CIS 4911 – SENIOR PROJECT

Picture Marketing’s Social Wall

**Design Document**

Members: Juan Gonzalez-Llanos, Joseph Gonzalez

Mentor: Cortney Mills

Instructor: Masoud Sadjadi

**COPYRIGHTS AND TRADEMARK NOTICES**

This is the work of Joseph Gonzalez and Juan Gonzalez-Llanos, unless specified otherwise. Aid and guidence was provided by Cortney Mills and Louis Zuckerman of Picture Marketing.

**Table of Contents**

|  |  |
| --- | --- |
| **Title** | **Page Number(s)** |
| 1. Introduction | 4-5 |
| 1.1. Design Methodology |  |
| 1.2. Terminology |  |
| 1.3. Overview of the Document |  |
| 2. System Design |  |
| 2.1. Overview |  |
| 2.2. Subsystem Decomposition |  |
| 2.3. Hardware and Software Mapping |  |
| 2.4. Security/Privacy |  |
| 3. Detailed Design |  |
| 3.1. Overview |  |
| 3.2. Static Model |  |
| 3.3. Dynamic Model |  |
| 3.4. Code Specification |  |
| 4. Glossary |  |
| 5. Appendix |  |
| 5.1. Appendix A - Use Case diagram |  |
| 5.2. Appendix B - Use Cases |  |
| 5.3. Appendix C - Class Interfaces |  |
| 5.4. Appendix D - Diary of Meetings |  |
| 6. References |  |

**1. Introduction**

This section deals with introducing our project and defining our purpose, scope, terms, and acronyms. It also deals with describing how the rest of this document will unfold, describing our feasibility study, as well as our project plan.

**1.1 Problem Definition**

The problem our project, the Social Wall, deals with is the inability for clients to display photo albums in a slideshow from social media sites and cloud storage apps in a quick and efficient manner. While it is possible to accomplish, it is normally tedious to set up and gives you a very limited amount of customization on how you want to show your slideshow.

**1.2 Design Methodology**

The software process model being utilized is that of agile development. Regarding system requirements, utilizing agile development allowed for dynamic development of the language, as well as easing the requirement elicitation with the client. Developing an application with various unknowns, the agile development model allows the development to be created in a dynamic, reoccurring, testing oriented way. Models that will be used to represent the design are as follows

- Class Diagrams

- Sequence Diagrams

- Use Case Diagrams

- Package Diagram

- Detailed/Minimal diagrams

**1.3 Terminology - Definitions, Acronyms, and Abbreviations**

PM - Picture Marketing

SW - Social Wall

FB - Facebook

PG - PhoneGap software

h/w - Hardware

s/w - Software

OS - Operating System

App - application

RSS - Rich Site Summary

CC - Chrome Cast

AJS - AngularJS Framework

JS - Javascript language

CSS - Cascading Style Sheets language

**1.4 Overview of document**

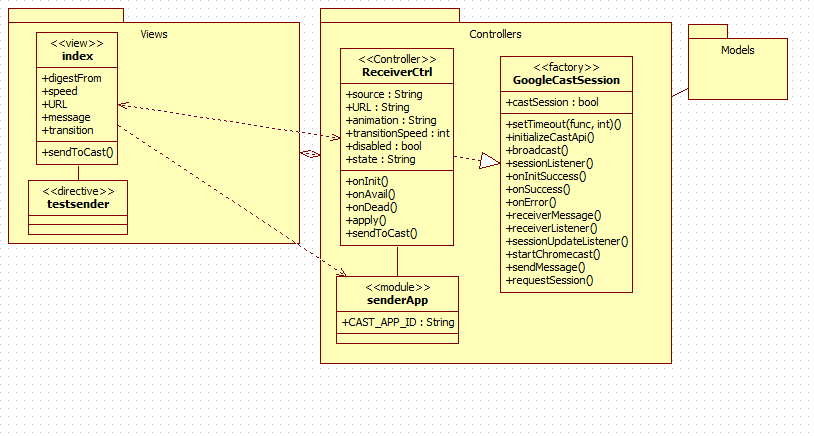
This document is broken up into 5 main chapters. Each chapter is further broken up into sections. This is the end of the first chapter. Chapter 2 focuses on the proposed software architecture of the *Social Wall*. Section 2.1 provides an overview of the chapter. Section 2.2 details the decomposition of the system into subsystems and identifies use cases associated with each subsystem. Section 2.3 focuses on the software and hardware utilized in our development process and maps each of the aforementioned subsystems to that hardware and software. Section 2.4 concerns itself with the persistent data our system utilizes and how it will be managed. Section 2.5 discusses the security concerns of *Social Wall*?

Chapter 3 is focused on the design of the system itself. Section 3.1 introduces the classes we have broken the system down into and identifies the design patterns used in the class structure. Section 3.2 is more concerned with the control flow of the system and how the various objects interact with each other. Section 3.3 is the heart of the chapter and focuses on the detailed design of the system and explains the purpose of each class. Chapters 4 and 5 are the metachapters in that they concern themselves with the content in the previous chapters. Chapter 4 serves as a glossary of terms, and Chapter 5 contains the appendices.

**2. System Design**

A piece of software’s architecture is a description of its overall structure. The *Social Wall* utilizes an MVC architecture for its different components. I.e. Model-View-Controller structure. An overview of this design will be described in section 2.1, with the succeeding sections detailing our system itself.

**2.1. Overview**

The design chosen to be used for the specific subsystems is the MVC architectural pattern, splitting the specific subsystems into three main parts, each responsible for a specific role in the system. The three parts are the Model, View, and Controller, respectively. The model is responsible for managing the data of the application. The view is ultimately the user interfaces which presents the data based on the controllers decisions. The controller of the system acts as the manager that is in charge of responding to user action and input, and is in control of what is shown in the view part of the application. Hence, the controller receives the data, validates the data, and modifies the model based on occurrences in the view. The social wall has three systems that are the chrome cast sender applications, the slideshow engine, and the custom chrome cast receiver application. 

Package Diagram of the sender application. Showing MVC Architecture.

