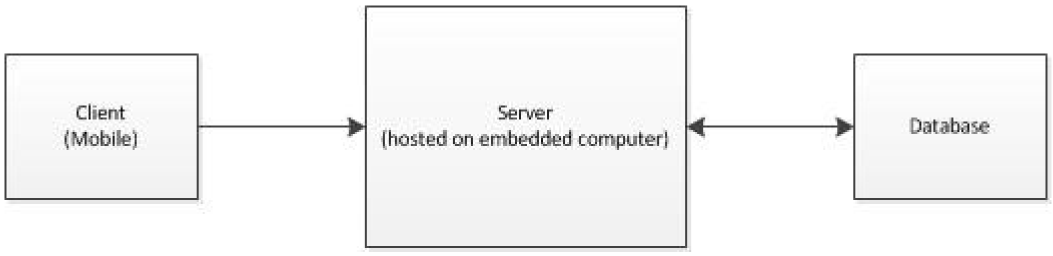


Figure 7 - Server-Client Model of Web Application

Django supports a lot of database servers such as Oracle, MySQL, and SQLite. Although choosing SQLite would have been one of the easier options because it does not require a personal server to perform its operations, it is an embedded database, with no built-in network capabilities. With this factor, we took MySQL as a better choice. MySQL can easily handle top end memory usage and perform smoothly when there are many ongoing processes.

2.4 APIs

The re-programmable presentation is predominantly founded on incorporating the APIs from various web sources and organizations to make "mirror gadgets." These applications furnished us with both real-time information and web-based social networking association. We firmly investigated an assortment of APIs to comprehend about the usefulness they give us, and any hazard factor in the event that we actualized them.

Twitter

Twitter is a long range interpersonal communication site that is utilized to share pictures, recordings, and content based messages. We got the Twitter API for nothing for designers and all it needs is just a Twitter record to demand an improvement key. Twitter got a medium-sized client base and offers the capacity to post your compositions, including pictures from the API to the web based life website. Twitter likewise ensures for engineers to pull the substance of a client's Twitter channel and show its data. The Twitter API's usefulness has gone down previously, yet hasn't altogether changed help. Twitter helps engineers by giving an online API testing console.

Instagram

Instagram is an internet based life site that is utilized to associate with your companions and offer pictures, recordings. Since the Instagram API is for nothing of cost for engineers and it requires all that solitary a Facebook record to demand an improvement key. Instagram has a medium-sized client base and offers the capacity to pull the pictures and recordings from another client's Instagram feed and show it from the API. Instagram doesn't permit pictures to be posted from anyplace other than its application. This cuts down the advantages that the Instagram API gives to its client. The Instagram API, is utilized to permit this usefulness, yet we have still broadened it. Instagram helps designers by giving an online API comfort. In any case, for API changes, Instagram usefulness wasn't executed.

Google Feed

Google Feed is totally free for engineers and doesn't require any record or key. The API gives requirefunctionality to a "RSS channel", with an elevated level of customization. A RSS channel can take information from numerous sites, which make it request to many "Smart Mirror" clients. Be that as it may, Google Feed API has been is not, at this point upheld. It would be not savvy to give a lot of need to Google Feed execution hence.

Feedly

Feedly is completely liberated from cost for designers and just requires a Feedly record to demand an improvement key. We displayed this API after the Google Feed API and has a serious same usefulness. Feedly just requires a client account not at all like Google Feed, but since the Feedly API has not been going down, it is a superior API to actualize.

OpenWeatherMap

"OpenWeatherMap" is a site that furnishes with the continuous climate data and estimates. "OpenWeatherMap" has a free API that just requires a record and obviously a key. This complementary plan offers a little specialized help and a less hearty climate data suite than paid levels, yet gives us fundamental climate data. "OpenWeatherMap" has a lifetime ensure that it will work on the present variant._

In the wake of social affair all the data about the APIs, we made the accompanying "Pugh diagram" to decide a need request for executing APIs. It positioned APIs on the significance of usefulness, ease in executing, wanted usefulness, client base size, and solidness in the API. Usefulness like picture sharing and ongoing data were considered progressively significant for the center usefulness of our mirror. The highlights are center and we should incorporate showcasing prerequisites or standard smart mirror usefulness that to contend with gadgets. APIs that are very much kept up and have given models are a lot simpler to actualize. The less chaotic an API is to execute; we invested greater improvement energy actualizing different APIs for usefulness. APIs that are enchanting to a bigger subset of clients are progressively wanted on the grounds that it will help drive reception of this item over contenders. At long last, the soundness of API is likewise significant in such a case that the present usefulness is changed or brought down, the "Smart Mirror" gadget won't work, decreasing the client experience. Figure 8 shows the automatic stream from a client order.

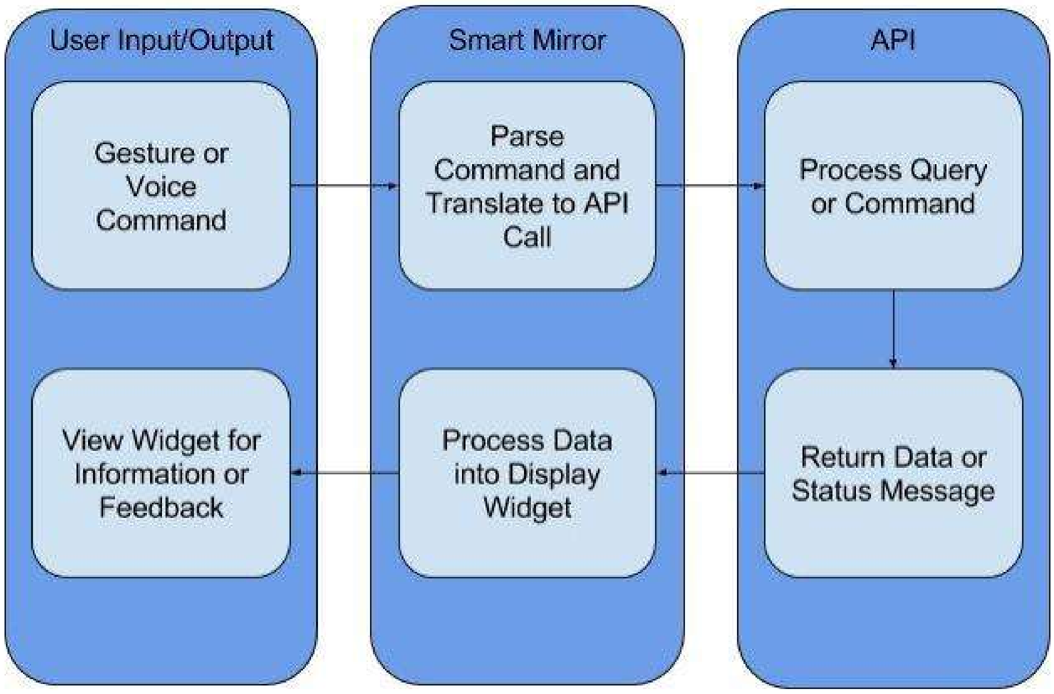

User Input/Output
Smart Mirror
API
Gesture or Voice Command
Parse Command and Translate to API Call
Process Query or Command
Return Data or Status Message
Process Data into Display Widget
View Widget for Information or Feedback

