EECS3215 – Smart Mirror Project

April, 19, 2016

Name	Student Number	Email	Prism Login
Kevin Arindaeng	213094016	azkevin@my.yorku.ca	azkevin
Kajanan Kanathipan	212846853	kana0603@my.yorku.ca	kana0603

The work in this report is our own. We have read and understood York University academic dishonesty policy and I did not violate the senate dishonesty policy in writing this report.

Signed,

[illegible]

Introduction

In today's society new innovative technologies are constantly being developed to enhance the quality of life for everyday people. A regular mirror is a surface capable of reflecting a clear image back at the viewer. A Smart Mirror can display much more than a typical mirror such as the current time, weather, and a news feed. By attaching a two-way mirror to a LCD monitor it would be possible to have the information displayed on the monitor bleed through the reflected image. In order to achieve this an efficient embedded system must be designed. An embedded system is a special purpose system which is designed to perform pre-defined tasks. Most embedded systems are typically based around a microcontroller. The microcontroller used in this project is Raspberry Pi 2 model B. The other components of the embedded system are a two-way acrylic mirror, a 12000mAH portable battery pack, and a flat panel LCD monitor. The two-way mirror was cut to precisely fit the LCD monitor while the microcontroller was connected to the LCD monitor through a HDMI cable. The microcontroller was then programmed to obtain information from the internet through wifi to be displayed on the LCD monitor. The battery pack was used to provide power to the embedded system without the use of an outlet to make the design portable. The completed Smart Mirror could be mounted onto a wall or placed on a stand.

Hardware and Schematic Explanation

Pictures of all components used in this project are located in the appendix. Figures 1-5 show each individual part while Figures 6-11 show the construction of the Smart Mirror. Figure 6 shows 10 scotch tape Velcro straps connected to the monitor. The straps were cut and placed on the monitor such that it would be able to securely hold the two-way mirror without the risk of it falling. Figure 7 shows the two-way mirror with its covering removed. From this figure it can be seen that the mirror reflects what is in front of it as well as displays what is behind it. Figure 8 shows the mirror connected to the monitor. As the monitor has not been turned on at this time, the mirror is only reflecting what is in front of it. Figure 9 shows the monitor displaying text as well as a keyboard and mouse being used to set up the text that should be displayed. From this figure it can be seen that the mirror is reflecting what is in front of it as well as allowing the text displayed on the monitor to bleed through the reflected image. Figure 10 shows the completed Smart Mirror. The microcontroller has been setup to obtain information from the internet through wifi and which is then displayed on the monitor. The two-way mirror has been attached to the monitor and is reflecting the viewer's image while allowing the text displayed on the monitor to bleed through the mirror. Figure 11 shows the back of the Smart Mirror which contains the battery pack and the microcontroller. Both the battery pack and the microcontroller have been connected to the back of the Smart Mirror with the use of duct tape in such a way that they would protrude from the sides of the Smart Mirror. In an actual product this would be covered with a frame or a case.

Conclusion

From our tests it can be seen that the constructed Smart Mirror functioned as expected. The microcontroller was able to display the required information onto the monitor. The scotch tape Velcro strap was able to hold the two way mirror in place without the risk of it falling. The two way mirror was able to show the reflection of the viewer as well as what was being displayed on the monitor. The smart mirror was able to clearly display various things such as the time, date, weather, news feed, and a calendar while reflecting the viewer’s image back at them. The designed Smart Mirror was seen to be compatible with a stand as well as able to be mounted on a wall. In conclusion, an efficient embedded system has been designed and built which is capable of providing information to everyday people in a smart and effective manner.

Software Code

```
var calendar = {
	eventList: [],
	calendarLocation: '.calendar',
	updateInterval: 1000,
	updateDataInterval: 60000,
	fadeInterval: 1000,
	intervalId: null,
	dataIntervalId: null,
	maximumEntries: config.calendar.maximumEntries || 10,
	calendarUrl: (typeof config.calendar.urls == 'undefined') ?
config.calendar.url : config.calendar.urls[0].url,
	calendarPos: 0,
	defaultSymbol: config.calendar.defaultSymbol || 'none',
	calendarSymbol: (typeof config.calendar.urls == 'undefined') ?
