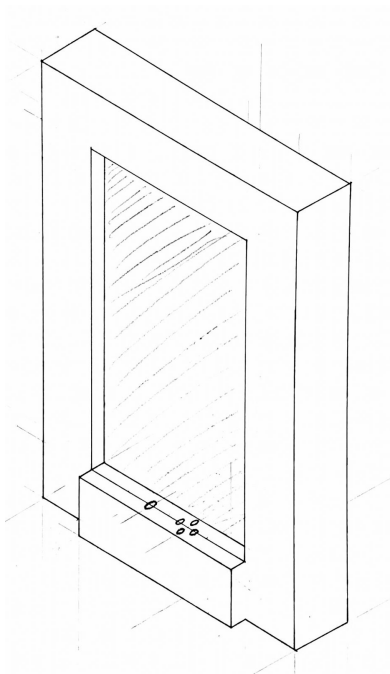


Department of Electrical Engineering and Computer Science
EECS 347 – Microprocessor System Projects
Winter 2017



Personalizable Display Frame ver 1.0
Design Document rev. 1.0

Names: Alex Gangwish, James Kim,
Karan Shah, Chris Sims
Submitted to: Professor Henschen
Report Due: 3/10/2017

Table of Contents

Table of Contents	1
Table of Figures	3
Executive Summary	4
Design Overview	5
Hardware Design	5
Exterior Components	6
Frame	6
Button Interface	7
Power Supply	8
Electronic Components	8
Microcontroller (BCM2837)	8
Liquid Crystal Display (LCD)	9
Expandable Memory	9
WiFi Card	10
Software Design	10
Embedded Software	10
Data Retrieval	10
Image Display	11
Physical Interface Control	11
Application Software	11
User Application	12
Management Application	16
Database Software	16
Suggested System Structure	17
Profile Management	17
Frame ID Management	19
Linking	19
Future Considerations	20
Additional Features	20
Music	20

Videos	20
RFID	20
Caller ID	21
Flight /Local Information	21
Adjustable Lighting	21
Alternative Hardware	21
WiFi Chip	21
External Memory	22
Microcontroller	22
Cost of the Project	22
References	22
Appendix: Completion Schedule	23
Appendix: Use Cases and Scenarios	24
Use Cases	24
Create a Profile	24
Upload Pictures	24
Upload Schedule	25
Change Displayed Information	26
Update Information	27
Discerning Between Users	27
Deleting Profile via Application	28
Deleting Profile via Frame	28
Extreme Cases for Further Consideration	29
Users with No Phone/Computer	29
Users with No Viable Pictures	29
Users with No Viable Calendar	30
Frame Operation is Too Complex	30
Failure Cases	30
User Information is Compromised	30
Application Failure	31
Frame Broken	31
Appendix: User Survey Feedback	32
Control	32
Social Media	32
Usefulness	32

Table of Figures

Figure 1:	Early Sketch of Personalizable Display Frame Design	4
Figure 2:	Block Diagram of system components	5
Figure 3:	Exploded-view diagram of hardware	6
Figure 4:	Personalizable Display Frame shell with button bar	7
Figure 5:	The “button bar” used to hold the button interface	7
Figure 6:	Overview of the provided peripherals on Raspberry Pi	9
Figure 7:	Basic Login Page	13
Figure 8:	Basic Home Page	14
Figure 9:	Basic Manage Photo Page	15
Figure 10:	Basic Manage Calendar Page	15
Figure 11:	Manage Account Page	16
Figure 12:	Linking/De-linking	17
Figure 13:	JavaScript example profile	18

Executive Summary

Personalizable Display Frame is a product intended to allow users to make hotels, offices, and other shared spaces more homelike by displaying their own pictures and schedules on an installed frame. Users will be able to use a phone or computer to access a web application that will allow them to upload pictures and calendar information. This information can then be retrieved by a physical display frame, allowing the user to personalize any location where a frame is available to them.

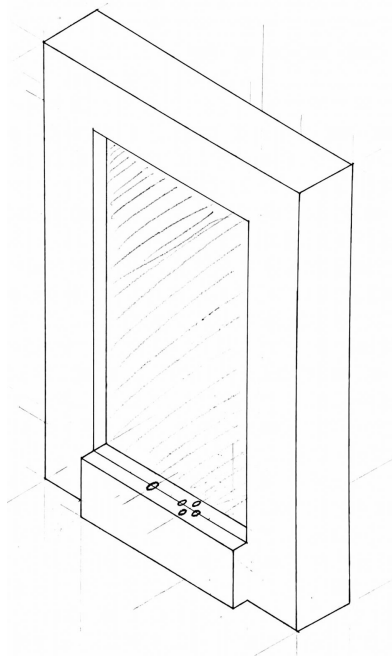


Figure 1: Early sketch of Personalizable Display Frame design

There are two main modules to our system: the frame and the server. The frame (as seen in Figure 1) is responsible for retrieving, and displaying a user's photographs and calendar files. The server is responsible for storing users' profiles and data, and hosting the applications for interfacing with the system. This document discusses the design of the following features:

- ❖ Hardware
 - Exterior Components
 - Electronic Components
- ❖ Software
 - Embedded Software
 - Application Software
 - Database Software

Design Overview

Personalizable Display Frame has two main modules: frame and server. Figure 2 below gives a block diagram representation of these two modules, and the different subcomponents that comprise them. The frame module represents the frame shell, display screen, and embedded electronic components that will be utilized on-site by users (See Figure 1). The server module represents the live database server that manages user profiles and directs user data to the correct frames.

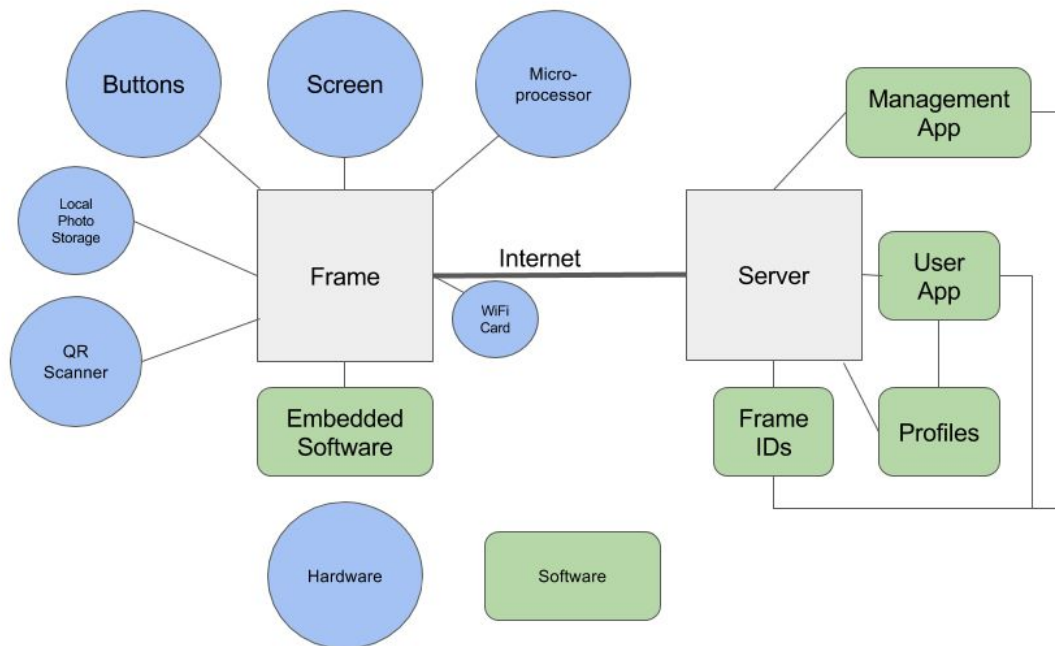


Figure 2: Block diagram of system components

This document discusses the development necessary for this product in terms of the hardware design and software design steps required. The hardware design portion discusses the creation of the physical frame and embedded system, whereas the software design section discusses the different software applications necessary to run the system.

Hardware Design

There are two main hardware factors to take into account when creating this product: the external physical structure of the frame and the internal electronic components that provide the desired functionality. Figure 3 shows an in-depth view of the frame hardware.