The system as a whole can be described as a client server architecture. Where the clients utilize a mobile application and Chrome desktop browser to make requests to a server side application that can connect to chrome cast devices and displays content requested by the user.

For the creation of each subsystem, the AngularJS framework was utilized for its creation. The sender subsystem, specific to the mobile application, utilizes the Cordova and AngularJS frameworks to create a hybrid mobile application.

**2.2 Subsystem Decomposition**

The system is composed of four subsystems:

* Sender Application Subsystem
  + Sender application subsystem plays the role of interacting with the user and allowing them to interact with the system. The third subsystem is the receiver subsystem. The sender application subsystem retrieves the input from the user such as the RSS Feed URL, transition, transition speed, as well as where he would like the data to be digested from. The sender application also allows the user to create a session with chrome cast devices connected on the same network, and sending the user input to the receiver subsystem.
* Slideshow Engine Subsystem
  + Slideshow engine subsystem is in charge of the photo manipulation which is the content that will be displayed after user requests. Slideshow engine subsystem works alongside the receiver subsystem to provide the user’s with the content that they requested utilizing the sender applications.
* Data Retrieval Subsystem
  + Data retrieval subsystem is the subsystem that deals with the digesting of digital media, mainly photos, and feeds into the receiver subsystem and slideshow engine subsystems in order to display the intended content. Its main function is to retrieve data from existing media online.
* Receiver Subsystem
  + The receiver subsystem is a custom chrome cast receiver application in charge of handling the sender subsystems requests, working alongside the slideshow engine subsystem, and displaying content onto chrome cast devices. The receiver subsystem receives the data input sent by the sender application and then requests data from the data retrieval subsystem.

**2.3. Hardware and Software Mapping**

* Users Hardware/Software
  + Hardware: Computer, PC, Android Device, Google Chrome cast, HDMI capable device.
    - Computer and android device allow users to run the sender applications that allow interaction with other parts of system.
    - Chrome cast - is required in order to display content onto intended device.
    - HDMI capable device - is necessary in order for the Chrome cast functionality, plays the role as the host device running the custom receiver application.
    - Chrome cast - allows hdmi capable devices of running custom receiver applications, and allowing interactions through devices such as PCs and mobile devices.
  + Software:
    - Desktop Google Chrome browser - installed alongside google chrome cast plugin. Sender application is created for
    - Android Operating System - Allows for the mobile sender application to run and allows user interaction with custom receiver application and chrome cast.

**2.4. Security/Privacy**

As the current system stands, security and privacy is not a major concern. However, through program design, the system must be able to take account for future security concerns. The data retrieval subsystem may in the future allow for user authentication from existing media requiring authentication online. In order to combat this, a modular approach is taken in the design of this component that will allow for user data to either be temporary, as well as secure from user misuse.

**3. Detailed Design**

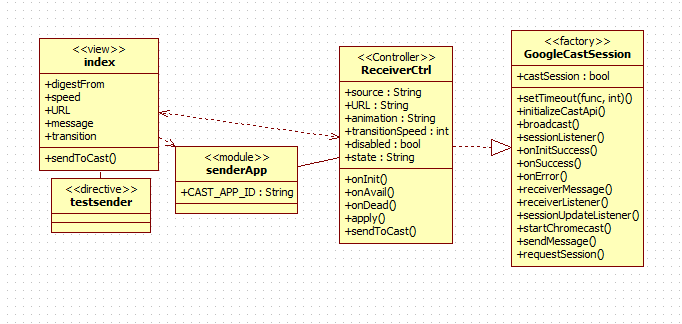
This section deals with the detailed design of our system. It includes the static and dynamic models that describe our system, as well as the documented code that it makes up.

**3.1. Overview**

The Social Wall system is comprised of four subsystems. One (Data Retrieval) that retrieves the source of the photo album, one (Sender) that sends the information the user inputs, one (Receiver) that receives and store the data that is sent, and one (Slideshow Engine) that displays the data that was retrieved.

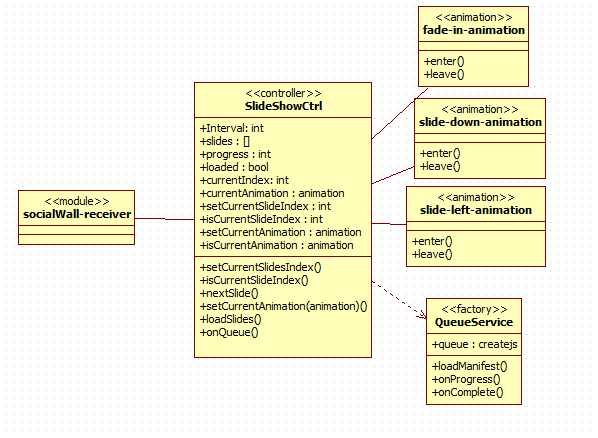
**3.2 Static Model**

As stated, the Social Wall system is composed of four subsystems.