config.calendar.defaultSymbol || 'none' : config.calendar.urls[0].symbol,
	displaySymbol: (typeof config.calendar.displaySymbol == 'undefined') ? false :
config.calendar.displaySymbol,
	shortRunningText: 'still',
	longRunningText: 'until',
}

calendar.processEvents = function (url, events) {
	tmpEventList = [];
	var eventListLength = this.eventList.length;
	for (var i = 0; i < eventListLength; i++) {
		if (this.eventList[i]['url'] != url) {
			tmpEventList.push(this.eventList[i]);
		}
	}
	this.eventList = tmpEventList;

	for (var i in events) {

		var e = events[i];
		for (var key in e) {
			var value = e[key];
			var seperator = key.search(';');
			if (seperator >= 0) {
				var mainKey = key.substring(0,seperator);
				var subKey = key.substring(seperator+1);

				var dt;
				if (subKey == 'VALUE=DATE') {
					//date
					dt = new Date(value.substring(0,4),
value.substring(4,6) - 1, value.substring(6,8));
				} else {
					//time
					dt = new Date(value.substring(0,4),
value.substring(4,6) - 1, value.substring(6,8), value.substring(9,11),
value.substring(11,13), value.substring(13,15));
				}

				if (mainKey == 'DTSTART') e.startDate = dt;
				if (mainKey == 'DTEND') e.endDate = dt;
```

```
                }
            }

            if (e.startDate == undefined){
                //some old events in Gmail Calendar is "start_date"
                //FIXME: problems with Gmail's TimeZone
                var days = moment(e.DTSTART).diff(moment(), 'days');
                var seconds = moment(e.DTSTART).diff(moment(), 'seconds');
                var startDate = moment(e.DTSTART);
                var endDays = moment(e.DTEND).diff(moment(), 'days');
                var endSeconds = moment(e.DTEND).diff(moment(), 'seconds');
                var endDate = moment(e.DTEND);
            } else {
                var days = moment(e.startDate).diff(moment(), 'days');
                var seconds = moment(e.startDate).diff(moment(), 'seconds');
                var startDate = moment(e.startDate);
                var endDays = moment(e.endDate).diff(moment(), 'days');
                var endSeconds = moment(e.endDate).diff(moment(), 'seconds');
                var endDate = moment(e.endDate);
            }

            //only add fututre events, days doesn't work, we need to check seconds
            if (seconds >= 0) {
                if (seconds <= 60*60*5 || seconds >= 60*60*24*2) {
                    var time_string = moment(startDate).fromNow();
                }else {
                    var time_string = moment(startDate).calendar()
                }
                if (!e.RRULE) {

        this.eventList.push({'description':e.SUMMARY,'seconds':seconds,'days':time_stri
ng,'url': url, symbol: this.calendarSymbol});
                }
                e.seconds = seconds;
            } else if  (endSeconds > 0) {
                // TODO: Replace with better lang handling
                if (endSeconds <= 60*60*5 || endSeconds >= 60*60*24*2) {
                    var time_string = this.shortRunningText + ' ' +
moment(endDate).fromNow(true);
                }else {
                    var time_string = this.longRunningText + ' ' +
moment(endDate).calendar()
                }
                if (!e.RRULE) {

        this.eventList.push({'description':e.SUMMARY,'seconds':seconds,'days':time_stri
ng,'url': url, symbol: this.calendarSymbol});
                }
                e.seconds = endSeconds;
            }
            // Special handling for rrule events
            if (e.RRULE) {
                var options = new RRule.parseString(e.RRULE);
                options.dtstart = e.startDate;
                var rule = new RRule(options);

                var oneYear = new Date();
                oneYear.setFullYear(oneYear.getFullYear() + 1);

                var dates = rule.between(new Date(), oneYear, true, function
(date, i){return i < 10});
                for (date in dates) {
                    var dt = new Date(dates[date]);
```

```
                        var days = moment(dt).diff(moment(), 'days');
                        var secds = moment(dt).diff(moment(), 'seconds');
                        var startDate = moment(dt);
                        if (seconds >= 0) {
                              if (seconds <= 60*60*5 || seconds >= 60*60*24*2) {
                                    var time_string = moment(dt).fromNow();
                              } else {
                                    var time_string = moment(dt).calendar()
                              }

      this.eventList.push({'description':e.SUMMARY,'seconds':seconds,'days':time_stri
ng,'url': url, symbol: this.calendarSymbol});
                        }
                  }
            }
      };

      this.eventList = this.eventList.sort(function(a,b){return a.seconds-
b.seconds});

      // Limit the number of entries.