Figure 8 – "Program Flow"

Reflective Surface

At the point when we were choosing an intelligent surface to be utilized for the "Smart Mirror", there are too top of the line planning alternatives: an intelligent material that will go about as a single direction mirror/an intelligent single direction movie on a bit of glass. Intelligent film is much more less expensive and more effectively accessible than an intelligent acrylic or single direction glass, however is increasingly average to introduce. Since, it is less expensive and all the more effectively accessible, we picked the mirror film as the intelligent surface sort. We made the accompanying Pugh table to play out the correlation between intelligent surface sorts.

	Glass one-way mirror	**Mirror film applied to glass**	**Reflective acrylic**
Cost - 0.25	0.5	1	0.25
Reflective Effectiveness - 0.25	0.6	0.4	0.5
Effort of Installation -0.25	1	0.25	0.75
Availability - 0.25	0.5	1	0.25
Total	0.65	0.6625	0.4375
Continue	No	yes	no

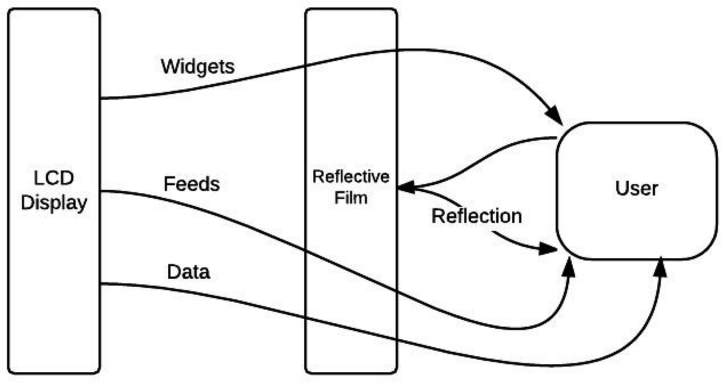


Figure 9 – "LCD Display to Reflective Film"

CHAPTER 3

3.1 USER INTERFACE AND CONTROLS

The UI is aesthetically appealing, it designated such that a good portion of the interface to show the reflection of the user. We designed the UI using Bootstrap, a popular framework for designing the front end of an application. The large range of templates allowed for a solid base for the layout which was later customized to suit the needs of the system.

When the user initially opens the webpage on the mobile component, the user interface displays the homepage with fields to enter an existing username and password, a "Login" button to login to an existing account with the provided username and password, a "New User" button to create a new account, and a "Forgot Password?" button to retrieve a forgotten password. See Figure 10 for the homepage mockup.

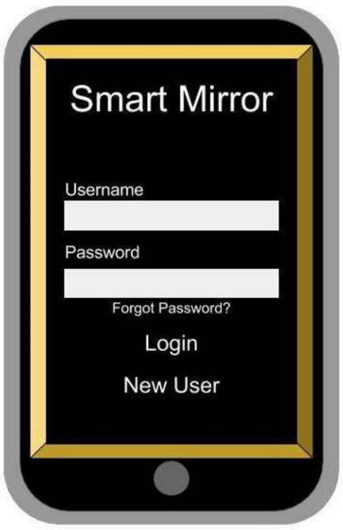
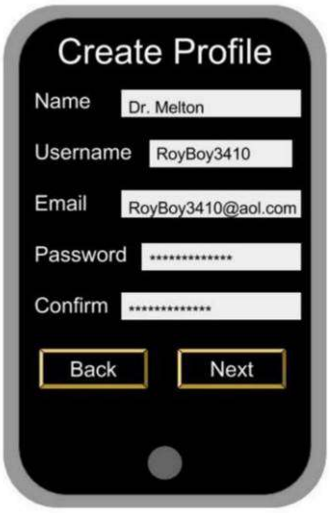


Figure: Mobile Component's UI: Homepage

If the person opted to create a new account through clicking on the "New User" button, they displayed the Create User web page with fields for important statistics to be entered. The required fields are call, Username, Email, Password, and Confirm Password. Once all fields had been well populated, the "Next" button might be pressed. See Figure 11 for the Create User web page mockup



After the user presses Next in the Create Profile page, a page with a message to go take a selfie from the Smart Mirror's camera will be displayed. This step provides a photo associated with the specific user for facial recognition. The user will press the Take a Selfie button, and the key component will take a photo of the user. Once the user is satisfied with the photo taken, the user would press the Next button. See Figure 12 for the mockup of the Take Selfie page.

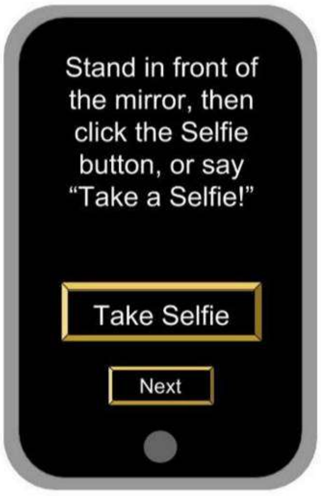
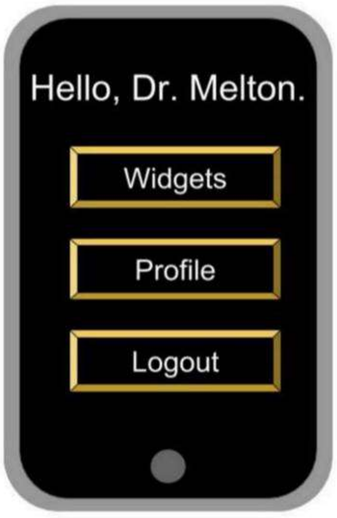

Stand in front of the mirror, then click the Selfie button, or say "Take a Selfie!"
Take Selfie
Next

Figure: Mobile Component's UI: Take Selfie Page

Once a user with an existing account logs in after entering his or her username and password and pressing the "Login" button on the Homepage, or once the user completes the photo aspect of the sign-up process, a home screen for a logged in user will display three buttons: "Widgets," "Profile," and "Logout." The "Logout" button will return the user to the home page. See Figure for the mockup of the UI as soon as the user successfully logs in.