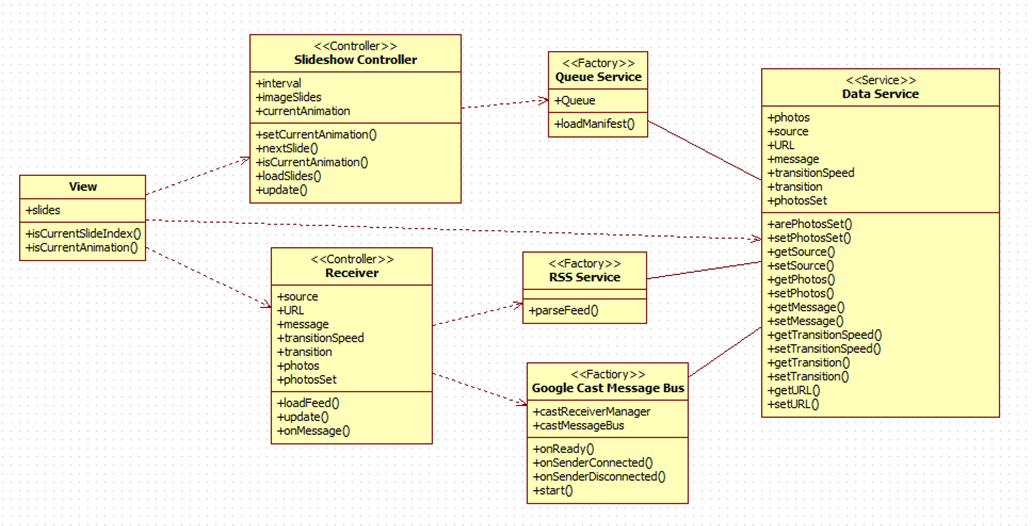
s

* **Sender Application Subsystem** 
  + HTML5-CSS
  + Framework: AngularJS
  + Architecture: MVC & Cordova(App Only)
  + The sender application subsystem is made up of an android application as well as a Chrome application. This system provides the user interface that will allow for users to input information and later retrieve the desired result. This subsystem interacts directly with the receiver subsystem and communicates messages along. These messages contain the user input data that will later be used in the other subsystems.



Slideshow Engine Class Diagram

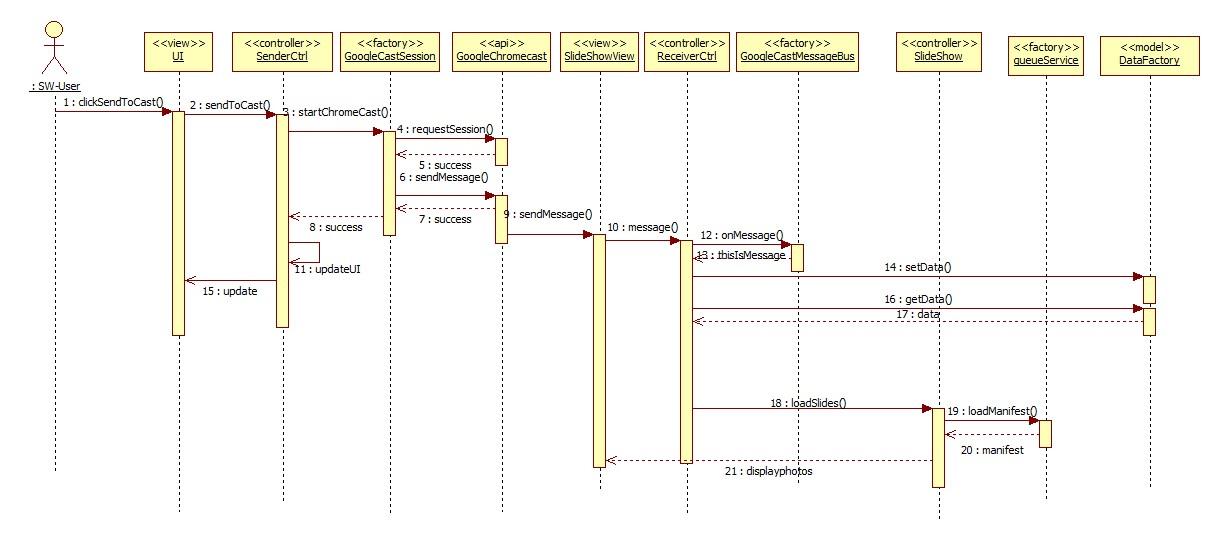
* **Slideshow Engine Subsystem** 
  + Framework: AngularJS
  + Architecture: MVC
  + Slideshow engine subsystem is in charge of the photo manipulation which is the content that will be displayed after user requests. Slideshow engine system utilizes an interface approach to work alongside other subsystems. The data retrieval subsystem feeds data directly into the data source that is utilized for the slideshow engine system, and the user input sent from the sender application is used in the generating of a slideshow that is displayed by the receiver application.
* **Data Retrieval Subsystem**
  + Framework: AngularJS
  + Architecture: MVC
  + Data retrieval subsystem is the subsystem that deals with the digesting of digital media, mainly photos, and feeds into the receiver subsystem and slideshow engine subsystems in order to display the intended content. Its main function is to retrieve data from existing media online. Depends on the sender subsystem as this subsystem retrieves data depending on the source that is decided on by the user. The receiver subsystem calls upon this subsystem and controls this subsystems actions.