      this.eventList = this.eventList.slice(0, calendar.maximumEntries);
}

calendar.updateData = function (callback) {
      new ical_parser("controllers/calendar.php" +
"?url="+encodeURIComponent(this.calendarUrl), function(cal) {
            this.processEvents(this.calendarUrl, cal.getEvents());

            this.calendarPos++;
            if ((typeof config.calendar.urls == 'undefined') || (this.calendarPos >=
config.calendar.urls.length)) {
                  this.calendarPos = 0;
                  // Last Calendar in List is updated, run Callback (i.e.
updateScreen)
                  if (callback !== undefined &&
Object.prototype.toString.call(callback) === '[object Function]') {
                        callback(this.eventList);
                  }
            } else {
                  // Loading all Calendars in parallel does not work, load them one
by one.
                  setTimeout(function () {
                        this.updateData(this.updateCalendar.bind(this));
                  }.bind(this), 10);
            }
            if (typeof config.calendar.urls != 'undefined') {
                  this.calendarUrl = config.calendar.urls[this.calendarPos].url;
                  this.calendarSymbol =
config.calendar.urls[this.calendarPos].symbol || this.defaultSymbol;
            }

      }.bind(this));

}

calendar.updateCalendar = function (eventList) {
      var _is_new = true;
      if ($('.calendar-table').length) {
            _is_new = false;
      }
      table = $('<table/>').addClass('xsmall').addClass('calendar-table');
      opacity = 1;
```

```
	for (var i in eventList) {
		var e = eventList[i];
		var row = $('<tr/>').attr('id',
'event'+i).css('opacity',opacity).addClass('event');
		if (this.displaySymbol) {
			row.append($('<td/>').addClass('fa').addClass('fa-
'+e.symbol).addClass('calendar-icon'));
		}
		row.append($('<td/>').html(e.description).addClass('description'));
		row.append($('<td/>').html(e.days).addClass('days dimmed'));
		if (! _is_new && $('#event'+i).length) {
			$('#event'+i).updateWithText(row.children(), this.fadeInterval);
		} else {
			// Something wrong - replace whole table
			_is_new = true;
		}
		table.append(row);

		opacity -= 1 / eventList.length;
	}
	if (_is_new) {
		$(this.calendarLocation).updateWithText(table, this.fadeInterval);
	}

}

calendar.init = function () {

	this.updateData(this.updateCalendar.bind(this));

	// this.intervalId = setInterval(function () {
		// this.updateCalendar(this.eventList)
	// }.bind(this), this.updateInterval);

	this.dataIntervalId = setInterval(function () {
		this.updateData(this.updateCalendar.bind(this));
	}.bind(this), this.updateDataInterval);

}

var compliments = {
	complimentLocation: '.compliment',
	currentCompliment: '',
	complimentList: {
		'morning': config.compliments.morning,
		'afternoon': config.compliments.afternoon,
		'evening': config.compliments.evening
	},
	updateInterval: config.compliments.interval || 30000,
	fadeInterval: config.compliments.fadeInterval || 4000,
	intervalId: null
};

/**
 * Changes the compliment visible on the screen
 */
compliments.updateCompliment = function () {



	var _list = [];
```

```
        var hour = moment().hour();

        // In the following if statement we use .slice() on the
        // compliments array to make a copy by value.
        // This way the original array of compliments stays intact.