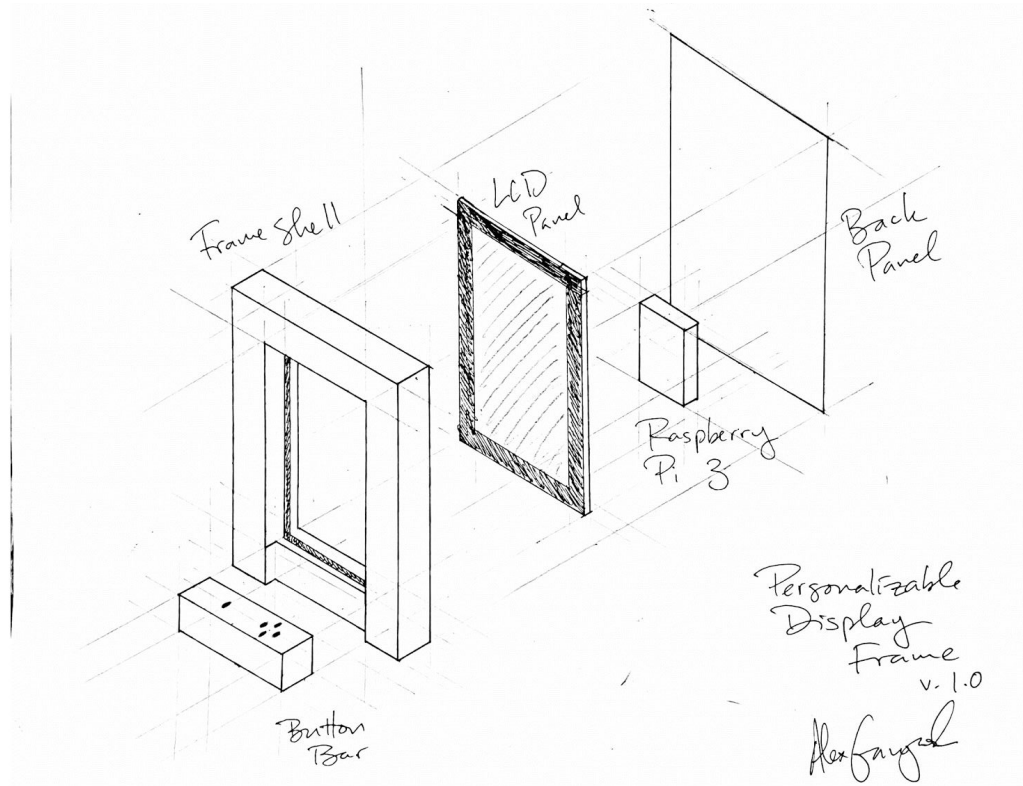


Figure 3: Exploded-view diagram of frame hardware

This exploded-view diagram shows a design of the frame with a bottom-mounted button bar, an inset LCD panel, and the Raspberry Pi 3 affixed between the LCD panel and the back panel. Note that the modularity of these components allows for some quick rearranging of this scheme to create several variations of the design (i.e. placing the button bar along the top instead of the bottom).

Exterior Components

The main exterior components of this product are the frame itself (including the built-in display screen), the button interface, and the power supply interface. These components provide the physical means by which the device can be utilized, manipulated, and powered.

Frame

The screen and electronic components of the Personalizable Display Frame will be housed by an external frame shell. Our initial design calls for a 10.1 inch monitor, which translates to a 125.28 mm x 222.72 mm screen. A 40mm front bezel around the entire perimeter brings the total frontal area to 205.28 mm x 302.7 mm. Figure 4 below shows an early sketch of the frame shell layout. Refer to Figure 3 above for more information about how the shell fits in with the other hardware components.

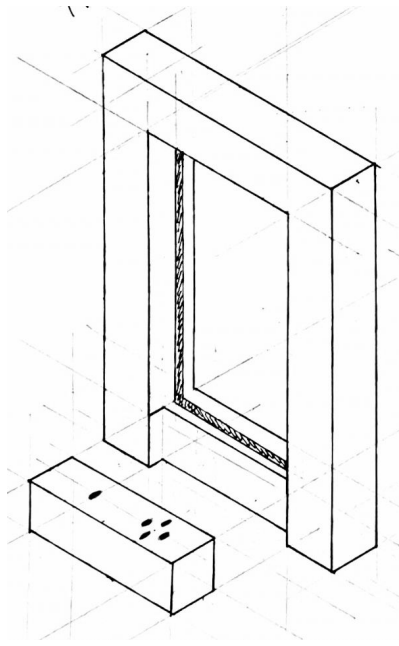


Figure 4: Personalizable Display Frame shell with button bar

Button Interface

Our design uses tactile push button switches with dimensions of 6mm x 6mm x 5mm. 4 such buttons are placed in an orientation that resembles a D-pad, and one additional button is utilized as a “Select” button. These buttons are located on a button bar beneath the display screen. This bar provides a flat plane perpendicular to the bottom of the display screen, protruding 20mm outwards. This bar can be placed on any side of the frame depending on the user. Figure 4 above shows an exploded-view diagram of the button bar and frame shell, and Figure 5 below shows a close-up view of a sketch of the final implementation. For prototyping, we plan to use standard breadboard tactile pushbutton switches located in the labs throughout the school. For the final design, we plan to purchase panel mount buttons designed to look good and perform well on the exterior of electronic devices.

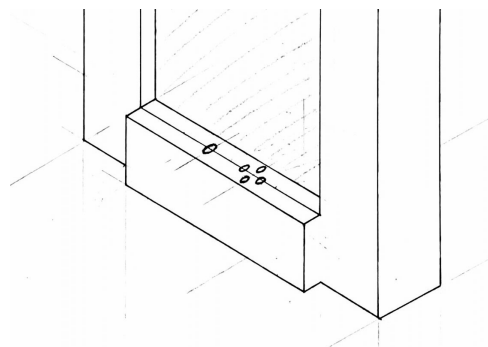


Figure 5: The “button bar” used to hold the button interface

Power Supply

Raspberry Pi 3 requires a 5.1 V micro USB supply from its manufacturer's specifications. The USB supply is given from a wall adapter having 5V output. The current requirement depends on the configuration of the GPIO pins and the different devices connected to it. The GPIO pins and HDMI port uses 50mA while the camera module requires 250mA current. When selecting an appropriate wall adapter to power this design, it is necessary to check the power rating of each device and plan the power required for proper functioning. We suggest using a wall adapter with around double the capacity for the worst-case power usage scenario.

Electronic Components

To provide the necessary functionality, the Personalizable Display Frame will need a few critical electronic components. First, a microcontroller to provide the processing power and act as the central hub for data reception, transmission, and storage. Second, a Liquid Crystal Display (LCD) screen capable of receiving and displaying image data from the microcontroller. Third, an expandable memory module that will allow users to customize the amount of on-board storage at their disposal. Finally, the frame will require a WiFi card in order to communicate with the Internet via wireless internet.

Microcontroller (BCM2837)

For our prototype version we use Broadcom chip BCM2837. It is the built-in chip in Raspberry Pi boards. The architecture of the chip is quad-core ARM Cortex A53 (ARMv8) cluster. The core has a frequency of 1.2 GHz. The chip contains the following peripherals which can be accessed by the ARM processor: timers, interrupt controller, GPIO, I2C master and Slave, UART, PWM, DMA controller and I2S. Figure 6 shows the different peripherals connected to the Microcontroller (See Raspberry Pi Model 3 Technical Specifications in references).