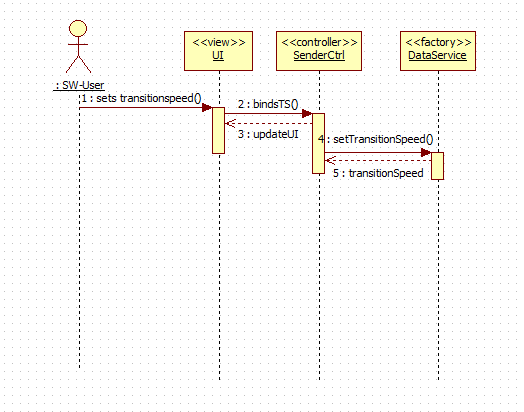


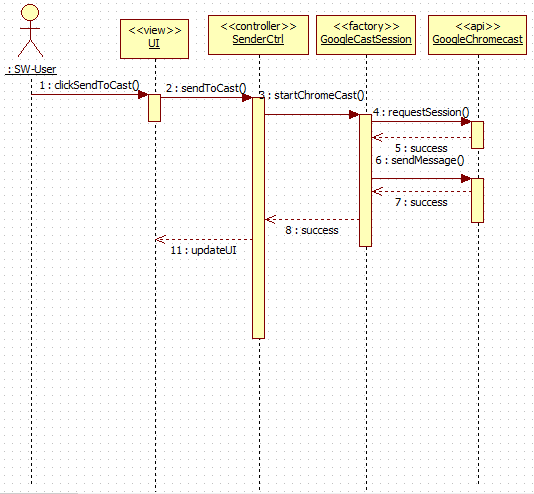
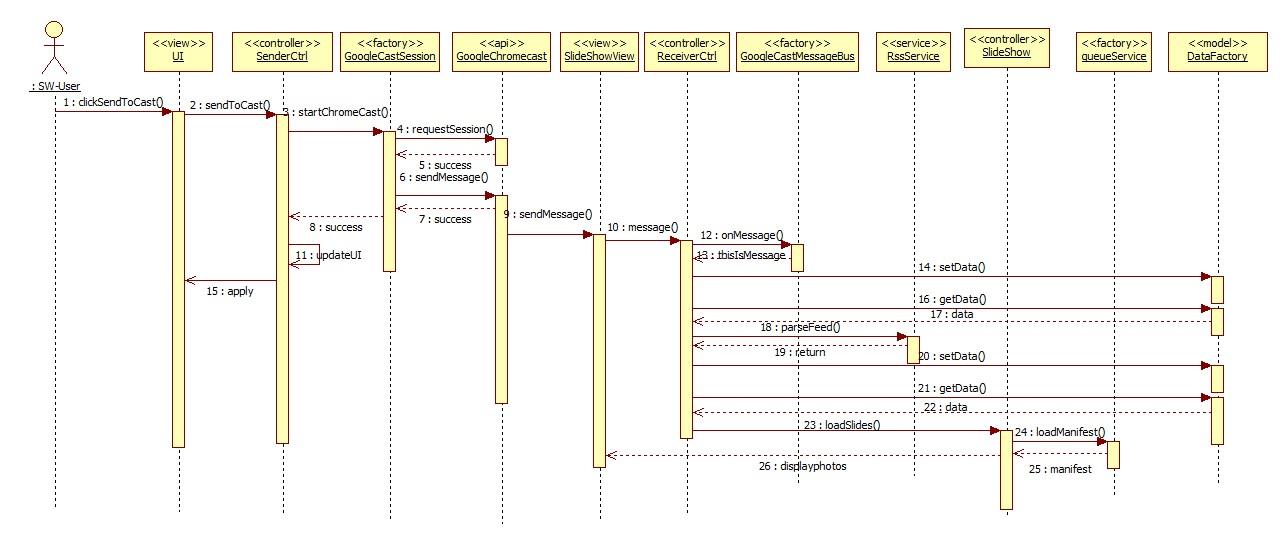
Class diagram of Custom Receiver Subsystem

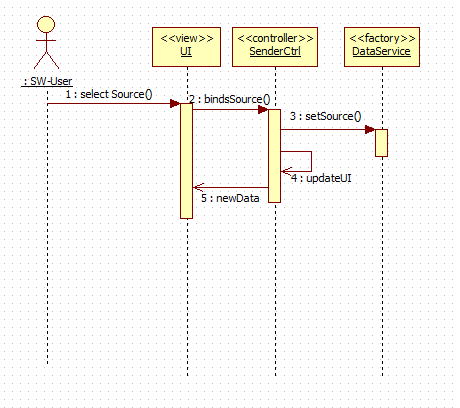
* **Receiver Subsystem**
  + Framework: AngularJS, HTML5
  + Architecture: MVC
  + The receiver subsystem is a custom chrome cast receiver application in charge of handling the sender subsystems requests, working alongside the slideshow engine subsystem, and displaying content onto chrome cast devices. The receiver subsystem receives the data input sent by the sender application and then requests data from the data retrieval subsystem. The receiver subsystem is a fully custom web application that has the capability of being deployed onto chrome cast devices. Receiver subsystem is in charge of handling the data sent by sender applications, as well as interacting with the other subsystems in order to generate the correct outcome, depending on what the user input.

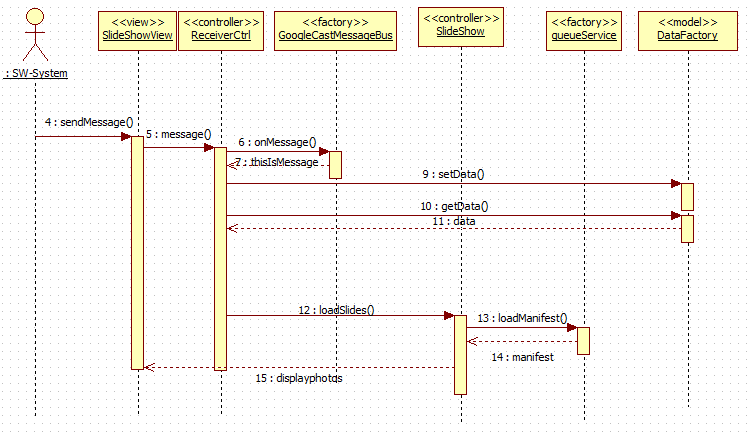
**3.3 Dynamic Model**

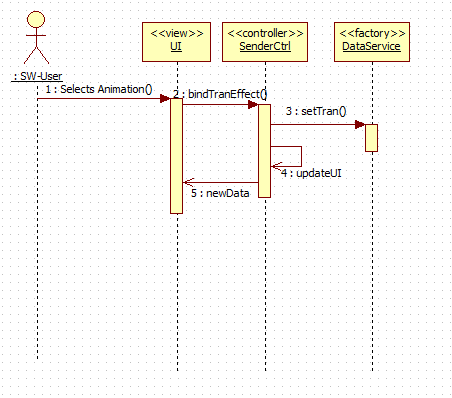
Sequence Diagram S-WALL/1-001/Generate Slideshow.

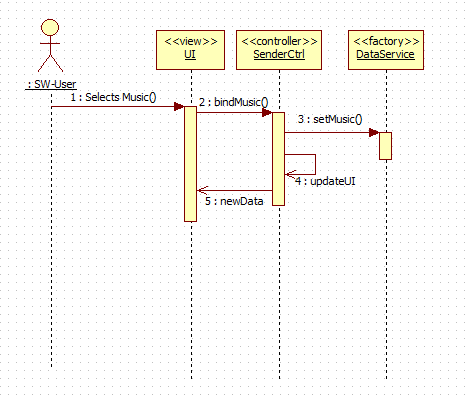
Sequence Diagram S-WALL/1-002/Image Cycle Speed

Sequence Diagram-WALL/1-008/Digest RSS Feed

Use Case ID: S-WALL/1-009/Select Photo Source

Use Case ID: S-WALL/1-011/Handle Cast Request

Use Case ID: S-WALL/1-012/Change Effect

Use Case ID: S-WALL/1-013/Add Music

**3.4 Code Specification**

//Filename: Receiver.js

//Author@JosephG

angular.module('socialWall-receiver')

.controller('ReceiverCtrl', ['$scope', '$timeout', '$http', 'GoogleCastMessageBus', 'DataService', 'rssService',

function ($scope, $timeout, $http, GoogleCastMessageBus, DataService, rssService) {

$scope.source = 'RSSFeed';

$scope.URL = "http://tinyurl.com/nynogx3";

$scope.message = '';

//On message received, perform function, e contains data.