        if (hour >= 3 && hour < 12) {
                // Morning compliments
                _list = compliments.complimentList['morning'].slice();
        } else if (hour >= 12 && hour < 17) {
                // Afternoon compliments
                _list = compliments.complimentList['afternoon'].slice();
        } else if (hour >= 17 || hour < 3) {
                // Evening compliments
                _list = compliments.complimentList['evening'].slice();
        } else {
                // Edge case in case something weird happens
                // This will select a compliment from all times of day
                Object.keys(compliments.complimentList).forEach(function (_curr) {
                        _list = _list.concat(compliments.complimentList[_curr]).slice();
                });
        }

        // Search for the location of the current compliment in the list
        var _spliceIndex = _list.indexOf(compliments.currentCompliment);

        // If it exists, remove it so we don't see it again
        if (_spliceIndex !== -1) {
                _list.splice(_spliceIndex, 1);
        }

        // Randomly select a location
        var _randomIndex = Math.floor(Math.random() * _list.length);
        compliments.currentCompliment = _list[_randomIndex];

        $('.compliment').updateWithText(compliments.currentCompliment,
compliments.fadeInterval);

}

compliments.init = function () {

        this.updateCompliment();

        this.intervalId = setInterval(function () {
                this.updateCompliment();
        }.bind(this), this.updateInterval)

}


var config = {
    lang: 'en',
    time: {
        timeFormat: 12,
        displaySeconds: true,
        digitFade: false,
    },
    weather: {
        //change weather params here:
        //units: metric or imperial
        params: {
            q: 'Toronto,Canada',
            units: 'metric',
            // if you want a different lang for the weather that what is set above,
change it here
            lang: 'en',
            APPID: '5f6e05a1dc3b4f8308dbb67233ac96a5'
        }
    },
    compliments: {
        interval: 30000,
        fadeInterval: 4000,
        morning: [
            'Good morning!',
            'Have a good day!',
             'Good luck on your project!'
        ],
        afternoon: [
            'Good afternoon.',
            'Have a nice day!',
            'Good luck on your project!'
        ],
        evening: [
            'Good evening.',
            'Sleep well!',
            'Good luck on your project!'
        ]
    },
    calendar: {
        maximumEntries: 10, // Total Maximum Entries
             displaySymbol: true,
             defaultSymbol: 'calendar', // Fontawsome Symbol see
http://fontawesome.io/cheatsheet/
        urls: [
             {
                    symbol: 'calendar-plus-o',
                    url:
'https://calendar.google.com/calendar/ical/oiohjt6dkr5gb5o6b0debbno78%40group.calendar
.google.com/public/basic.ics'
             },
             {
                    symbol: 'soccer-ball-o',
                    url:
'https://calendar.google.com/calendar/ical/karindaeng%40gmail.com/public/basic.ics',
             },
             // {
                    // symbol: 'mars',
                    // url: "https://server/url/to/his.ics",
             // },
             // {
                    // symbol: 'venus',
                    // url: "https://server/url/to/hers.ics",
             // },
             // {
                    // symbol: 'venus-mars',
                    // url: "https://server/url/to/theirs.ics",
             // },
             ]
    },
    news: {
        feed: 'http://rss.cbc.ca/lineup/topstories.xml'
    }
}
```

```
// A lot of this code is from the original feedToJson function that was included with
this project
// The new code allows for multiple feeds to be used but a bunch of variables and such
have literally been copied and pasted into this code and some help from here:
http://jsfiddle.net/BDK46/
// The original version can be found here: http://airshp.com/2011/jquery-plugin-feed-
to-json/
var news = {
	feed: config.news.feed || null,
	newsLocation: '.news',
	newsItems: [],
	seenNewsItem: [],
	_yqURL: 'https://query.yahooapis.com/v1/public/yql',
	_yqlQS: '?format=json&q=select%20*%20from%20rss%20where%20url%3D',
	_cacheBuster: Math.floor((new Date().getTime()) / 1200 / 1000),
	_failedAttempts: 0,
	fetchInterval: config.news.fetchInterval || 60000,
	updateInterval: config.news.interval || 5500,
	fadeInterval: 2000,