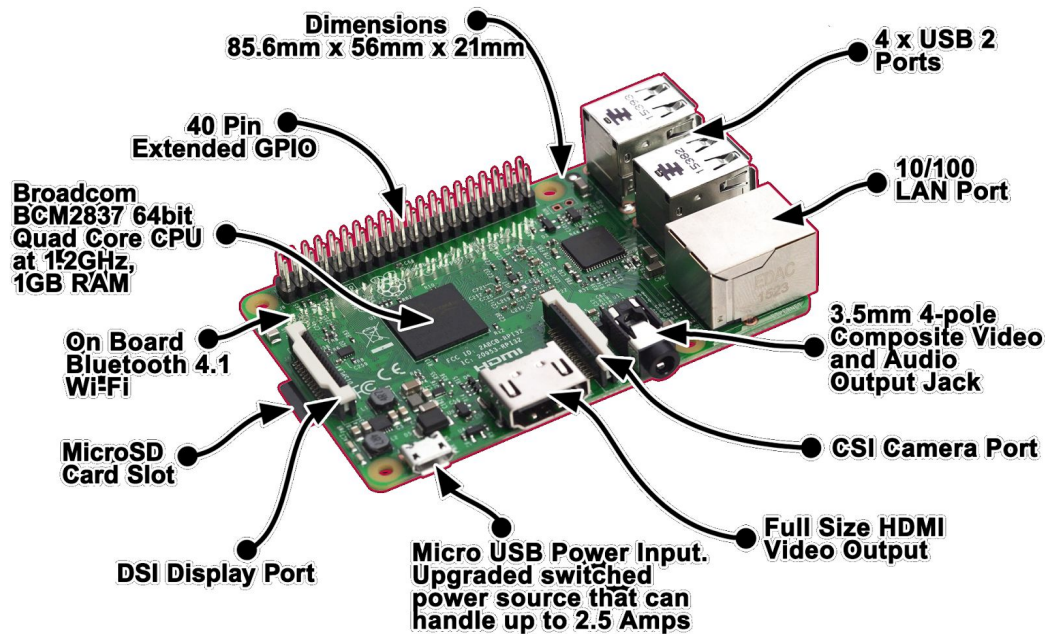


Figure 6: Overview of the provided peripherals on the Raspberry Pi

Our design uses UART communication for transfer of data from an external peripheral device. The data to be transmitted can be 8 bits along with 2 start and stop bits. We use GPIO14, GPIO15, GPIO16 and GPIO17 as TX, RX, CTS and RTS pin configuration. We use Rising Edge Detect for data transmission. There are 40 GPIO lines with each having alternative functions. For connection with the Wifi card, GPIO 22 to GPIO 27 are used. When different input - output devices are connected, one should keep in mind the alternate functions of the pins to be used.

Liquid Crystal Display (LCD)

We use a 10.1 inch TFT display screen of resolution 1024x600 compatible with Raspberry Pi 3. The screen is connected to the Raspberry Pi using an HDMI cable. It has a contrast ratio of 500:1 and typical brightness of 90. It has a power supply of 12V/1A. The contrast and brightness of the screen can be adjusted using the buttons located beneath the frame. We have chosen this particular screen because of lower heat release, less power consumption, higher screen life, high degree of visual comfort due to less shiny screens, and perfect stable and clear images (See “10.1 inch Capacitive...” in References).

The HDMI cable uses Transition Minimized Differential Signaling (TMDS). It reduces the clutter and bandwidth during transmission but also increases the presentation quality of the data. TMDS uses Low - Voltage differential sampling which compares the signal with each other and not the ground thus eliminating the factor of noise. It also uses a TMDS algorithm, which manipulates the data that is in the byte, with the goal of making it the most easily transmitted and least likely to be

damaged data possible. There are two stages of this process, each of which the algorithm selects automatically. When transmitting data, TMDS adds two control bits (a one or zero) to the data byte, making it a 10 bit byte. These two control bits tell the receiving piece of equipment what manipulations were done to that particular byte, so that the algorithm can decode the byte. The screen displays the images that were transferred from the memory by using the TMDS channel and algorithm, thus making a screen only a hardware concern (See “How HDMI Works” in References)

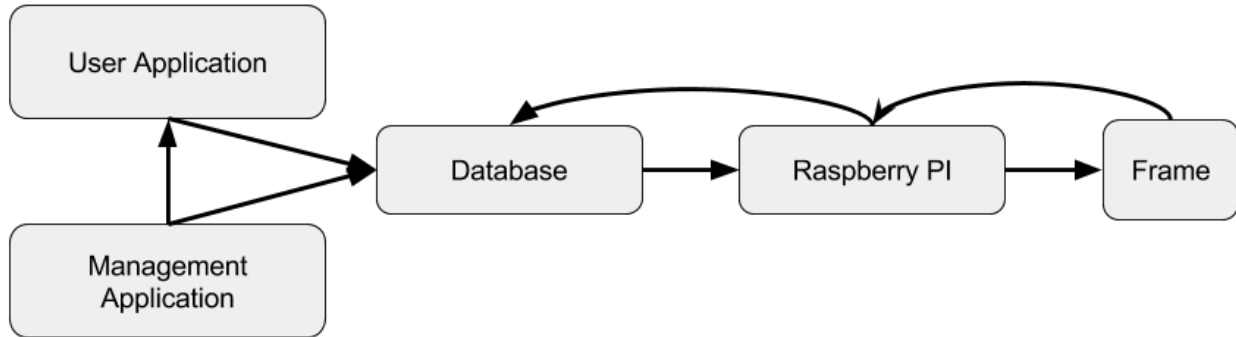
Expandable Memory

Raspberry Pi 3 has a expandable memory card slot for adjustable storage. The board provides 1GB of SRAM, so storage of an image buffer to the HDMI driver should not be a concern. We use a class 4 micro-SD card that has storage capacity greater than 8GB for storing the images along with the software used for controlling the Raspberry Pi. The SD card has a minimum writing speed of 4MB/s which is sufficient for writing images fast enough for display in full HD.

WiFi Card

Raspberry Pi 3 has an built-in wifi chip, the BCM43438, with Bluetooth 4.1 functionality. It is manufactured by Cypress Semiconductor. The BCM43438 supports 2.4 GHz WLAN IEEE 802.11 b/g/n wireless networking. The WLAN host interface supports SDIO modes providing transfer rate up to 200 Mbps. The chip supports three different functions depending on the block size to be transferred. It provides the ability to map an interrupt signal on a GPIO pin. SDIO pins can be configured in 4-bit, 1 bit or gSPI mode. It supports all the features specified in IEEE 802.11 base standard and amended by IEEE 802.11n. The wifi chip will also support secondary users by connecting to the same network as the frame is connected (See Raspberry Pi Documentation in References).

Software Design



The software requirements for the Personalizable Display Frame can be divided into three main categories: embedded software, application software, and database software.. The embedded software controls the operation of the components located inside the frame itself. The application software is the online interface for both primary and secondary users to access and edit their account and/or photograph/calendar information. The database software describes the means by which both user and frame information is stored and manipulated.

Embedded Software

The embedded software that will be programmed onto the Raspberry Pi 3 and control the interior components of the Personalizable Display Frame will have three main purposes. First, it will be responsible for managing the retrieval and storage of all necessary user information. Second, it will be responsible for creating appropriate images for display from the user's calendar file and for selecting and transmitting the correct image to the display interface. Third, it will need to listen for interactions with the frame via the exterior buttons and create appropriate responses.

Development Choice

The Raspberry Pi 3 was chosen for its speed, memory, built-in Wifi, and extensive libraries which simplifies the development process. The Raspberry Pi will be developed using Python since most of the libraries are in Python. We chose this platform in order to minimize development time and to prototype rapidly. Future designs should consider the benefits of using other platforms. For example, using a PIC microcontroller with a custom PCB could allow for a smaller chip and more flexibility in the frame design, while still taking advantage of many existing PIC libraries and utilities. Using something even lower-level, like an 8051 for example, could allow for an even smaller board and a

very lightweight assembly language program. These choices would increase development cost, but in exchange for additional customization and flexibility.

Data Retrieval

The frame should start in a default state in which it will wait for a linked profile from the database. Once the frame is linked to a user profile, it will download the associated photos and calendars from the database and store them inside the expandable memory module. The photos will be stored in the order that they were organized in the profile. After retrieving the files, the Raspberry Pi will periodically check the database for updated information about the profile it is linked to. Once the profile is delinked, either by reset or application, the frame will delete the profile's photos and calendars to create space for future profiles to be linked.