GoogleCastMessageBus.onMessage = function (e) {}

//Load Feed

function loadFeed(){}

//Filename: sshow.js

(function () {

'use strict';

angular.module('socialWall-receiver')

.controller('SlideShowCtrl', function($scope, $timeout, $window, QueueService, DataService) {}

var IINTERVAL;

var slides;

//Sets currentslide index, tracker

function setCurrentSlideIndex(index){}

//Checks too see whether current slide is last

function isCurrentSlideIndex(index)

//Show next slide/photo

function nextSlide(){}

//Set animation from setting

function setCurentAnimation(animation){}

//if animation is current, contineu

function isCurrentAnimation(animation){}

progres;

loaded;

currentIndex;

currentAnimation;

//

//Filename: DataFactory.js

//Author: JosephG

//The model of the architecture, Contains data shared across controllers.

var photos;

var source;

var URL;

var message;

var transitionSpeed;

var transition;

var photosSet;

this.arePhotosSet = function() {}

this.setPhotosSet = function(newSet) {}

this.getSource = function() {}

this.getPhotos = function(){}

this.getMessage = function(){}

this.setPhotos = function(newPhoto){}

this.getTransitionSpeed = function(){}

this.setTransitionSpeed = function(newSpeed){}

this.getURL = function(){}

this.setURL = function(newURL){}

this.getTransition = function(){}

this.setTransition - function(newTran){}

//

//Filename: googlecastmessagebus.js

//Author: JosephG

//Handles chrome cast on network connection

angular.module('socialWall-receiver')

.factory('GoogleCastMessageBus', ['MESSAGE\_NAMESPACE', function (MESSAGE\_NAMESPACE) {

// Initialize the chromecast

cast.receiver.logger.setLevelValue(0);

var castReceiverManager ;

// Handle the 'Ready' event

castReceiverManager.onReady = function (e) {};

// Handle the 'SenderConnected' event

castReceiverManager.onSenderConnected = function (e) {};

// Handle the 'SenderDisconnected' event

castReceiverManager.onSenderDisconnected = function (e) {};

// Create the Cast Message Bus to handle messages for the custom namespace

var castMessageBus;

// Initialize the CastReceiverManager with a base application status

castReceiverManager.start(();

//Filename: rssService.js

//Author: JosephG

//Uses google api to retrieve photos

(function () {

'use strict';

angular.module('socialWall-receiver')

.factory('rssService',['$http',function($http){

return {

parseFeed : function(url){

return $http.jsonp('//ajax.googleapis.com/ajax/services/feed/load?v=1.0&num=100&callback=JSON\_CALLBACK&q=' + encodeURIComponent(url));

}

}

}]);

})();

//Filename: queueService.js

//Author: Juan G.

angular.module('socialWall-receiver')

.factory('QueueService', function($rootScope){

var queue

function loadManifest(manifest) {}

queue.on('progress', function(event) {}

queue.on('complete', function() {}

//Filename: main.js

//Author: JosephG

//Receiver Control for managing the custom receiver applicaiton and receing

//messages from the sender applications.

angular.module('socialWall-sender')

.controller('MainCtrl', ['$rootScope', '$scope', 'GoogleCastSession', function ($rootScope, $scope, GoogleCastSession) {

$scope.source

$scope.URL

$scope.animation

$scope.transitionSpeed

$scope.disabled

$scope.stat

// Listen to the different states

$rootScope.$on('INITIALIZING\_CAST\_API', function () {}

});

$rootScope.$on('RECEIVER\_AVAILABLE', function () {}

});

$rootScope.$on('RECEIVER\_DEAD', function () {}

// Send the data to the chrome cast

$scope.sendToCast = function () {}

//Filename: googlecastmessagebus.js

//Author: JosephG

//factory used by receiver application to handle requests made by chromecast

//returns a case message session

//messages from the sender applications.

angular.module('socialWall-sender')

.factory('GoogleCastSession', ['$rootScope', 'CAST\_APP\_ID', 'MESSAGE\_NAMESPACE', function ($rootScope, CAST\_APP\_ID, MESSAGE\_NAMESPACE) {

var castSession = null;

// Timeout to initialize the API

if (!chrome.cast || !chrome.cast.isAvailable) {}

// Initialize the Google Cast API for use

function initializeCastApi() {}

function sessionListener(e) {}

function onInitSuccess() {}

function onSuccess(message) {}

function onError(message) {}

function receiverMessage(namespace, message) {}

function receiverListener(e) {}

function sessionUpdateListener(isAlive) {}

return

}

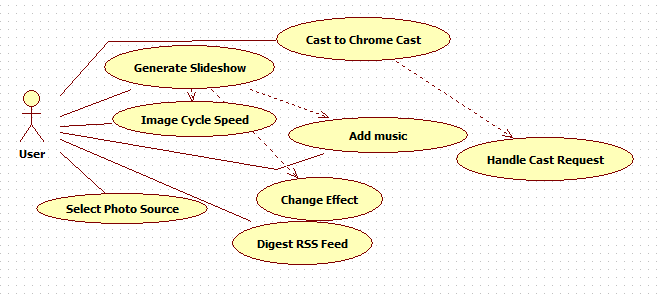
}]);

})();

**4. Glossary**

|  |  |
| --- | --- |
| Term | Definition |
| Slideshow | An array of images that are displayed one at a time. |
| Dynamic | Capable of action or change |
| Static | Stationary, fixed |
| Functional Requirements | Features that are integral to the desired output of the system |
| Non-functional Requirements | Constraints on the system that determine the quality of said system. |

**5. Appendix**

**5.1 Appendix A - Use case diagram**

Use Case Diagram displaying implemented use cases.