If the user presses the "Widgets" button, the display will then show a list of all the widgets available for use. A Back button can be pressed if the user chooses not to change his or her widgets. See Figure for the mockup of the display for the list of widgets.

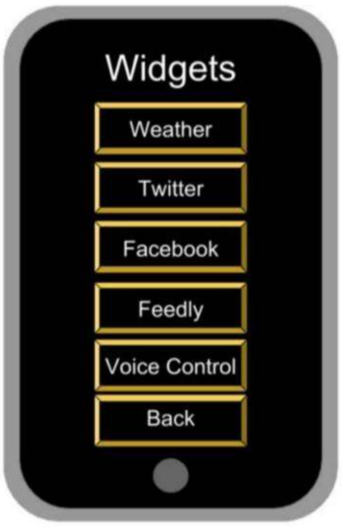
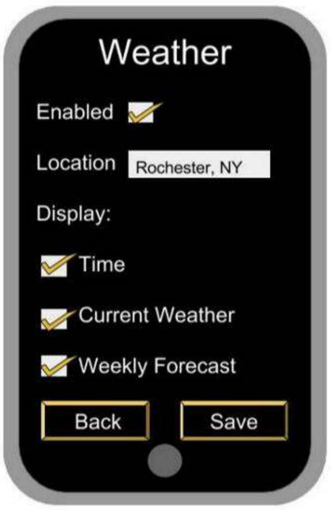


Figure - Mobile Component's UI: List of Widgets Page

Choosing one of the widget buttons will bring up the preferences of the widget's display, which may vary between the different widgets. Using the Weather Widget as an example, preferences may include whether it enables the widget to be displayed on the Smart Mirror and the location for the desired weather information. Additional preferences include whether the current time, current weather, and the weekly weather are displayed. See Figure 15 for the Preference page mockup of the Weather Widget.



Weather

Enabled ✓

Location: Rochester, NY

Display:

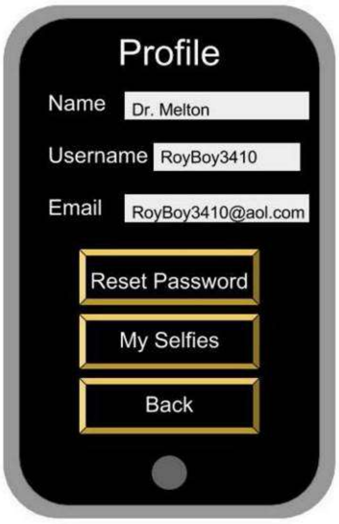
✓ Time

✓ Current Weather

✓ Weekly Forecast

Figure - Mobile Component's UI: Preference Page for Weather Widget

If the user presses the "Profile" button from the Logged-In Home page, the Profile page will be displayed, which contains the user's name, username, and email address , and the "Reset Password," "My Selfies," and "Back" buttons. See Figure 16 for the Profile page mockup.



Profile

Name: Dr. Melton

Username: RoyBoy3410

Email: RoyBoy3410@aol.com

Figure - Mobile Component's UI: Profile Page

Pressing the My Selfies button from the Profile page will display My Selfies page, which contains pictures taken from the Smart Mirror's camera. If they select a photo, the Delete Selfie and Email Selfie options will be available to be pressed.

.

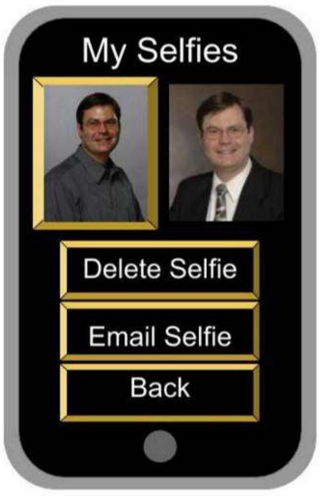


Figure - Mobile Component's UI: My Selfies Page

Last, if an existing user forgets his or her password, they may press the "Forgot Password?" button from the mobile component's Homepage to get a link to reset the password sent to them via email.

After signing up for an account using the mobile component, the user must log in to use the Smart Mirror through facial recognition, which is done by facing the mirror. Once the user has been authenticated, the web application opens, allowing the user to control the interface through hand gestures and voice commands.

When the user logs into the mirror for the first time, the layout of the user interface will display the default settings, which include weather information and the current date and time. It will include no widgets for social media sites or RSS feeds. I can change the display settings through the mobile interface or through the key component using gesture and voice commands.

For the UI control, hand gestures and voice commands were used. Voice commands were used to trigger events, such as taking a selfie or searching the web. Hand gestures can interact with events, such as selecting yes or no when prompted by an event, and supporting voice commands to handle prompts. Hand gestures can also trigger events, the same way as voice commands, although gesture support lacks some features, such as searching the web and adding comments to posts. I geared the overall UI control more towards voice commands and less towards gesture support.

3.2 ENGINEERING STANDARDS

I made most API calls with JavaScript and Python. All JavaScript code adhered to the Google JavaScript Style Guide as tons as feasible, and all Python code adhered to PEP 8 standards, when possible. They maintained all code the usage of Git version control. JSON, a information-interchange layout, changed into used extensively for sending messages among the mirror programs and APIs.

The optionally available Wi-Fi connection used the Raspberry Pi Wi-Fi Adapter. It complies with the IEEE 802.11n (Draft 2.0), IEEE 802.11g, and IEEE 802.11b requirements. The Odroid XU4 has supported drivers for the Wi-Fi adapted; therefore, no extra steps had been had to provide Wi-Fi connection. The reflect can also connect with the net the use of Ethernet/IP protocols.