Image Display

The initial image should be a prompt and directions for linking a profile to the frame. Once linked, the first photo in the profile's associated photos will be displayed. The Raspberry Pi will retrieve the photo from the expandable memory and display it on the LCD screen using an image viewer on the Raspberry Pi. If a button is pressed to change the currently displayed photo the Raspberry Pi will retrieve the next or previous photo in memory and display it on the screen. If the currently displayed photo is the last photo in memory and a button to display the next photo is pressed, the first photo in memory will be displayed and vice versa. If a button is pressed to change between photo and calendar display, the Raspberry Pi will display the calendar file using a calendar viewer on the Raspberry Pi.

Physical Interface Control

Two different button classes, the select and the directional classes, control what is displayed on the LCD. The button inputs for both classes will be interrupt driven from the GPIO pins that the physical buttons are connected to. The select class is used to link, switch between photographs and calendars, and reset the frame. This class will have an input from one button and will store values for if the button is currently being pressed and for the amount of time the button has been held. The directional class is used as to navigate between photographs as well as within the calendar. The class will take inputs from four buttons and store values for button pressed and for button direction. Detailed uses for the buttons can be found in the Use Cases and Scenarios Appendix under Change Displayed Information.

Application Software

The application software is a web application that can be located on the same server as the database, and provides the public-access interface for the users of the Personalizable Display Frame. There

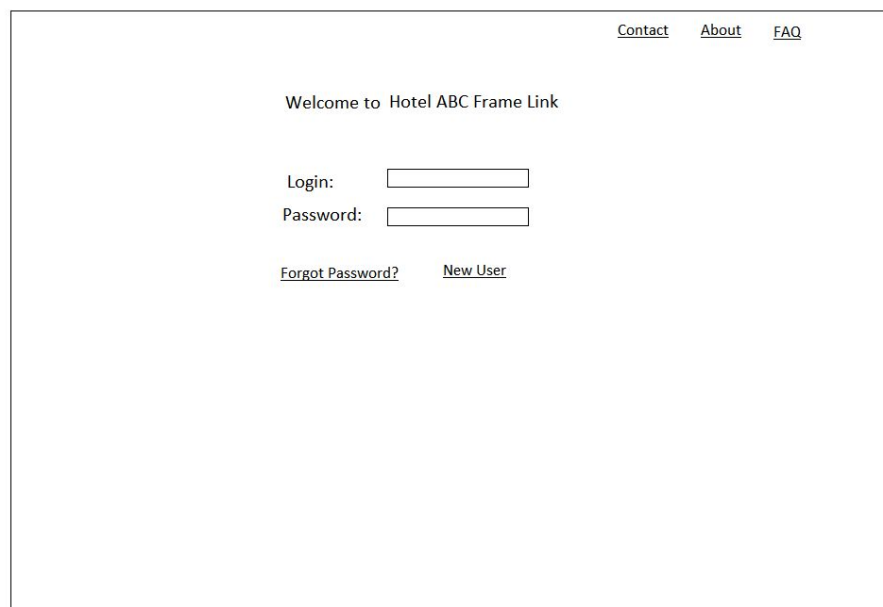
will be two classes of applications, user and management, respectively targeting the primary and secondary users of the product.

For our design, we will implement the web applications using the three main cornerstones of Internet development: Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript. HTML and CSS will construct and style the interface through which users will be able to interact with both the Personalizable Display Frame and their profiles, while JavaScript will provide the core functionality for these interactions.

User Application

The user application needs to provide the ability for primary users to create, edit, and delete their profiles. This application will have functionality for adding photographs/calendars and linking to specific frames. The application will be structured into 5 distinct pages (plus the login page), herein referred to as Home, Manage Photographs, Manage Calendar, Manage Account, and Link.

The login page should be a standard login interface prompting the user to input an email and password. If they successfully enter credentials that match a user in our database, the application then routes them to the Home page. Figure 7 shows the basic login page with different links to different pages.



[Contact](#) [About](#) [FAQ](#)

Welcome to Hotel ABC Frame Link

Login:

Password:

[Forgot Password?](#) [New User](#)

Figure 7: Basic Login Page

The Home page should provide a welcoming message to the user and an easily utilizable interface for navigating to each of the other 4 pages of the application. An omnipresent navigation bar should allow users to travel to every page of the application from any other page, and can be implemented with HTML and CSS. Figure 8 gives an example framework for the Home page.

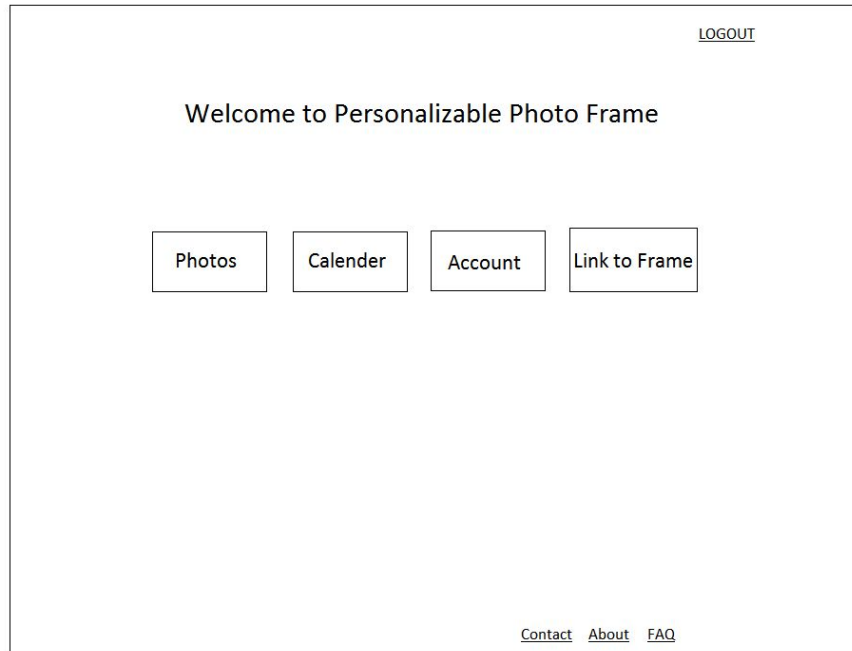


Figure 8: Basic Home Page

The Manage Photographs page will create the interface users will use to add and remove their photographs. It should contain easily viewable and accessible buttons for uploading and editing photos. These buttons will be tied to JavaScript handlers that will provide the desired functionality. “Upload” will bring up a prompt that asks the user to select photographs from local storage on the device with which they are accessing the application. Once files are selected and uploaded, they should be examined by the application to determine their compatibility with the system. If they are incompatible for any reason (wrong file type, not a photograph, etc.) the application should prompt the user with a message explaining the failure and a potential recourse. “Edit” will bring up a prompt asking the user if they would like to select or delete photographs. “Select” should then provide an interface for choosing the 5 photographs that will be kept in local storage on the frame itself out of the 30 possible photographs in server storage. “Delete” will provide an interface for the user to select photographs they would no longer like to include in our database. Figure 8 shows the manage photographs page.

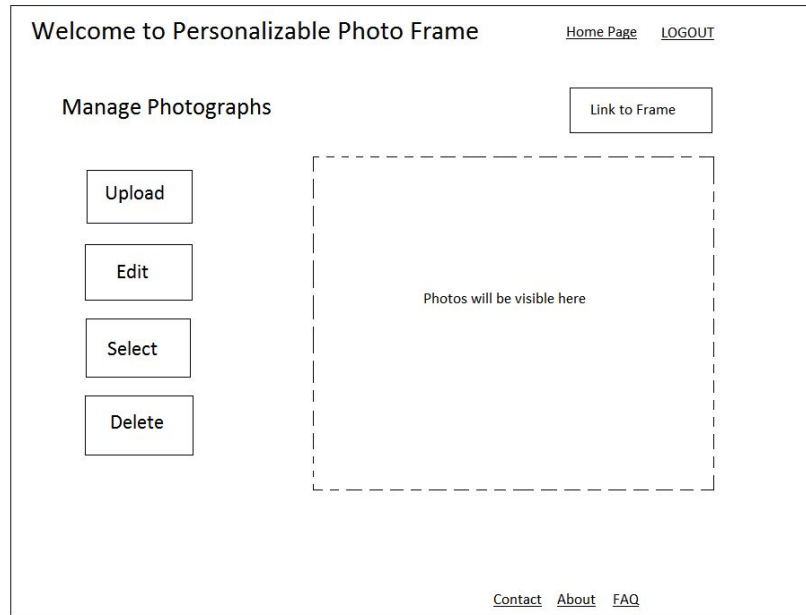


Figure 9: Basic Manage Photographs Page

The Manage Calendar page should mirror the Manage Photographs page almost exactly in both form and function, with the exception being the file type handled. Figure 9 shows example layouts for the structure of these pages. Note that the Manage Photographs page can include a preview section, with small versions of the user's photographs for quick reference.