**5.2 Appendix B - Use cases**

**Use Case ID:** S-WALL/1-001/Generate Slideshow

**Use Case Level:** High Level

**Details:**

* **Actor:** User
* **Preconditions:**

1. User must be on web app.

2. User must have selected source of images.

3. Chrome cast must be on network.

* **Description:**

1. The use case begins when the user clicks cast to chromecast.

2. The system shall send the request to the chrome cast device.

3. The system shall display the images from source destination on chrome cast enabled device.

4. The use case ends when the slideshow engine has began showing the images.

**Postconditions:**

1. The user shall be presented with with a slideshow of images according to the settings input by the user.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to generate the slideshow.
* The system is unable to display images.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** Will be used almost everytime software is used.
* **Criticality:** High. Core functionality of software.
* **Risk:** High. Dependent upon external entities.

**Constraints:**

* System shall present user with slideshow within 5 seconds.

**Modification History:**

* **Owner:**
* **Initiation Date:** September 8, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-002/Image Cycle Speed

**Use Case Level:** High Level

**Details:**

* **Actor:** User
* **Preconditions:**

1. User must be on web application.

2. User must be in slideshow settings.

* **Description:**

1. The use case begins when the user enters data into the “Image Cycle Speed” setting.

2. The system shall maintain user input as image cycle speed.

3. The use case ends when the system records the users input and displays the changed number.

* **Postconditions:**

1. The user can purchase the item from an external site or can simply close the window.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to change image cycle speed.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** Medium- High Frequency. Setting utilized at least once every application use.
* **Criticality:** Medium-High. Most users may utilize feature, but it is not the core feature of the site.
* **Risk:** Low.

**Constraints:**

* The system must change the image cycle speed in under .2 seconds.

**Modification History:**

* **Owner:**
* **Initiation Date:** June 8, 2014
* **Last Modified:** July 9, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-008/Digest RSS Feed

**Use Case Level:** High Level

**Details:**

* **Actor:** System(Chrome cast)
* **Preconditions:**

1. System has message already set.

* **Description:**

1. The use case begins when the system sends a message to the receiver app.

2. The receiver app shall receive the message and set data.

4. The receiver app shall digest photos from the rss feed.

5. The system shall present the user with an input box.

6. User shall enter the RSS Feed information

5. User shall click on the “next” button.

6. Use case ends when user has set RSS Feed information and has proceeded.

* **Postconditions:**

1. User has set up the RSS Feed link information and system is ready to digest photos.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to access source.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** High Frequency.
* **Criticality:** High. Needed for core functionality of system.
* **Risk:** Medium.

**Constraints:**

**Modification History:**

* **Owner:**
* **Initiation Date:** September 8, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-009/Select Photo Source

**Use Case Level:** High Level

**Details:**

* **Actor:** User
* **Preconditions:**

1. User must be on web application.

* **Description:**

1. The use case begins when the user clicks “photo source” button.

2. The system shall present the user with the different source options available.

4. The user shall select an option from those listed to use as the source.

5. The system shall present the user with the function of accessing source.

6. User shall complete source process.

5. The use case ends when the system shows the user that the source option that has been chosen is active.

* **Postconditions:**

1. Source of images is that which user has chosen.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to access source.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** High Frequency.
* **Criticality:** High. Needed for core functionality of system.
* **Risk:** Medium.

**Constraints:**

**Modification History:**

* **Owner:**
* **Initiation Date:** October 5, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-010/Cast to Chrome Cast

**Use Case Level:** High Level

**Details:**

* **Actor:** User
* **Preconditions:**

1. User is utilizing the desktop application
2. User has setup source and settings.

* **Description:**

1. The use case begins when the user touches the “Send to Chromecast” icon.

2. The system shall present the user with devices found on network.

3. The user shall select the Chrome Cast device wanted.

4. The use case ends when the user has been able to select the chrome cast on the network.

* **Postconditions:**

1. User knows cast has begun.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to access source.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** High Frequency.
* **Criticality:** High. Needed for core functionality of system.
* **Risk:** Medium.

**Constraints:**

**Modification History:**

* **Owner:**
* **Initiation Date:** October 5, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-011/Handle Cast Request

**Use Case Level:** High Level

**Details:**

* **Actor:** System
* **Preconditions:**

1. User has sent a Chrome Cast request to the server.

* **Description:**

1. The use case begins when the receiver application receives a message.

2. The system shall handle corresponding request adequately.

3. The system shall then handle request, and cast the photo feed with the users settings onto intended device.

**Postconditions:**

1. Source of images is that which user has chosen.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to access source.
* The system cannot connect to web.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** High Frequency.
* **Criticality:** High. Needed for core functionality of system.
* **Risk:** Medium.

**Constraints:**

**Modification History:**

* **Owner:**
* **Initiation Date:** October 5, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-012/Change Effect

**Use Case Level:** High Level

**Details:**

* **Actor:** Server
* **Preconditions:**

1. User is on browser or mobile device.

* **Description:**

1. The use case begins when the user loads the sender application.

2. The system shall present the user with an option of changing slideshow effect.

3. The user shall select the setting of their choice.

4. Use case ends when the user can see their selected effect is set.

**Postconditions:**

1. When slideshow generation begins, user set effect is being displayed.

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable to access source.
* The system cannot connect to web.

**Related Use Cases:**

**Decision Support:**

* **Frequency:** High Frequency.
* **Criticality:** Medium. Needed for user satisfaction, expectation. Aesthetic functionality.
* **Risk:** Low.