	intervalId: null,
	fetchNewsIntervalId: null
}

/**
 * Creates the query string that will be used to grab a converted RSS feed into a JSON
object via Yahoo
 * @param  {string} feed The original location of the RSS feed
 * @return {string}      The new location of the RSS feed provided by Yahoo
 */
news.buildQueryString = function (feed) {

	return this._yqURL + this._yqlQS + '\'' + encodeURIComponent(feed) + '\'';

}

/**
 * Fetches the news for each feed provided in the config file
 */
news.fetchNews = function () {

	// Reset the news feed
	this.newsItems = [];

	this.feed.forEach(function (_curr) {

		var _yqUrlString = this.buildQueryString(_curr);
		this.fetchFeed(_yqUrlString);

	}.bind(this));

}

/**
 * Runs a GET request to Yahoo's service
 * @param  {string} yqUrl The URL being used to grab the RSS feed (in JSON format)
 */
news.fetchFeed = function (yqUrl) {

	$.ajax({
		type: 'GET',
		datatype:'jsonp',
		url: yqUrl,
		success: function (data) {
```

```
                    if (data.query.count > 0) {
                            this.parseFeed(data.query.results.item);
                    } else {
                            console.error('No feed results for: ' + yqUrl);
                    }

            }.bind(this),
            error: function () {
                    // non-specific error message that should be updated
                    console.error('No feed results for: ' + yqUrl);
            }
    });

}

/**
 * Parses each item in a single news feed
 * @param  {Object} data The news feed that was returned by Yahoo
 * @return {boolean}      Confirms that the feed was parsed correctly
 */
news.parseFeed = function (data) {

    var _rssItems = [];

    for (var i = 0, count = data.length; i < count; i++) {

            _rssItems.push(data[i].title);

    }

    this.newsItems = this.newsItems.concat(_rssItems);

    return true;

}

/**
 * Loops through each available and unseen news feed after it has been retrieved from
Yahoo and shows it on the screen
 * When all news titles have been exhausted, the list resets and randomly chooses from
the original set of items
 * @return {boolean} Confirms that there is a list of news items to loop through and
that one has been shown on the screen
 */
news.showNews = function () {

    // If all items have been seen, swap seen to unseen
    if (this.newsItems.length === 0 && this.seenNewsItem.length !== 0) {

            if (this._failedAttempts === 20) {
                    console.error('Failed to show a news story 20 times, stopping any
attempts');
                    return false;
            }

            this._failedAttempts++;

            setTimeout(function () {
                    this.showNews();
            }.bind(this), 3000);

    } else if (this.newsItems.length === 0 && this.seenNewsItem.length !== 0) {
```

```
			this.newsItems = this.seenNewsItem.splice(0);
		}

		var _location = Math.floor(Math.random() * this.newsItems.length);

		var _item = news.newsItems.splice(_location, 1)[0];

		this.seenNewsItem.push(_item);

		$(this.newsLocation).updateWithText(_item, this.fadeInterval);

		return true;

}

news.init = function () {

		if (this.feed === null || (this.feed instanceof Array === false && typeof 
this.feed !== 'string')) {
			return false;
		} else if (typeof this.feed === 'string') {
			this.feed = [this.feed];
		}

		this.fetchNews();
		this.showNews();

		this.fetchNewsIntervalId = setInterval(function () {
			this.fetchNews()
		}.bind(this), this.fetchInterval)

		this.intervalId = setInterval(function () {
			this.showNews();
		}.bind(this), this.updateInterval);

}
```

```
var time = {
		timeFormat: config.time.timeFormat || 24,
		dateLocation: '.date',
		timeLocation: '#time',
		updateInterval: 1000,
		intervalId: undefined,
		displaySeconds: (typeof config.time.displaySeconds == 'undefined') ? true : 
config.time.displaySeconds,
		digitFade: (typeof config.time.digitFade == 'undefined') ? false : 
config.time.digitFade,
};

/**
 * Updates the time that is shown on the screen
 */
time.updateTime = function () {
	var timeLocation = this.timeLocation;
	var _now = moment();
	var _date = _now.format('[<span class="dayname">]dddd,[</span> <span 
class="longdate">]LL[</span>]');

	$(this.dateLocation).updateWithText(_date, 1000);
	$('.fade').removeClass('fade')
	var html = ''
	if (this.displaySeconds) {
		html = _now.format(this._timeFormat+':mm').replace(/./g, '<span 
class="digit">$&</span>') +
			'<span class="sec">' + _now.format('ss').replace(/./g, '<span 