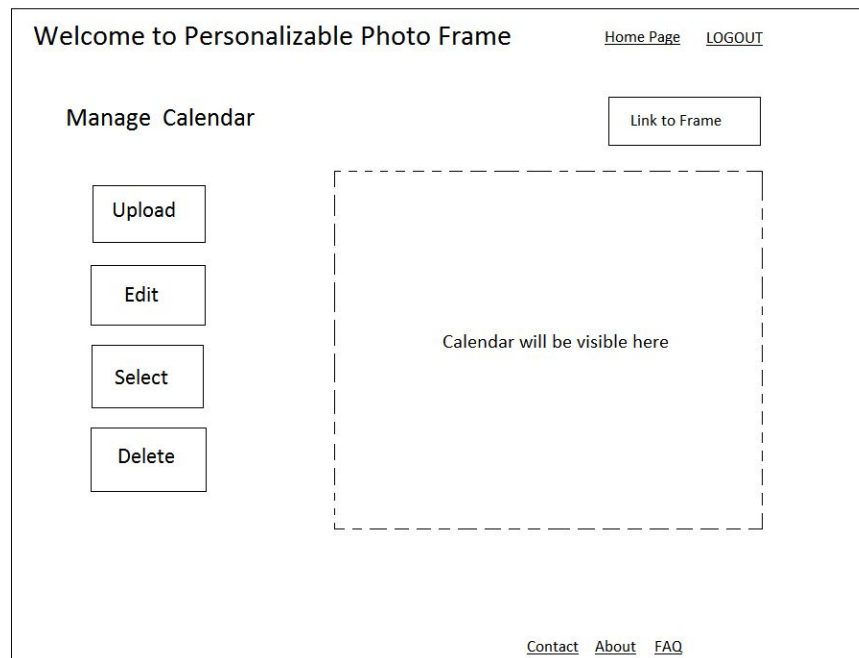


Figure 10: Basic Manage Calendar Page

The Manage Account page will provide the tools for the user to edit the personal information associated with their profile. There should be clearly labelled buttons for manipulating each of the appropriate data fields in the profile database class, and an option to completely delete the user's profile as well. Figure 11 shows an example of the Manage Account page. When changes are submitted by the user, the application should send a notification to the database to update the user's records.

The screenshot shows a web page titled "Welcome to Personalizable Photo Frame" with navigation links for "Home Page" and "LOGOUT". The main heading is "Manage Accounts". Below this, there are input fields for "Name:", "Address:", "DOB (optional):", "Tel No:", "Email:", and "Password:". At the bottom of the form area are two buttons: "Update Info" and "Delete Account". At the very bottom of the page are links for "Contact", "About", and "FAQ".

Welcome to Personalizable Photo Frame [Home Page](#) [LOGOUT](#)

Manage Accounts

Name:

Address:

DOB (optional):

Tel No:

Email:

Password:

[Contact](#) [About](#) [FAQ](#)

Figure 11: Manage Account Page

Finally, the Link page will control linkages between user profiles and physical frame devices. This page should attempt to utilize the user's camera (on their web device) if available to scan a QR image generated by a frame that has been placed in Link mode. If a code is successfully scanned, the application should compare the result with the set of known frame information to see if a frame has been successfully matched. If it has, the frame is notified of its new profile and the requisite information is downloaded for display. Figure 12 gives an example of what the Link/De-link page could look like.

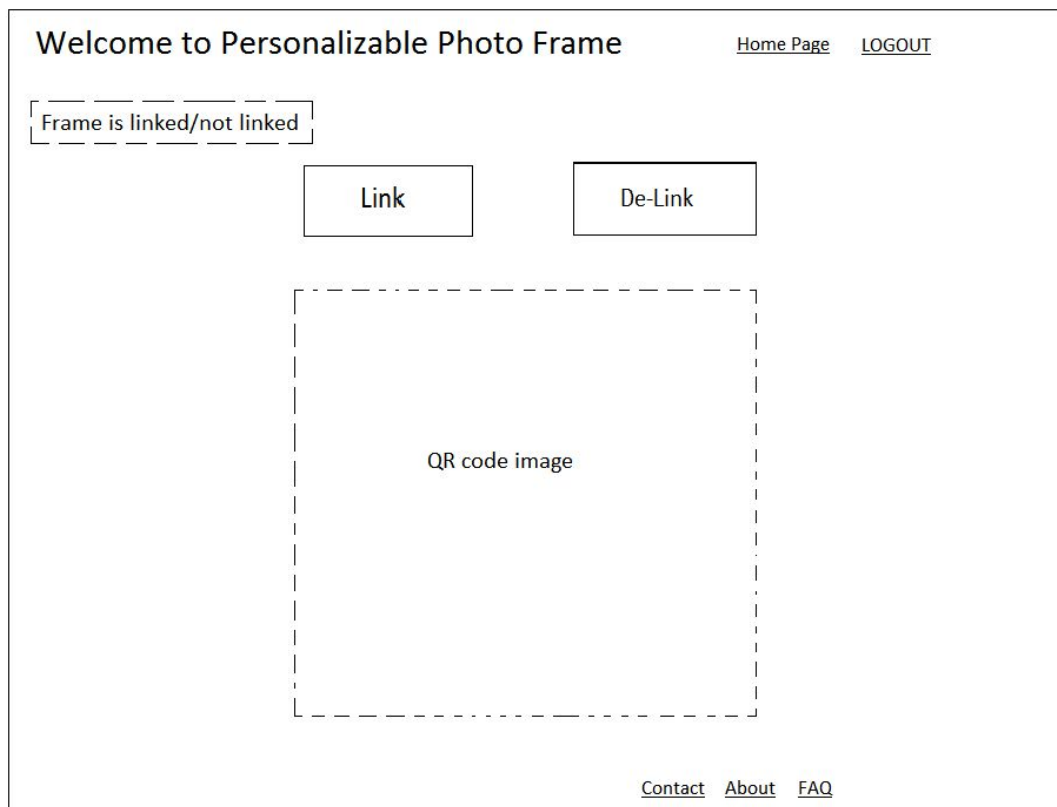


Figure 12: Linking/De-linking Page

Management Application

The management application should look essentially the same as the user application. However, it will differ by a few distinct details. First, the manager application will need to include a “Manage Frames” page in place of a “Link” page, to provide the interface for managers to register all of their purchased frames to their profile. The Manage Frames page should provide the options to reset frames remotely, de-link users from frames, and add and remove frames from the manager’s profile.

Second, the “Manage Photographs” and “Manage Calendar” pages will not be necessary in the management application unless a decision is made to interweave primary and secondary user functionalities throughout the applications. Currently the design is to keep these functions in separate applications.

Database Software

The goal of the database software is to manage two basic sets of information: user profiles and frame registrations. User profiles uniquely identify users and store their photographs and calendar files.

Frame registrations hold information on frames that are currently live, for the purposes of identification and user linking.

To implement our database, we will take advantage of JavaScript Object Notation (JSON). This will allow us to manipulate data as a JavaScript object via a web application, and then convert that data into text-only JSON format for easy transmission between our server and any clients. Note that the JSON format, as it is text-only, can be utilized with any desired programming language and not only JavaScript. In this document we will describe our database as the suggested JavaScript objects for use with the application software (see previous section). The JavaScript function `JSON.stringify()` can be used to easily convert these objects into JSON notation that can be stored on the server as a text file.

Suggested System Structure

There are many different ways the database system can be organized depending on the scope and needs of the implementation. For our design, we recommend a two-tier approach: one first-tier database responsible for storage and routing of all user data, and many second-tier databases hosted at local hubs. For example, a hotel using several of our frames in their building would have a second-tier server responsible for contacting the first-tier database and downloading the profiles and data for users linked to the hotel's frames. Once this information is stored on the second-tier server, it can use the hotel's LAN or WLAN to distribute data to the frames.