The show output preferred is a 1920x1080 resolution outputted thru HDMI. I based totally all user interface visible layout in line with those standards. The cell component interaction is an extension to the Django net-application through a web browser, so the mobile interface became designed based on cell cellphone decision constraints.

Multidisciplinary Aspects

They require fundamental electronics knowledge to get all of the components linked to one another; there's no low-stage circuitry layout involved within the product's creation. Very basic mechanical engineering know-how is required to layout the body of the device that homes the digicam, display, and embedded laptop at the same time as retaining it aesthetically appealing. It calls for the development methods of Software Engineering and Computer Science to create the software components. Computer Engineering makes use of the combination of hardware implementation with software development to create a whole operating gadget.

Prior Background

All the members of this crew have had prior enjoy each growing on Linux and the usage of embedded computers because of private initiatives and Computer Science courses. Everyone inside the layout group has taken at least one route in software engineering. The development, trying out, validation, and assessment of the code accompanied by using design practices mentioned in these training. The team however did not have very an awful lot revel in interfacing with a Kinect or working with Web APIs, so we spent a massive quantity of time researching and gaining revel in in the subjects. The Codecademy class "How to apply APIs with JavaScript" provided extra guidance to assist all and sundry on the development team strange with JavaScript or the use of APIs. A few of the members have had revel in with CAD software. I designed the body of the Smart Mirror in CAD.

Outside Contributors

I consulted handiest two other human beings out of doors of the group with components of this venture. Devin Sherman assisted within the CAD layout of the clever mirror frame, and Dylan Zingler helped with various technical questions involving the Django framework.

3.3 CONSTRAINT & CONSIDERATION

Extensibility

APIs

I designed every widgetmodular and characteristic independently of other widgets or other Smart Mirror components. This layout allows the code from any widget to be utilized by builders in other websites or web-primarily based programs if they have an API key, wherein relevant. Additional APIs may be carried out later, or they will enforce additional capability with the chosen APIs.

Manufacturable

APIs

If the Smart Mirror became a commercially to be had product, use of the APIs may also no longer be unfastened because of the extended frequency of API calls. Before commercialization, each API company ought to be contacted and we should discuss all utilization of the API to prevent any issues after production. Many large agencies like Facebook and Twitter require special agreements for commercial products. During the product development, the Facebook and Feedly APIs modified, which led to neither API getting used in the closing product. Instead, SoundCloud and FeedParser were used. If the product becomes commercialized, they might combine Facebook. I might require similarly improvement to put in force it, though.

Reflective Surface

The reflective movie surface introduces greater exertions costs than different reflective substances. If it commercialized the Smart Mirrorcommercialized, the hard work procedure for putting in the film could boom the producing time and fee.

Health and Safety Issues

Embedded Computer

The handiest highbrow belongings hurdles seem in the event that they commercialize the tool. The highbrow belongings problems related to the embedded computer could no longer be a situation so long as the remaining product's pricing accounted for the cost of the embedded computer.

Camera

Because of the area of a typical toilet replicate, placement and operation time of the digital camera has to be stored in consideration. I will join the device to the net, meaning it might be compromised. If an attacker had get admission to to the tool remotely, they could use the digicam without the owner's expertise. Having some bodily blocker at the cameras or a kill switch could save you unwanted usage of the digicam. Restricting access to the digicam and saving images only when the person grants permission may also be preventative measures. Ideally, the device is best as prone as every other computer on the network. If malicious attackers received access to the network, they might have loose rein over anything on the community. Encryption ought to apply to all of the stored snap shots, however like previously said, if the hacker had get entry to to the network or even the reflect itself, decrypting all of the files stored could be trivial.

Intellectual Property

<u>Facebook</u>

I accomplished tOS investigation on Facebook. However, because of design changes, a Facebook widget turned into by no means implemented so the worries aren't relevant. Facebook has a totally specific policy guideline for its API. For the Smart Mirror layout, the subsequent hints need to be adhered to: user permissions must be requested for all facts accumulated, any statistics amassed ought to be secured, all content published have to be “high fine” and keep away from junk mail, and any API calls have to not create any mistakes along with consistent failed logins. Facebook forbids any replication of its center functionality. However, the Smart Mirror application ought to not infringe on any of this center capability.

Twitter

The Twitter APIs allow both the posting of content, inclusive of snap shots, and the show of a consumer’s feed. Twitter calls for that explicit consumer consent be got earlier than every object is published or modified by means of the Smart Mirror. Content published have to now not incorporate junk mail or records hidden from the person. Any “tweets” displayed ought to conform to the Twitter show guidelines and can not be processed and made to be had thru other display techniques. Twitter prevents the replication of the “middle person revel in”; but, the Smart Mirror layout would now not infringe on the conventional workflow that most users take by means of Twitter’s definition.

Instagram

Instagram prevents any duplication of center capability from being applied using the API. Specifically, photographs cannot be posted outside of the Instagram app. The app lets in pictures to be pushed from a 3rd-birthday celebration utility to the Instagram app, but the workflow for the user is messy and discouraged via Instagram. If content is pushed to Instagram from the Smart Mirror, it ought to avoid any spamming and use reasonable ability. The Instagram API additionally strictly limits how many photos from Instagram can be displayed using the API and boundaries the ability to cache pics. Similar to the scenario with the Facebook widget, an Instagram widget turned into now not carried out. Therefore, these worries are not applicable.

Feedly

The Feedly TOS are not very restrictive and awareness on freedom and privacy. The Smart Mirror application have to no longer junk mail the person, Feedly, or any 0.33—birthday celebration web page. This approach that RSS feed need to be introduced to the show only if the consumer requests it, no extra statistics should be injected into the feed, and the Smart

Mirror must limit the API calls to avoid spamming the servers. During the development procedure, a change inside the Feedly API turned into introduced, and development keys have been not available. Therefore, the Feedly API may want to now not be included into the Smart Mirror, and sufficient RSS Feed aggregated changed into investigated.

Open Weather Map

OpenWeatherMap is covered below the innovative common public license CC BY—SA. It is essential to notice for the Smart Mirror that seen attribution should take delivery of to OpenWeatherMap in the display, and that any implementations made to the OpenWeatherMap API should be blanketed under the same license. This might also make it important to launch the precise implementation of the climate widget for the mirror under a exceptional license than the relaxation of the code base.