class="digit">$&</span>') + '</span>';
		if (typeof this.intervalId == 'undefined') {
			this.intervalId = setInterval(function () {
				this.updateTime();
			}.bind(this), this.updateInterval);
		}
	} else {
		html = _now.format(this._timeFormat+':mm').replace(/./g, '<span 
class="digit">$&</span>');
		if (this.intervalId) {
			clearInterval(this.intervalId);
			this.intervalId = undefined;
		}
		seconds = 60 - (new Date()).getSeconds();
		setTimeout(function () {
			this.updateTime();
		}.bind(this), seconds*1000);
	}
	if (this.digitFade) {
		var diff = $('<div>').html(html);
		diff.find('.digit').each(function( index ) {
			var _text  = $( this ).text();
			var _i = index+1;
			var liveNode = $(timeLocation).find('.digit')[index]
			if (typeof liveNode != 'undefined') {
				liveNode = $(liveNode);
				var _text2 = liveNode.text();
				if (_text != _text2) {

					liveNode.addClass('fade');
					$(this).addClass('fade');
				}
			} else {
				$(this).addClass('fade');
			}
		});
		if ($('.fade').length == 0) {
			// Initial Update
			$(this.timeLocation).html(diff.html());
			diff = undefined;
		} else {
			$('.fade').fadeTo(400, 0.25, function() {
				if (typeof diff != 'undefined') {
					$(this.timeLocation).html(diff.html());
					diff = undefined;
				}
				$('.fade').fadeTo(400, 1).removeClass('fade');
			}.bind(this));
		}
	} else {
		if (this.displaySeconds) {
			$(this.timeLocation).html(_now.format(this._timeFormat+':mm[<span
class="sec">]ss[</span>]'));
		} else {
			$(this.timeLocation).html(_now.format(this._timeFormat+':mm'));
		}
	}
}

time.init = function () {

	if (parseInt(time.timeFormat) === 12) {
		time._timeFormat = 'hh'
	} else {
		time._timeFormat = 'HH';
	}
	this.updateTime();

}

var version = {
	updateInterval: 600000,
	intervalId: null
}

/**
 * Checks the version and refreshes the page if a new version has been pulled
 */
version.checkVersion = function () {

  $.getJSON('controllers/hash.php')
    .success(function(data) {
			// The githash variable is located in index.php
			if (data && data.gitHash !== gitHash) {
				window.location.reload();
				window.location.href = window.location.href;
      }
    });

}

version.init = function () {

	this.intervalId = setInterval(function () {
		this.checkVersion();
	}.bind(this), this.updateInterval);

}


var weather = {
	// Default language is Dutch because that is what the original author used
	lang: config.lang || 'nl',
	params: config.weather.params || null,
	iconTable: {
		'01d':'wi-day-sunny',
		'02d':'wi-day-cloudy',
		'03d':'wi-cloudy',
		'04d':'wi-cloudy-windy',
		'09d':'wi-showers',
		'10d':'wi-rain',
		'11d':'wi-thunderstorm',
			'13d':'wi-snow',
			'50d':'wi-fog',
			'01n':'wi-night-clear',
			'02n':'wi-night-cloudy',
			'03n':'wi-night-cloudy',
			'04n':'wi-night-cloudy',
			'09n':'wi-night-showers',
			'10n':'wi-night-rain',
			'11n':'wi-night-thunderstorm',
			'13n':'wi-night-snow',
			'50n':'wi-night-alt-cloudy-windy'
		},
		temperatureLocation: '.temp',
		windSunLocation: '.windsun',
		forecastLocation: '.forecast',
		apiVersion: '2.5',
		apiBase: 'http://api.openweathermap.org/data/',
		weatherEndpoint: 'weather',
		forecastEndpoint: 'forecast/daily',
		updateInterval: config.weather.interval || 6000,
		fadeInterval: config.weather.fadeInterval || 1000,
		intervalId: null,
		orientation: config.weather.orientation || 'vertical',
}

/**
 * Rounds a float to one decimal place
 * @param  {float} temperature The temperature to be rounded
 * @return {float}             The new floating point value
 */
weather.roundValue = function (temperature) {
		return parseFloat(temperature).toFixed(1);
}

/**
 * Converts the wind speed (km/h) into the values given by the Beaufort Wind Scale
 * @see http://www.spc.noaa.gov/faq/tornado/beaufort.html
 * @param  {int} kmh The wind speed in Kilometers Per Hour
 * @return {int}     The wind speed converted into its corresponding Beaufort number
 */
weather.ms2Beaufort = function(ms) {
		var kmh = ms * 60 * 60 / 1000;
		var speeds = [1, 5, 11, 19, 28, 38, 49, 61, 74, 88, 102, 117, 1000];
		for (var beaufort in speeds) {
			var speed = speeds[beaufort];
			if (speed > kmh) {
				return beaufort;
			}
		}
		return 12;
}

/**
 * Retrieves the current temperature and weather patter from the OpenWeatherMap API
 */
weather.updateCurrentWeather = function () {

		$.ajax({
			type: 'GET',
			url: weather.apiBase + '/' + weather.apiVersion + '/' +
weather.weatherEndpoint,
			dataType: 'json',
			data: weather.params,
            success: function (data) {