Profile Management

User profiles need to contain enough information to uniquely identify every user. This could include a set of information made of things like first name, last name, user id, email, password, etc. For our implementation, we will describe our user's profiles using a JavaScript object with fields for first name, last name, email, password, `is_management`, `photos`, `photos_select`, `calendar`, and `frames`. See Figure 13 for an example of what this would look like as a JavaScript implementation.

```

1 ▼ var user1 = {
2     firstName: "John",
3     lastName: "Doe",
4     email: "jdoe@gmail.com",
5     password: "hunter2",
6     is_management: false,
7 ▼   photos: [
8       "jdoephoto1.bmp",
9       "jdoephoto2.bmp",
10      "jdoephoto3.bmp",
11      "et_cetera.bmp"
12   ],
13 ▼   photos_select: [
14       "jdoephoto1.bmp",
15       "jdoephoto3.bmp"
16   ],
17   calendar: "jdoecalendar.csv",
18   frames: null
19 };
20
21 ▼ var manager1 = {
22     firstName: "Buzz",
23     lastName: "Aldrin",
24     email: "buzz@nasa.gov",
25     password: "moonman3000",
26     is_management: true,
27     photos: null,
28     photos_select: null,
29     calendar: null,
30 ▼   frames: [
31       "frame_id_xxx",
32       "frame_id_yyy",
33       "frame_id_zzz",
34       "frame_id_etc"
35   ]
36 };

```

Figure 13: JavaScript example profiles

The first name, last name, email, and password fields are used for user identification and authentication. Data can be extracted from these fields and used to provide personalized information on any of the application screens (i.e. once the user has signed in, their first name and last name can be extracted and displayed in a welcome message on the Homepage of the application. When users attempt to login to our application, the email and password they input will be compared against the emails and passwords stored in the database. If the user's combination is found they are considered authenticated and allowed to access their profile.

The `is_management` field is used to provide a quick reference as to whether the user is registered as a primary or secondary user in our database. Primary users will have access to the user application, but will be denied access to the management application. Likewise, secondary users will have access to the management application but will be denied access to the user application (unless they make a

separate account as a primary user). To register as a secondary user, the user will need to provide an appropriate Frame ID number that matches an existing, purchased frame as verification of the ownership of a Personalizable Display Frame. Once verified, secondary users will be able to add additional frames to their profile using their Frame IDs.

The photos field is an array that contains the names of the photograph files associated with that user's account (up to 30 files for our design). The photos_select field contains the file names of the photos chosen by the user for local storage on a Personalizable Display Frame (up to 5 files for our design). Extracting the data from this field will allow the frame and application software to request any desired photographs from the database server.

The calendar field will contain the name of the calendar file associated with the user. Extracting this name from the JSON text file will allow the frame and application software to request the desired calendar file from the database server.

The frames field has different purposes depending on the class of user. If the user is classified as a primary user (i.e. is_management is false) then the frames field will contain the Frame ID of whatever frame the user is currently linked to (if any). If the user is classified as a secondary user (i.e. is_management is true) then the frames field will be an array containing all the frames currently registered to the user's account.

Frame ID Management

Each frame will have a unique serial number identifier and a unique MAC address that can be used to identify that particular frame.. The serial address will be used to uniquely identify the devices when they are communicating with the server. When a profile is linked to the picture frame the serial ID will be sent as well as a request for the profile. The database will then send the data over the internet to a main hub that controls the frame in a given area (hotel, etc.). From the hub the data will be sent over LAN to the picture frame. The MAC address will be used to uniquely identify the device on the network. We intend for the MAC addresses to be dynamically assigned via an ARP request.

Linking

When a user wants to link a profile to the picture frame, they will submit a request to the server. This request will contain the the user's ID and password and the frame's serial number in a JSON text file. The server will extract the user's ID and password from the JSON and attempt to authenticate the user's ID and password. If successful, the user's profile and the frame's profile (found using the serial number) are "linked" in the database. The server will then read the user's profile and send any selected photographs or calendar files to the frame.

Future Considerations

Due to a variety of factors like cost and complexity, certain design choices were made in the development of this product that led to the use of the materials outlined in this document. This section discusses some of the features that were left out of this iteration, with considerations for how it might conceivably be implemented in future versions. Additionally, it provides some basic guidelines for selection of alternative electronic hardware if so desired.

Additional Features

These features were discussed during the planning of the Personalizable Display Frame, but ultimately were left out of the first iteration. Future versions of this product should consider the benefits of adding these features as they compare to the additional costs and complexities.

Music

A speaker for playing user's music can be added as a part of the system. Preferences should be given to streaming music over local music files in order to conserve memory--this could potentially be achieved by having users associate their Spotify accounts or other music streaming services to their profiles.

Note that the biggest obstacle to the inclusion of music is cost. Providing quality speakers in a small form is no inexpensive feat, and using cheap speakers might turn users away from our product. It is important to consider the balance between cost and benefit carefully for this case.

Videos

Another possible next step for Personalizable Display Frame is the addition of video-playing functionality. With a display module already in place for showing photographs it should be possible to add a video driver that would allow this feature to be added in a simple manner.

RFID

An RFID scanning module can be added to the product in order to add more responsive user identification. The objective would be to recognize a user automatically when they enter the room by reading a unique RFID chip that they carry somewhere on their person. This would remove the hassle of having to incorporate QR scanning for device linkages. The initial requirement will be to first develop a fully comprehensive system to ensure there is no false match i.e will always recognize the correct user.

Caller ID

Future versions could display a user's caller ID information on the frame. This would require a bluetooth connection between the user's phone and frame, which would add to the development cost of both the user application and the embedded software. Note, however, that the Raspberry Pi 3 does support Bluetooth, so the necessary hardware is already in place.

Flight/Local Information

The frame could also be capable of displaying a user's flight information or information about the local surroundings, particularly targeting travellers in hotel rooms. A new interface would need to be created for the user to input the necessary information about their flight. Additionally, a method for retrieving the information would be required as well. This could potentially be accomplished through the use of Rich Site Summary (RSS) feeds to extract data from websites.

Adjustable Lighting

Finally, adding an adjustable lighting panel or stand to the back of the frame, to allow users to have personalizable accent lighting in their room. Our vision for the eventual inclusion in future products is to have a controller in the user application that would allow the user to select brightness, color, etc. of the lighting, and that information would be retrieved and applied at the frame.

Alternative Hardware

When selecting our hardware, a strong emphasis was placed on using existing developmental components instead of assembling custom PCBs ourselves. While this was done to keep rapid prototyping easy, it is also certainly possible to design a cheaper and more purpose-built circuit board from discrete components. This section discusses the considerations that should be made for each of the major electronic pieces, as well as a discussion of the tradeoffs for different design decisions.

WiFi Chip

For our prototype we use BCM43438 designed by Cypress Semiconductors. For large scale production we can use ATSAM 25- MR210PA. It performs many of the MAC functions, including but not limited to association, authentication, power management, security key management, and MSDU aggregation/de-aggregation. It has 768 bits of volatile memory. It supports IEEE 802.11b/n/g with 20 MHz bandwidth. Another alternative is using ESP8266 wifi module. It support SPI, SDIO, I2C & UART communication and works at 3.3 V. It has a RAM size of less 36 kB and requires external SPI flash memory.

Note that when deciding between a WiFi chip and a WiFi module it is important to take into account potential extra storage and software considerations. While oftentimes modules come with their own software onboard, standalone chips often rely on another microcontroller and memory unit to control them.

External Memory

An external memory card of greater than 512 MB is required to store the necessary files required for smooth running of the product along with the user selected files. The SD card module should be compatible with microcontroller as it will be required to store the necessary information of the user.

Microcontroller

If choosing a standalone microcontroller instead of one embedded in a development board, an option is the ATMEGA2560-16AU, which has 86 programmable I/O pins. This fulfills the requirement needed for connection of different peripherals that are required by the project. It also contains USART interface that can be used for establishing the connection between the wifi chip and the microcontroller. It has very low power consumption of 500 uA at 1.8V.