CHAPTER 4

4.1 TESTING STRATEGY

Embedded Computer

We designed the embedded computer testsdetermined the overall overall performance of the tool within the gadget. There are few assessments that run to make sure that we obtain the optimum capability. Kinect capability tested by means of putting in freelibnect2, an open-supply driver for the Kinect, and walking the supplied check applications. There are open-source GPU and CPU benchmarks that have been run on Ubuntu that gave statistics to compare. Installing the ones benchmarks at the special embedded computers and evaluating the numerical values turned into the easiest and most trustworthy way to verify that the embedded computer changed into jogging near the producer-specified costs.

Determining that the device met the advertising requirements became as simple as developing the device and testing normal functionality. Timing the facial detection pace changed into honest in conjunction with how nicely the gesture manage labored. The only flaw with figuring out how the movement controls is the reality that "responsive" motion controls are depending on the tester's opinion. The most effective way to test long-term usage was to create the device and study how nicely it plays over a protracted span of time, however due to the short time span of this mission, this is not feasible. However, in the course of the mission, no troubles discovered with always going for walks the device. We can see the embedded pc assessments inside the Appendix below Camera/Embedded Computer Tests.

Camera

To verify that the digicam subcomponent was operating efficaciously with the rest of the general device, the digital camera connected to the embedded computer to validate that the camera produced pictures of decision 720 x 480 at 40 fps from the RGB and intensity cameras and sent the frames over USB to the embedded pc with out a body loss. After verifying that there has been no frame loss between the embedded pc and the camera, we validated it that the embedded pc may want to process the snap shots to perform facial popularity and gesture recognition. Verifying these tests proved that the embedded computer has enough USB throughput to address receiving the frames appropriately, andas having enough processing energy to interpret the photos to perform the responsibilities had to meet the overall necessities. We can see the digicam tests in the Appendix underneath Camera/Embedded Computer Tests.

Web Application

I did unit checking out the functions and components of the net software to make certain accurate functionality. I shall check the server side on typical UI navigation, even as we tested the mobile client considering the interactions with the server side. This consists of registering for a user account and uploading snap shots for the facial recognition software. They also tested the verbal exchange between the server and consumer became additionally. I analyzed the database to ensure that information, inclusive of consumer login credentials and the associated account alternatives, written and pulled nicely. We can see the web application checks in the Appendix underneath Web Application Tests.

APIs

They finished API testing for every API in priority order. Testing turned into first finished within the API console furnished by using the 1/3-birthday celebration employer. All API calls vital for the widget tested in the console. Next, it made all API calls from a simple JavaScript software on a computer with an internet connection. Once all calls have been confirmed, the JavaScript application runs from the embedded computer and all of the outputs had been then validated. They then integrated the API calls had been then into widgets that were first examined on the pc, and then the embedded pc. I continually completed development changed into constantly at the laptop first due to the fact more strong improvement equipment are to be had, and it left the embedded computer available for other trying out. We can see the API tests within the Appendix underneath API Tests.

Reflective Surface

A pattern of reflective film were given and set up on a small pane of acrylic. The same pane should have been positioned in front of the LCD display to validate that finished as a one-way mirror, while permitting photos on the display screen to pass thru. We ought to additionally area the pane should also in a excessive humidity surroundings to validate that the film will hold to adhere with the presence of moisture; however, this check did not completed. Once we had finished the checking out, they then established the reflective movie become then onto an acrylic pane sized for the device. We can see the reflective floor checks within the Appendix underneath Reflective Surface Tests.

4.2 RISKS

Embedded Computer

The wide variety of dangers involving an embedded laptop is quite low. Long-term stress testing of the device done. The XU4 confirmed no degradation of performance, even after being one for over a week at a time. As a precautionary measure, a day by day Cron process to restart they carried out the XU4. High-humidity testing no longer accomplished. Testing might positioned the hardware at chance for harm and determined to be out of doors this improvement cycle.

Long Term Usage: Before acquiring the embedded pc, not much trying out had performed on lengthy-term usage of the XU4 because of how new the tool is. The cooling machine for the CPU utilizes a fan, and the processor generates extra warmness. Long time period computing stress monitored as soon as a strong working prototype created, and we located no troubles.

Environment Resistance: The improvement and prototyping surroundings of the device differed considerably from real person usage. Because it is a reflect, it'd show up in toilets. Depending at the replicate’s placement within the room and how nicely the body waterproofed, the embedded pc may want to have uncovered to excessive ranges of humidity each day. Waterproofing now not examined due to the excessive danger of damaging the diverse additives.

Camera

The embedded laptop used in the task design at once associated with the digital camera element and confined to which digital camera become decided on. Motion detection cameras, just like the Microsoft Kinect, require about 50% of the throughput on USB 2.Zero so it is critical that an embedded laptop selected to satisfy this requirement, and feature sufficient processing power to interpret the snap shots and locate gestures.

The Microsoft Kinect taken into consideration and chosen to the camera. This selection was primarily based on the open-supply availability of help with examples, andas the provision of the Kinect (already proudly owning one). The Kinect first of all paired with the Raspberry Pi. This resulted in a few troubles with the USB throughput between the Kinect and the embedded laptop. We determined it through hardware boundaries, in which the USB bus connected to the equal bus because the Ethernet and SD card, which reasons a bottleneck at the USB ports. After understanding this bottleneck, we decided on a new embedded pc that is more powerful and has better USB throughputs, the Odroid-XU4.

Design of the use of the Microsoft Kinect minimizes risk as compared to other devices which includes the Asus Xtion, another 3-D digicam. Kinect offers open-source libraries and a huge variety of network help to allow software program to be developing to meet the necessities for the assignment.

Microsoft Kinect tested and proven to paintings effectively the usage of a sixty four-bit x86 architecture Desktop. Upon seeking to reflect the functionality via connecting the Kinect to a Raspberry Pi, we located it that the USB throughput of the embedded laptop become no longer enough and therefore observed body loss. The primary concern of the overall undertaking is to pick out an embedded laptop that has enough USB throughput and processing electricity to receive and interpret the statistics from the digicam.

Upon deciding on the Odroid-XU4, the USB for the duration of situation eliminated, and we determined no frame loss. However, processing power and USB all through constantly examined as a challenge that there may grow to be a bottleneck when greater sources I need.

Humidity could be an trouble for the digicam if there was no casing to defend it. For now, I would advise it that the smart replicate now not area in an area with high humidity, including a bathroom. A means to defend the digicam and they'd take into account all of the hardware for the smart replicate for later releases.