**Constraints:**

**Modification History:**

* **Owner:**
* **Initiation Date:** October 5, 2014
* **Last Modified:** October 5, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Use Case ID:** S-WALL/1-013/Add Music

**Use Case Level:** High Level

**Details:**

* **Actor:** User
* **Preconditions:**

1. User must be on web application.

2. User must be in slideshow settings.

* **Description:**

1. The use case begins when the user enters a YouTube URL in the “Add Background Music” field.

2. The system records the users input and displays the URL.

3. The use case ends when the user clicks “Generate Slideshow”

* **Postconditions:**

1. System shall play the youtube audio during the slideshow

**Alternative Courses of Action**: N/A

**Exceptions:**

* The system is unable retrieve YouTube video
* YouTube video does not allow embedding
* System cannot connect to web

**Related Use Cases:** SW/1-001/Generate Slideshow

**Decision Support:**

* **Frequency:** Medium- High Frequency. Setting utilized at least once every application use.
* **Criticality:** Low. Most users may utilize feature, but it is not the core feature of the app.
* **Risk:** Low.

**Constraints:**

* The system must play audio in under .2 seconds.

**Modification History:**

* **Owner:**
* **Initiation Date:** October 17, 2014
* **Last Modified:** October 24, 2014

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**5.3 Appendix C - Class interfaces**

//Filename: Receiver.js

//Author@JosephG

angular.module('socialWall-receiver')

.controller('ReceiverCtrl', ['$scope', '$timeout', '$http', 'GoogleCastMessageBus', 'DataService', 'rssService',

function ($scope, $timeout, $http, GoogleCastMessageBus, DataService, rssService) {

$scope.source = 'RSSFeed';

$scope.URL = "http://tinyurl.com/nynogx3";

$scope.message = '';

//On message received, perform function, e contains data.

GoogleCastMessageBus.onMessage = function (e) {}

//Load Feed

function loadFeed(){}

//Filename: sshow.js

(function () {

'use strict';

angular.module('socialWall-receiver')

.controller('SlideShowCtrl', function($scope, $timeout, $window, QueueService, DataService) {}

var IINTERVAL;

var slides;

//Sets currentslide index, tracker

function setCurrentSlideIndex(index){}

//Checks too see whether current slide is last

function isCurrentSlideIndex(index)

//Show next slide/photo

function nextSlide(){}

//Set animation from setting

function setCurentAnimation(animation){}

//if animation is current, contineu

function isCurrentAnimation(animation){}

progres;

loaded;

currentIndex;

currentAnimation;

//

//Filename: DataFactory.js

//Author: JosephG

//The model of the architecture, Contains data shared across controllers.

var photos;

var source;

var URL;

var message;

var transitionSpeed;

var transition;

var photosSet;

this.arePhotosSet = function() {}

this.setPhotosSet = function(newSet) {}

this.getSource = function() {}

this.getPhotos = function(){}

this.getMessage = function(){}

this.setPhotos = function(newPhoto){}

this.getTransitionSpeed = function(){}

this.setTransitionSpeed = function(newSpeed){}

this.getURL = function(){}

this.setURL = function(newURL){}

this.getTransition = function(){}

this.setTransition - function(newTran){}

//

//Filename: googlecastmessagebus.js

//Author: JosephG

//Handles chrome cast on network connection

angular.module('socialWall-receiver')

.factory('GoogleCastMessageBus', ['MESSAGE\_NAMESPACE', function (MESSAGE\_NAMESPACE) {

// Initialize the chromecast

cast.receiver.logger.setLevelValue(0);

var castReceiverManager ;

// Handle the 'Ready' event

castReceiverManager.onReady = function (e) {};

// Handle the 'SenderConnected' event

castReceiverManager.onSenderConnected = function (e) {};

// Handle the 'SenderDisconnected' event

castReceiverManager.onSenderDisconnected = function (e) {};

// Create the Cast Message Bus to handle messages for the custom namespace

var castMessageBus;

// Initialize the CastReceiverManager with a base application status

castReceiverManager.start(();

//Filename: rssService.js

//Author: JosephG

//Uses google api to retrieve photos

(function () {

'use strict';

angular.module('socialWall-receiver')

.factory('rssService',['$http',function($http){

return {

parseFeed : function(url){

return $http.jsonp('//ajax.googleapis.com/ajax/services/feed/load?v=1.0&num=100&callback=JSON\_CALLBACK&q=' + encodeURIComponent(url));

}

}

}]);

})();

//Filename: queueService.js

//Author: Juan G.

angular.module('socialWall-receiver')

.factory('QueueService', function($rootScope){

var queue

function loadManifest(manifest) {}

queue.on('progress', function(event) {}

queue.on('complete', function() {}

//Filename: main.js

//Author: JosephG

//Receiver Control for managing the custom receiver applicaiton and receing

//messages from the sender applications.

angular.module('socialWall-sender')

.controller('MainCtrl', ['$rootScope', '$scope', 'GoogleCastSession', function ($rootScope, $scope, GoogleCastSession) {

$scope.source

$scope.URL

$scope.animation

$scope.transitionSpeed

$scope.disabled

$scope.stat

// Listen to the different states

$rootScope.$on('INITIALIZING\_CAST\_API', function () {}

});

$rootScope.$on('RECEIVER\_AVAILABLE', function () {}

});

$rootScope.$on('RECEIVER\_DEAD', function () {}

// Send the data to the chrome cast

$scope.sendToCast = function () {}

//Filename: googlecastmessagebus.js

//Author: JosephG

//factory used by receiver application to handle requests made by chromecast

//returns a case message session

//messages from the sender applications.

angular.module('socialWall-sender')

.factory('GoogleCastSession', ['$rootScope', 'CAST\_APP\_ID', 'MESSAGE\_NAMESPACE', function ($rootScope, CAST\_APP\_ID, MESSAGE\_NAMESPACE) {

var castSession = null;

// Timeout to initialize the API

if (!chrome.cast || !chrome.cast.isAvailable) {}

// Initialize the Google Cast API for use

function initializeCastApi() {}

function sessionListener(e) {}

function onInitSuccess() {}

function onSuccess(message) {}

function onError(message) {}

function receiverMessage(namespace, message) {}

function receiverListener(e) {}

function sessionUpdateListener(isAlive) {}

return

}

}]);

})();

**5.4 Appendix D - Diary of meeting and tasks.**

Diary Entry 1:

Date: September 4, 2014

Location: Picture Marketing Offices

Start time: 2:30 pm

End time: 5:00 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills

Late: N/A

Agenda:

* Introduction and initial meeting
* Learn about Picture Marketing
* Explanation and details of the project
* Go over requirements of the system
* Talk about expectations of the project

Summary of Discussion:

Introduced ourselves to Cortney and the Picture Management. Discussed the Social Wall project, what is needed and what is expected.