Cost of the Project

The overall cost of the prototype is around \$150 to \$200, according to the following approximate breakdown.

- The cost of Raspberry Pi 3 is around \$80 to \$90, depending on vendor.
- The cost of an 8GB SD card for is \$13.
- LCD screen costs \$60 to \$80.
- A pack of 20 push buttons is \$3.
- Power Adapter cost is \$10.

References

Raspberry Pi 3 Model B Technical Specifications. (n.d.). Retrieved March 10, 2017 from <https://www.element14.com/community/docs/DOC-80899/l/raspberry-pi-3-model-b-technical=specifications#>

Raspberry Pi Documentation. (n.d.). Retrieved March 17, 2017, from <https://www.raspberrypi.org/documentation/>

10.1 inch Capacitive Touch Screen LCD 1024×600 HDMI with Bicolor Case for Raspberry Pi/BB BLACK/PC Systems - US. (n.d.). Retrieved March 17, 2017, from https://www.newegg.com/Product/Product.aspx?Item=9SIA3525527044&ignorebbbr=1&nm_mc=KNC-GoogleMKP-PC&cm_mmc=KNC-GoogleMKP-PC-_-pla-_-Monitors%2B-%2BLCD%2BFlat%2BPanel-_-9SIA3525527044&gclid=CjwKEAajwkq7GBRDun9iu2JjyhmsSJADHCD_H1ZiCRbkjHkZpEJjuvM7FdactVr7AW_4NnLjnRuhw6RoCBjrw_wcB&gelsrc=aw.ds

"How HDMI Works". *HowStuffWorks*. N.p., 2017. Web. 17 Mar. 2017.

Appendix: Use Cases and Scenarios

This appendix is intended to summarize our examination of different use cases involved with the operation of Personalizable Display Frame, as well as to list a few extreme cases that should be researched further and a few failure cases to be accounted for.

Use Cases

This section details a few of the different use cases we considered when developing the Personalizable Display Frame. Note that while this section does not exhaust every possible scenario, it still provides context on both normal and abnormal usage of the product.

Create a Profile

Summary:

The primary user creates a profile in our database for associating his choice of pictures and schedules with his particular user id. The user will be prompted to create a new account upon first access to our web application.

Preconditions:

- User has a phone or computer
- User has internet access
- Server is live and connected

Course of Events:

1. User visits webpage
2. User is prompted to log in or create a profile
 - a. If they select log in, they give their current username/password and access their account
 - b. If they select create a profile, they are taken to the profile creation screen
 - i. Here, they are prompted to create a username/password
 - ii. Email-account association to help prevent spam/DDOS
 - iii. Once account creation is completed, users will be routed to the profile management screen, where they will be able to add/remove pictures and schedules, and change their account settings.

Upload Pictures

Summary:

The user will have the ability to add up to 30 photos to their account, and select 5 photos for local storage. These photos will be the set that is rotated through upon use of the buttons mounted on the physical frame.

Preconditions:

- User has a phone or computer
- User has an internet connection
- User has a viable account with our database
- User is logged into said account
- Server is live and connected
- User has viable photographs for upload

Course of Events:

1. Once user is at account management screen, they select the “Manage Photographs” button
2. User is directed to the Manage Photographs page, which contains an “Add Photographs” option:
 - a. If less than 30 photographs, user will be able to continue to upload photos at will
 - b. If at 30 photographs, or current upload would take user beyond 30 photographs, user will be prompted to delete a photograph in order to continue with upload

*Upload Schedule**Summary:*

The user will be able to add their schedule, either in an appropriate CSV format, or by direct association with their Google account.

Preconditions:

- User has a phone or computer
- User has an internet connection
- User has a viable account with our database
- User is logged into said account
- Server is live and connected
- User has a viable way of acquiring calendar in correct format

Course of Events:

1. Once user is at account management screen, they select “Manage Calendar”
2. User is directed to calendar management screen
 - a. If no Google account is associated with the profile, the user is prompted to link their Google account
 - i. If user has been asked before, database remembers and doesn’t prompt again unless user changes this in account settings
 - b. If user has successfully linked Google account, Google API is used for retrieval of calendar files automatically
 - c. If user opts not to use Google account, they will have the option of uploading a calendar in a CSV format by selecting the “Upload Calendar” button
 - i. Upon upload, calendar file is examined to ensure compatibility with our platform
 1. If file is compatible, the calendar information will be associated with the user’s profile

2. If the file is incompatible, an error notification will be displayed to the user with information on how to create a compatible file

Change Displayed Information

Summary:

The user will be able to change the displayed information on our screen by manually operating buttons mounted on the frame. Control will be by a set of two buttons: one “Select” button, used for toggling particular features, and one 4-directional “D-pad” button, used for navigation.

Preconditions:

- User has a valid account
- User’s account is associated with a particular product
- User is present in the same room as product
- Product has been successfully loaded with user information

Course of Events:

1. Scenario 1: Frame is currently on default (unlinked to profile) display
 - a. Pressing the “Select” button will begin QR code scanning (there will be directions for this on the default screen)
 - b. Pressing any direction on the “D-pad” will have no effect
 - c. Holding the “Select” button will prompt the user to reset the system
 - i. If “Select” is pressed again, system will reset
 - ii. If any direction on the “D-pad” is pressed, prompt will close with no change
 - iii. If nothing is pressed within 5 seconds, prompt will close with no change
2. Scenario 2: Frame is currently displaying photographs
 - a. Pressing the “Select” button for less than 3 seconds will prompt the user to switch to calendar display
 - i. If “Select” is pressed again, display will switch to calendar
 - ii. If any direction on the “D-pad” is pressed, prompt will close with no change
 - iii. If nothing is pressed within 5 seconds, prompt will close with no change
 - b. Pressing any direction on the “D-pad” will change the currently displayed picture
 - i. “Up” and “Right” directions will navigate to the next picture in the queue
 - ii. “Down” and “Left” directions will navigate to the previous picture in the queue
 - c. Holding the “Select” button for 3 seconds will prompt the user to reset the system
 - i. If “Select” is pressed again, system will delink from profile and reset
 - ii. If any direction on the “D-pad” is pressed, prompt will close with no change
 - iii. If nothing is pressed within 5 seconds, prompt will close with no change
3. Scenario 3: Frame is currently displaying calendar
 - a. Pressing the “Select” button for less than 3 seconds will prompt the user to switch to photograph display
 - i. If “Select” is pressed again, display will switch to photograph
 - ii. If any direction on the “D-pad” is pressed, prompt will close with no change

- iii. If nothing is pressed within 5 seconds, prompt will close with no change
- b. Pressing any direction on the “D-pad” will navigate through the calendar display
 - i. “Up” and “Down” directions will switch the calendar display from daily view to weekly view
 - ii. “Left” and “Right” directions will navigate to the previous or next day/week, depending on current view
- c. Holding the “Select” button for 3 seconds will prompt the user to reset the system
 - i. If “Select” is pressed again, system will delink from profile and reset
 - ii. If any direction on the “D-pad” is pressed, prompt will close with no change
 - iii. If nothing is pressed within 5 seconds, prompt will close with no change

Update Information

Summary:

User will be able to update their profile information from the online webpage by logging in with their user account information.

Preconditions:

- Server is live and connected
- User has an account in our profile database
- User has information associated with said account
- User has logged into account management system via webpage

Course of Events:

1. Upon visiting the account management page, there will be three main buttons for managing user information presented to the user
 - a. “Manage Photographs” - this button will take the user to a screen where they will be able to add/remove their photographs
 - b. “Manage Calendar” - this button will take the user to a screen where they will be able to update their calendar information
 - c. “Manage Account” - this button will take the user to the account settings screen, where they will be able to update their account information, including things like:
 - i. Updating username
 - ii. Updating password
 - iii. Deleting account
 - iv. Changing specific privacy settings

Discerning Between Users

Summary:

The product will be able to link to specific users via a QR code scanning interface.