Web Application

There are few risks concerning the development of the Smart Mirror's web application. Although the development group is gifted with Python programming, getting familiar with the tools provided by means of the Django framework took some time. In addition, the chosen embedded computer to host the net application wished high processing power to method the net web page requests made by the person.

APIs

It turned into hard to put in force all desired of the API functionality within this challenge. If we evolved many widgets in parallel, there was a hazard that none will completely combine into the Smart Mirror inside the timeframe of the challenge. To mitigate this risk, they did development in short cycles, implementing every favored widget completely earlier than beginning development on the next. This ensured that capability advanced inside the priority order and effectively integrated into the Smart Mirror.

Relying on support from 0.33-celebration applications creates the chance that the 0.33 party will exchange or end aid for middle functionality used inside the mirror. It is impossible to take away this threat except there aren't any outside APIs used. This is an unreasonable answer because social media interplay is a middle layout requirement. To mitigate this danger, excessive priority capability such as sharing photos need to implemented the use of a couple of APIs and social media networks so it is less likely that each one photograph—sharing capability lost without delay, however, on this undertaking simplest one API used for image-sharing.

Reflective Surface

We do no longer intend the reflective movie to be used as a Smart Mirror surface. If the show is brightly lit, the widgets will be display, more however the effectiveness of the replicate will lower. After I established the reflective surface, it calibrated the brightness of the LCD show to optimize the readability of the widgets versus the effectiveness of the surface as a replicate. There was also some threat that air bubbles might shape underneath the reflective film while it is carried out to the acrylic pane. Air bubbles might create a less professional appearance to the surface. Extra reflective film bought, and it implemented the film several times till the movie seemed bubble unfastened.

REFERENCES

1) Smart mirror with focus control by Ken-Hyung ParkTae-Seon KIMChang-Ryong HeoMin-Young KimTae-Kyun Kim.

2) Smart mirror for ambient home environment by MA Hossain, PK Atrey, A El Saddik.

3) Augmented Rendering of Makeup Features in a Smart Interactive Mirror System for Decision Support in Cosmetic Products Selection by A. S. M. Mahfujur Rahman ; Thomas T. Tran ; Sk Alamgir Hossain ; Abdulmotaleb El Saddik.

4) Smart Makeup Mirror: Computer-Augmented Mirror to Aid Makeup Application by Eriko Iwabuchi, Maki Nakagawa, Itiro Siio.

5) Smart mirror: A novel framework for interactive display by S Athira ; Frangly Francis ; Radwin Raphel ; N S Sachin ; Snophy Porinchu ; Seenia Francis.

6) Individualized Deliberate Practice on a Virtual Reality Simulator Improves Technical Performance of Surgical Novices in the Operating Room: A Randomized Controlled Trial by Palter, Vanessa N. MD, PhD*; Grantcharov, Teodor P. MD, PhD.

7) Skills acquired on virtual reality laparoscopic simulators transfer into the operating room in a blinded, randomised, controlled trial by PH Cosman, TJ Hugh, CJ Shearer, ND Merrett.

8) Fundamentals of Laparoscopic Surgery simulator training to proficiency improves laparoscopic performance in the operating room—a randomized controlled trial by Author links open overlay panel Gideon Sroka M.D.Liane S.FeldmanM.D.Melina C.Vassiliou M.D.Pepa A.Kaneva M.Sc.RaadFayez M.D.Gerald M.FriedM.D.

APPENDIX

The codes used in our project Smart Mirror.