Diary Entry 2:

Date: September 8, 2014

Location: Virtual meeting via Skype

Start time: 7:30 pm

End time: 8:40 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez

Late: N/A

Agenda:

* Talk about document drafts due
* Discussed use cases
* Come up with requirements and constraints

Summary of Discussion:

Discussed and worked on drafts that are due on September 8th and we came up with the use cases.

Diary Entry 3:

Date: September 19, 2014

Location: Picture Marketing Offices

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Discuss Development Approach
* Create Schedule
* Discuss requirements/finalize

Summary of Discussion:

Louis would become a mentor as well of the project. Had meeting to discuss how to approach the project and finalize requirement elicitation. Decided upon an android application, chrome cast compatible, as well as a web application. Discussed tools needed as well as framework that will be utilized being angularjs. Joseph would be in charge of android application, Juan in charge of Slide show engine and web app.

Diary Entry 4:

Date: September 26, 2014

Location: Picture Marketing Offices

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Show Facebook demo
* Discuss development approach

Summary of Discussion:

Discussion and acceptance of major work must be done on core parts of system. Must get android application running phonegap and slideshow engine must begin development.

Diary Entry 5:

Date: October 3, 2014

Location: Picture Marketing Offices

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Discuss phonegap project
* Slideshow engine

Summary of Discussion:

Phonegap project discussed in detail. Along with getting application running on emulator. See benefits of angular and begin approach to develop using framework. Slideshow engine displayed, must be converted to angularjs.

Diary Entry 6:

Date: October 22, 2014

Location: FIU ECS Computer lab

Start time: 1:50 pm

End time: 3:00 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez

Late: N/A

Agenda:

* Discuss progress
* Slideshow engine demo

Summary of Discussion:

We spoke about our recent progress and breakthroughs. Juan spoke of his improvements to the slideshow engine, and Joseph spoke about his improvement of the mobile app.

Diary Entry 7:

Date: October 28, 2014

Location: Picture Marketing Offices

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Discuss phonegap project
* Slideshow engine

Summary of Discussion:

Phonegap project discussed in detail. Along with getting application running on emulator. See benefits of angular and begin approach to develop using framework. Slideshow engine displayed, must be converted to angularjs.

Diary Entry 8:

Date: November 2, 2014

Location: FIU SCS Computer Lab

Start time: 2:00 pm

End time: 4:00 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez

Late: N/A

Agenda:

* Discuss progress
* Develop UML diagrams

Summary of Discussion:

We discussed our work up until that point in time, displaying our progress in our respective portion of the project. Afterwards we began developing UML diagrams, mostly sequence diagrams, based on our progress so far.

Diary Entry 9:

Date: November 11, 2014

Location: Picture Marketing

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Discuss progress
* Discuss Chrome Cast
* Discuss Slideshow engine

Summary of Discussion:

Chrome Cast was the main focus of this meeting. Picture Marketing lent us their Chrome Cast so that we could begin developing the app so it could utilize it. Afterwards, we discussed the slideshow engine, and getting it working using the AngularJS framework.

Diary Entry 10:

Date: November 24, 2014

Location: FIU SCS Computer Lab

Start time: 1:50 pm

End time: 2:50 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez

Late: N/A

Agenda:

* Discuss progress
* Update documentation
* Discuss direction of project

Summary of Discussion:

Like all our meetings, we discussed how far we’ve done with our respective portions of the project. We updated our documentation accordingly. Once completed, we discussed the direction of our portions so we could schedule when we would be able merge our sections to finish the Social Wall app.

Diary Entry 11:

Date: November 29, 2014

Location: Skype (virtual meeting)

Start time: 5:30 pm

End time: 8:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph GonzalezLate: N/A

Agenda:

* Assemble final Social Wall app

Summary of Discussion:

We used this meeting to assemble both portions of the Social Wall app. We were able to get our application functioning as expected, with only minor bugs.

Diary Entry 12:

Date: December 4, 2014

Location: Picture Marketing

Start time: 2:30 pm

End time: 4:30 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Display our full demo
* Discuss current bugs
* Discuss shortcomings of the project

Summary of Discussion:

At this meeting, we were able to display our full demo to our mentors at Picture Marketing. After the demo, we highlighted our bugs, as well as the features we weren’t able to accomplish due to time restraints.

Diary Entry 13:

Date: December 10, 2014

Location: Picture Marketing

Start time: 1:30 pm

End time: 6:00 pm

In Attendance: Juan Gonzalez-Llanos, Joseph Gonzalez, Cortney Mills, Louis Zuckerman

Late: N/A

Agenda:

* Display demo with bug fixes
* Record videos
* Finish documentation

Summary of Discussion:

For this meeting, we put the final touches on the Social Wall project. We had already cleaned up most of the bugs from the past week, and showed our mentors a more complete Social Wall application. We used the remaining time to record our instruction videos and to finish documentation.

**6. References**

1. <http://angularjs.org/> - AngularJS Framework for Javascript
2. <http://ionicframework.com/> - Ionic Framework for developing HTML mobile apps
3. <http://phonegap.com/> - Porting web app to Android
4. <http://cordova.apache.org/> - Porting web app to Android
5. <http://nodejs.org/> - Porting web app to Android
6. <http://getbootstrap.com/> - Bootstrap Framework
7. <http://bower.io/> - Package manager for web app
8. <http://www.techsmith.com/camtasia.html/> - Screen Recorder for videos