Preconditions:

- The user has a valid account with our profile database
- The user is able to access their profile on our webpage via a smartphone

- If not available, then a printed version of the code would suffice

Course of Events:

1. User will go to “Account Management” on webpage
2. User will select the “Link to Device” button
3. User will be given a unique QR code image that is associated with their account
 - a. This image can be scanned directly from a phone screen, or from a sheet of paper
4. When no profile is associated, product will be in default mode, which will have some message prompting the user to scan their code
5. Upon successful scan of code at indicated scanning area, product will query database for profile associated to code, and download relevant data for that profile
6. Once data is downloaded, the product will display the user’s personal information

Deleting Profile via Application

Summary:

The user will have the ability to disassociate their profile with a frame from the account management section of the webpage. In addition, management (of the office or hotel) will have a separate application, giving them the ability to reset and/or disable frames without the ability to see private information.

Preconditions:

- User has a valid account and has associated their profile with a frame

Course of Events:

1. Scenario 1: User visits webpage, logs in, and goes to account management
 - a. If currently associated with a frame, the “Link to Device” button will be replaced by a “De-Link from Device” button
 - b. Selection of this button will result in disassociation from the physical frame
 - i. All user information will be deleted from local memory
 - ii. Database decouples frame from profile
2. Scenario 2: Management visits management webpage, logs in, and goes to account management
 - a. Here, they will be able to see the frames they have control of, as well as tags they have associated with said frames (e.g. room numbers)
 - b. From this screen, they will be able to reset frames and wipe their local storage
 - i. This will allow them to ensure they have final say on use of displays
 - ii. This will allow them to ensure rooms are reset at the proper times (i.e. when the hotel reservation ends)

Deleting Profile via Frame

Summary:

The user will have the ability to disassociate their profile with a frame via physical interaction with the frame itself. From any display screen, holding down the “Select” button for more than 3 seconds will prompt a device reset.

Preconditions:

- User has a valid account and has associated their profile with a frame
- User is in same room as frame

Course of Events:

1. User presses the “Select” key for at least 3 seconds, causing a prompt to appear asking the user if they would like to continue with the reset
 - a. If “Select” is pressed again, system will delink from profile and reset
 - b. If any direction on the “D-pad” is pressed, prompt will close with no change
 - c. If nothing is pressed within 5 seconds, prompt will close with no change

Extreme Cases for Further Consideration

[overview of extreme cases section]

Users with No Phone/Computer

Our initial specification requires primary users to access the profile interface via a web application, which will require access to a computer or mobile computing device (cell phone, tablet, etc.) with a compatible web browser. However, it would be potentially possible to accommodate the system even for users without access to such means.

There are a couple of different ways this could be achieved. For example, a USB or SD card interface could be added to frame itself, allowing the user to locally upload their profile data. Alternatively, the management application could in future versions include the ability to create and manage profiles, so users without proper means could have their profiles created for them. However, there are tradeoffs to these additions- adding a physical interface adds to cost per unit but protects user privacy, whereas upgrading the management application requires only development cost at the potential expense of user privacy. More research should be done on the pros and cons of different features targeting this issue before inclusion in future versions.

Users with No Viable Pictures

It is possible that some primary users may successfully access our webpage and create a profile, but will be unable to successfully add pictures to their profile. This could be caused by a number of reasons--for example, a user might have no photos in storage on their computer or mobile device, or a user might have photos but in an incompatible format.

It should be possible to diagnose some issues via the web application itself. For example, if a user uploads a photo that is in a format that will not display on the frame the application can recognize this and refuse the upload, while notifying the user of the error and a potential solution.

Additionally, it might be pertinent to discuss the creation of an “FAQs” page located somewhere within the web application. This page would act as a one-stop location for many answers to common questions,

including things like which photo formats are compatible or how to add photos to a location that can be accessed by the application.

Users with No Viable Calendar

See “Users with No Viable Pictures.” These users are essentially the same set, replacing viable photographs with viable calendar files. In much the same way, as many problems as possible should be diagnosed and displayed to the user on the web application side, not the frame side, when upload or linking occurs. Again, an “FAQs” page could be a good place to put relevant information on troubleshooting for this area.

Frame Operation is Too Complex

The design of the system should be careful to ensure that at no step is operation too complex for a primary user with average or even limited technical literacy. However, it is always possible that unforeseen circumstances could arise in which users are unable to discern how to operate the interface without assistance.

Careful attention should be paid to the success of the product’s first iteration, and particularly user feedback on its operation. If data shows that there are some confusing operations, or that the frame is being operated in an unintended manner, then this should be taken into consideration when revising the design for future versions.

Failure Cases

This section describes a few cases in which we would describe our project as having failed--note that this does not include all potential failures for our system, but instead represents a few issues that we believe should take priority in this design.

User Information is Compromised

We believe that user privacy should be of the utmost importance. This represents the protection of all profile information, including names, usernames, email addresses, passwords, pictures, and calendars. It also represents the protection of data locally stored on a particular frame, such as pictures and calendars.

This is an area that will require special attention during the development stage of this product, to ensure that our server and frame have sufficient protections in place to prevent compromise of user data. Future research in areas like end-to-end encryption could potentially provide even more security.

If a scenario does arise in which one or more users’ information does become compromised, it will be necessary to provide responsive support to diagnose and fix any potential flaws or vulnerabilities. This requires more labor cost over the life of the product, but we believe this tradeoff would need to be considered in order to maintain a high product standard.

Application Failure

This represents cases involving a failure of the user or management applications, or a failure by the server. This could be caused by the server losing internet connection, a software update that renders our applications out of date, a power outage, an unforeseen bug in the application code, or many other potential reasons.

Again, the best way to combat this issue is through responsive support that can investigate and diagnose issues as they occur--this would also include things like ensuring the server's software and firmware is up to date, ensuring the application is up to date, and other minor bug fixes. We again stress that we believe that this increased labor cost should be considered in order to maintain a high product standard over the life of the product.

Frame Broken

This represents the set of all cases in which the frame itself ceases to operate correctly. This could be because the frame is physically broken, because a connection inside the frame was lost, because a chip no longer works correctly, because of an unforeseen bug in the code, or a variety of other reasons.

A large part of dealing with this use case is prevention. Care should be taken in the design to ensure that all parts have an estimated longevity of at least the minimum intended life span of an individual frame, and that the structure of the frame itself is sufficiently sturdy enough to pass a selected set of drop and stress tests--this should be done during the prototyping and testing phase. Additionally, there should be consideration into how other elements of the design can lend itself to longevity. For example, changing the location of the buttons on the frame could potentially change the likelihood of the frame being knocked over or off the wall.

Once again, responsive support is the best way to diagnose and fix any of these potential issues when they arise. However, there is increased cost for these issues because of the difficulty in remotely diagnosing and solving physical issues with the frame, as well as the potential cost of replacement parts. This should be taken into account when considering any potential warranties or guarantees to be sold alongside the product.

Appendix: User Survey Feedback

This appendix contains some notes about the user survey that was conducted at the beginning of the term. The purpose of the survey was to garner feedback on various features of a potential product and determine the value each of those features had to potential users.

Control

Based on the responses we received, people seemed generally favorable to a few different methods for controlling our product. In particular, users under the age of 25 showed a strong indication that a phone app would be preferred (although other forms of control would be acceptable), whereas older users tended to favor a more widely ranging selection.

Social Media

There were mixed responses on linking our product to social media accounts. Some of this was most likely due to people's natural inclination to protect their own privacy, and perhaps more evaluation (with a more advanced line of questioning) in this area could lead to more helpful results. We believe that with the pros and cons of linking social media in this scenario laid out more carefully before them that people might respond differently.

Usefulness

Based on the responses, the majority of the people preferred music, local information, schedule and pictures to be included in the frame. People in the age range 25-50 preferred having their schedule and local information being displayed on the screen. Some people also felt lightning to be useful but there were some misinterpretation regarding the control of the same (i.e lightning of the room or the screen).

Notes

There may have been some misinterpretation with some of the questions that we asked in the survey. The one misinterpretation being that our phrasing caused people to take the question of how useful certain features are as how much they would improve productivity or something similar rather than how homelike it would make hotels/offices. Another misinterpretation being that the question on the best methods to control the display should have been made into two questions about how to change what information is being displayed as well as how to update their profile information