1.1 Code for Smart Mirror UI/UX

```
“from Tkinter import *
 import locale
 import threading
 import time
 import requests
 import json
 import. traceback
 import. feedparser



   from PIL import Image, ImageTk
 from’ contextlib import contextmanager’



 LOCALE_LOCK == threading.Lock()



" ui_locale = " # e.g. 'fr_FR' fro French, " as default”
" time_format = 12 # 12 or 24”

" @contextmanager”
" def setlocale(name): #thread proof function to work with locale’
     ‘with LOCALE_LOCK:”
           “yield lo’cale.setlocale(locale.LC_ALL, name)’
        finall’y:
```

```
        lo’cale.setlocal’e(loc’ale.LC_ALL, saved)


“icon_lookup = {”
  'c’lear-da’y': "ass.ets/Sun.png",  # clea.r sk.y day
   ‘wi.nd': "assets/W.ind.png",   #wi.nd
   'clo.u.dy': "a.sse.ts/Cl.oud.png",  # cloud.y day
   'part.ly-clo.udy-day': "assets/Pa.rtlySun.ny..png",  # par.tly cl.oudy da.y
   ‘ra.in': "ass.ets./Rain.png",  # rain da.y
   ‘sn.ow': "as.sets/Snow..png",  # s.now day
    'thun.derstorm': "assets/.Storm.png",  # thun.derstorm
   'torna.do': "assests/Tor.nado.png",  .  # tornad.o
   'hail': "a.ssests/Hail.png"  # h.ail”
}


cl.ass Cl.ock(Frame):
   de.f __ini.t__(s.elf, par.ent, *args, **kwargs):
      Fram.e.__init__(self, pa.rent, bg='black')
      # init.ialize time label
      self.ti.me1 = ''
      self.ti.meLbl = Label(self, .font=('Helve.tica', larg.e_text_size), fg="white".,
bg="black")
      self.time.Lbl.pack(.side=T.OP, anchor=E).
      # initializ.e day o.f week
      self.day_of_w.eek1 = ''
      self.dayO.WLbl =. Label(se.lf, text=s.elf.day_o.f_week1, .font=.('Helvetica',
.small_text_size), fg=."white", bg="black")
      self.day.OWLbl..pack(side=TOP, anchor.=E)
```

```
        # initializ.e dat.e label
        self.date.1 = ''
        self.date.Lbl = Label(self, text=self.dat.e1, font=('Helvet.ica', small_text_size),
fg="white"., bg="black")
        self.dat.eLbl.pack(side=TOP, anchor.=E)
        self.tick(.)




   class. Weather(Frame):
     def. __init__(self, parent, *args, **kwargs):
        F.rame.__init__(self, parent, bg='black')
        se.lf.temperature = ''
        sel.f.forecast = ''
        self..location = ''
        self..currently = ''
        self.i.con = ''
        self.d.egreeFrm = Frame(self, bg="black")
        self.d.egreeFrm.pack(side=TOP, anchor=W)
        self.te.mperatureLbl = Label(self.degre.eFrm, font=('Hel.vetica', xla.rge_text_size),
fg.="white", bg="black")
        self.te.mperatureLbl.pack(si.de=LEFT, anchor=.N)
        self.ico.nLbl = Label(self.deg.reeFrm, bg="black")
        self.ico.nLbl.pack(side.=LEFT, anchor=N, padx=20)
        self.cur.rentlyLbl = La.bel(self, font=('Helvetica', medium_text_size), fg="white",
bg="black")
        self.currentlyLbl.pack(.side=TOP, anchor=W)
        self.forecastLbl = Labe.l(self, font=('Helvetica', small_text_size), fg="white",
bg="black")
        self.forecastLbl.pack(si.de=TOP, anchor=W)
        self.locationLbl = Labe.l(self, font=('Helvetica', small_text_size), fg="white",
bg="black")
```

```
        self.locationLbl.pack(si.de=TOP, anchor=W)
        self.get_.weather().
  .
 .
    def. get_ip(self):
        try:
          .  ip_url = "http://jsonip.com/"
           . re.q. = requests.get(ip_u.rl)
            .ip_json = json.loads(req.t.ext)
            r.eturn ip_json['ip'].
        except Excepti.on as e.:
            traceback.print_exc.()
            return "Error.: %s. C.annot get ip." % e



                location.2 = "%s, %s" % (locati.on_obj['city'], .loc.ation_obj['region_code'])



                # ge.t weather
                weather._req_url =
"htt.ps://api.darksky.net/forecast/%s/%s,%s?lang=%s&units=%s" % (weather_api_token,
lat,lon,weather_lan.g,weather_unit.)
            else:.
                lo..cation2 = ""
                # get weather
                we.ather_req_url =
"https://api.darksky.net/forecast/%s/%s,%s?lang=%s&units=%s" % (weather_api_token,
la.titude, longi.tude, weather_lan.g, weather_unit)
  .
 .
            r = requests.get(weathe.r_req_url)
            weather_obj = json.load.s(r.text)
            degree_sign= u'\N{DEGR.EE SIGN}'
            temperature.2 = "%s%s" .% (str(int(weather_obj['currently']['temperature'])),
degree_sign)
  .          currently.2 = weather_obj['currently']['summary']
            forecast.2 = weather_obj[“h.ourly"]["summary"]
 .

            icon_.id = weather_obj[.'currently']['icon']
            icon2. = None
 .

            if icon_.id in icon_lookup:
               icon2 = icon_lookup[..icon_id]



  class Fullscreen_Window:


    def __init_._(self):
         self.tk =. Tk()
       self.tk.co.nfigure(background='black')
       self.topFr.ame = Frame(self.tk, background = 'black')
       self.botto.mFrame = Frame(self.tk, background = 'black')
       self.topFr.ame.pack(side = TOP, fill=BOTH, expand = YES)
       self.botto.mFrame.pack(side = BOTTOM, fill=BOTH, expand = YES)
       self.state .= False
       “self.tk.bin.d("<Return>", self.toggle_fullscreen)
       self.tk.bin.d("<Escape>", self.end_fullscreen)
         # clock.
       self.cloc.k = Clock(self.topFrame)
       self.cloc.k.pack(side=RIGHT, anchor=N, padx=100, pady=60)
       # weath.er
       self.weat.her = Weather(self.topFrame)
       self.weat.her.pack(side=LEFT, anchor=N, padx=100, pady=60)
       # news.
       self.new.s = News(self.bottomFrame)
       self.new.s.pack(side=LEFT, anchor=S, padx=100, pady=60)
       # calend.er - removing for now
       # self.cal.ender = Calendar(self.bottomFrame)
       # self.cal.ender.pack(side = RIGHT, anchor=S, padx=100, pady=60)
```

1.2 Virtual Trial Room

```
import os
 import gtk
 import gtk.gdk as gdk
 import gobject
 import cv2
 import numpy as np
 import time



 from Video import Video
 from Back_sub import RemoveBackground
 from color_replace import Replace
 from normalized import NormalizedRGB
 from Tshirt import DetectShirt



 from config import red
 from config import green,blue
 from config import yellow,pink,brown,plain,temp1,temp2,color,design



 templates="templates/"
 assets = os.path.join(".", "assets")
 tests = os.path.join(".", "tests")
 TEST_MODE = bool(int(os.environ.get('VDR_TEST', 0)))
```

```
class MainUI:

   def __init__(self):

      self.isBanner_mode=True
      self.final=None
      self.design=7
      self.color=None

      #Initialize systems objects
      if TEST_MODE:
         self.vid=cv2.VideoCapture(os.path.join(tests, "demo.avi"))
      self.v=Video()
#----------------testing purpose--------------#

      self.back=RemoveBackground()
      self.norm=NormalizedRGB()

      self.replace=Replace()
      #print self.replace
      self.tshirt=DetectShirt()



      self.p_mat=np.array(np.mat([[0,0],[100,0],[100,100],[0,100]],np.float32))
      self.design_template=cv2.imread(templates+"Green.png")
      #---------------------------------------------------

      self.__glade__=gtk.Builder()
      self.__ui__=self.__glade__.add_from_file(os.path.join(assets, "main_ui.glade"))
self.__main_win__=self.__glade__.get_object("vdr_main")
self.drawing_area=self.__glade__.get_object("da1")

self.about=self.__glade__.get_object("abt")
self.about.connect("clicked",self.show_abt)
self.about_dia=self.__glade__.get_object("about_dia")

#init Drawing Area
self.drawing_area.realize()
self.drawing_area.set_size_request(config.width, config.height)
#get canvas
self.canvas=self.drawing_area.window

#get GC
self.gc=self.canvas.new_gc()
#draw rectnagle
#print self.canvas,self.gc

self.gc.set_background(gdk.Color(0,0,0,0))
self.gc.set_foreground(gdk.Color(255,0,0,0))

#print dir(self.canvas)
self.canvas.draw_rectangle(self.gc,False,10,10,100,200)
self.canvas.draw_line(self.gc,10,10,100,100)

self.tview1=self.__glade__.get_object("tv1")
self.lstore1=self.__glade__.get_object("ls1")
self.tview2=self.__glade__.get_object("tv2")
self.lstore2=self.__glade__.get_object("ls2")
self.apply_btn=self.__glade__.get_object("apply")
self.apply_btn.connect("clicked",self.change_avtar)

" self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Red.png'),red])
self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Green.png'),green])
self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Blue.png'),blue])
self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Yellow.png'),yellow])
self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Pink.png'),pink])
self.lstore1.append([gtk.gdk.pixbuf_new_from_file(templates+'Brown.png'),brown])"

#treeview = gtk.TreeView(self.lstore1)
self.tview1.set_model(self.lstore1)
self.tview2.set_model(self.lstore2)

self.tview1.modify_base(gtk.STATE_NORMAL,gtk.gdk.Color(0,0,0,0))
self.tview2.modify_base(gtk.STATE_NORMAL,gtk.gdk.Color(0,0,0,0))



cell = gtk.CellRendererPixbuf()
column = gtk.TreeViewColumn("Pixbuf", cell)
column.add_attribute(cell, "pixbuf", 0)
self.tview1.append_column(column)
cell.set_padding(30,50)

cell2=gtk.CellRendererPixbuf()
column2 = gtk.TreeViewColumn("Pixbuf", cell2)
column2.add_attribute(cell2, "pixbuf", 0)
self.tview2.append_column(column2)
cell2.set_padding(30,50)
```

```
    #self.canvas.draw_
    self.drawing_area.set_app_paintable(True)
    self.drawing_area.connect("expose-event",self.display_frame)

    selection_color=self.tview1.get_selection()
    selection_color.connect("changed",self.change_cloth_plain)

    selection_design=self.tview2.get_selection()
    selection_design.connect("changed",self.change_cloth_design)

    self.canvas.set_background(gtk.gdk.Color(0,0,0,0))

    self.__main_win__.connect("delete-event",gtk.mainquit)
    self.img=cv2.cv.LoadImage(os.path.join(assets, "Banner.png"))
    #

    self.__main_win__.show()
    self.__main_win__.maximize()

def show_abt(self,event):
    self.about_dia.run()
    self.about_dia.hide()



def change_avtar(self,event):
    print 'ok'
    if self.isBanner_mode:
        self.img=cv2.cv.LoadImage(os.path.join(assets, "Banner_2.png"))
```

```
            self.timer_id=gtk.timeout_add(20*1000,self.reset)
            self.isBanner_mode=False
            self.drawing_area.queue_draw()

    def reset(self):
        self.img=cv2.cv.LoadImage(templates+"Banner.png")

        self.isBanner_mode=True
        self.drawing_area.queue_draw()
        gtk.timeout_remove(self.timer_id)



    def display_frame(self,a,b):

self.drawing_area.window.draw_rectangle(self.drawing_area.get_style().white_gc,False,0,0,7
99,599)

        if self.isBanner_mode:

self.canvas.draw_rgb_image(self.gc,1,1,798,598,gtk.gdk.RGB_DITHER_NORMAL,self.img
.tostring(),2400)
            #self.final=self.getOutput_frames()
            #self.final=cv2.cvtColor(self.final,cv2.cv.CV_BGR2RGB)

        elif not self.isBanner_mode:

            if self.final==None:
                self.final=self.getOutput_frames()
                self.final=cv2.cvtColor(self.final,cv2.cv.CV_BGR2RGB)
                self.drawing_area.queue_draw()
                return
            else:
```

```
self.canvas.draw_rgb_image(self.gc,1,1,798,598,gtk.gdk.RGB_DITHER_NORMAL,self.fina
l.tostring(),2400)
            self.final=self.getOutput_frames()
            self.final=cv2.cvtColor(self.final,cv2.cv.CV_BGR2RGB)
            self.drawing_area.queue_draw()


#self.canvas.draw_rgb_image(self.gc,1,1,798,598,gtk.gdk.RGB_DITHER_NORMAL,self.fin
al.tostring(),2400)



    def getOutput_frames(self):
        # using VideoCapture for test


        if TEST_MODE:
            _,frame=self.vid.read() #NOTE: Testing without camera. uncomment this to feed
from camera.
        else:
            frame=self.v.outFrame() #NOTE: feeding from Camera.



        self.norm.getRGB(frame) #input to Normalized RGB


        norm_rgb=self.norm.normalized() #normalized RGB
        print 'Got normalized RGB '
        rgb_planes=self.v.imagePlanes(norm_rgb)


#-------bg subtraction part-----------
#       load background samples
        self.back.loadBackground()
        self.back.getFrames(rgb_planes[1])
        self.back.subtract_back(norm_rgb)

        subtracted=self.back.remove(frame)
        print 'background subtracted now'

        self.tshirt.getFrames(norm_rgb)
        mask,cntr=self.tshirt.detect_shirt()
        print 'found tshirt'

        self.replace.getFrames(subtracted,mask)

        res=self.replace.replace_color(self.color)

        if self.design is not 7:
            res=self.replace.replace_design(cntr,self.p_mat, self.design_template, res)

        #replace.replace_design(cntr, p_mat,design_template,res)



        #cv2.imshow("subtracted", res)

        return res



    def change_cloth_plain(self,selected):
        global color
        model,iter=selected.get_selected()
        color=model.get_value(iter,1)
        self.color=color
        def change_cloth_design(self,selected):
            global design
            model,iter=selected.get_selected()
            self.design=model.get_value(iter,1)


            if self.design==8:

    self.design_template=cv2.imread(templates+"nike.png",cv2.cv.CV_LOAD_IMAGE_UNCH
    ANGED)
            if self.design==9:

    self.design_template=cv2.imread(templates+"Rbk.png",cv2.cv.CV_LOAD_IMAGE_UNCH
    ANGED)




    def sample_bg(frame):
        '''This function sample some background images for bakground subtraction.'''
        if frame==None:
            print 'Training skipped...NOTE: VDR will use already sampled background.'
        else:
            cv2.imwrite("./train/back_g.jpg",frame)



    def Main():


        train=False
        print 'training started'


        if train:
            v=Video()
        n=NormalizedRGB()
        for i in range(100):
            frm=v.outFrame()
            n.getRGB(frm)
            norm=n.normalized()
            plain=v.imagePlanes(norm)

            sample_bg(plain[1])
        print 'training ends...'
        del v


    m=MainUI()
```