                  var _temperature = this.roundValue(data.main.temp),
                        _temperatureMin = this.roundValue(data.main.temp_min),
                        _temperatureMax = this.roundValue(data.main.temp_max),
                        _wind = this.roundValue(data.wind.speed),
                        _iconClass = this.iconTable[data.weather[0].icon];

                  var _icon = '<span class="icon ' + _iconClass + ' dimmed
wi"></span>';

                  var _newTempHtml = _icon + '' + _temperature + '&deg;';

                  $(this.temperatureLocation).updateWithText(_newTempHtml,
this.fadeInterval);

                  var _now = moment().format('HH:mm'),
                        _sunrise = moment(data.sys.sunrise*1000).format('HH:mm'),
                        _sunset = moment(data.sys.sunset*1000).format('HH:mm');

                  var _newWindHtml = '<span class="wind"><span class="wi wi-strong-
wind xdimmed"></span> ' + this.ms2Beaufort(_wind) + '</span>',
                        _newSunHtml = '<span class="sun"><span class="wi wi-sunrise
xdimmed"></span> ' + _sunrise + '</span>';

                  if (_sunrise < _now && _sunset > _now) {
                        _newSunHtml = '<span class="sun"><span class="wi wi-sunset
xdimmed"></span> ' + _sunset + '</span>';
                  }

                  $(this.windSunLocation).updateWithText(_newWindHtml + ' ' +
_newSunHtml,this.fadeInterval);

            }.bind(this),
            error: function () {

            }
      });

}

/**
 * Updates the 5 Day Forecast from the OpenWeatherMap API
 */
weather.updateWeatherForecast = function () {

      $.ajax({
            type: 'GET',
            url: weather.apiBase + '/' + weather.apiVersion + '/' +
weather.forecastEndpoint,
            data: weather.params,
            success: function (data) {

                  var _opacity = 1,
                        _forecastHtml = '<tr>',
                        _forecastHtml2 = '<tr>',
                        _forecastHtml3 = '<tr>',
                        _forecastHtml4 = '<tr>';

                  _forecastHtml = '<table class="forecast-table"><tr>';

                  for (var i = 0, count = data.list.length; i < count; i++) {
                        var _forecast = data.list[i];

                        //don't show yesterday's forecast; each date, .dt is 12p
local;
                        var _12hours = 60 * 60 * 12 * 1000;
                        if (_forecast.dt < Math.floor((Date.now() - _12hours) /
1000)) continue;

                        if (this.orientation == 'vertical') {
                              _forecastHtml2 = '';
                              _forecastHtml3 = '';
                              _forecastHtml4 = '';
                        }

                        _forecastHtml += '<td style="opacity:' + _opacity + '"
class="day">' + moment(_forecast.dt, 'X').format('ddd') + '</td>';
                        _forecastHtml2 += '<td style="opacity:' + _opacity + '"
class="icon-small ' + this.iconTable[_forecast.weather[0].icon] + '"></td>';
                        _forecastHtml3 += '<td style="opacity:' + _opacity + '"
class="temp-max">' + this.roundValue(_forecast.temp.max) + '</td>';
                        _forecastHtml4 += '<td style="opacity:' + _opacity + '"
class="temp-min">' + this.roundValue(_forecast.temp.min) + '</td>';

                        _opacity -= 0.155;

                        if (this.orientation == 'vertical') {
                              _forecastHtml += _forecastHtml2 + _forecastHtml3 +
_forecastHtml4 + '</tr>';
                        }
                  }
                  _forecastHtml  += '</tr>',
                  _forecastHtml2 += '</tr>',
                  _forecastHtml3 += '</tr>',
                  _forecastHtml4 += '</tr>';

                  if (this.orientation == 'vertical') {
                        _forecastHtml += '</table>';
                  } else {
                        _forecastHtml += _forecastHtml2 + _forecastHtml3 +
_forecastHtml4 +'</table>';
                  }

                  $(this.forecastLocation).updateWithText(_forecastHtml,
this.fadeInterval);

            }.bind(this),
            error: function () {

            }
      });

}

weather.init = function () {

      if (this.params.lang === undefined) {
            this.params.lang = this.lang;
      }

      if (this.params.cnt === undefined) {
            this.params.cnt = 6;
      }
```

```
		this.intervalId = setInterval(function () {
			this.updateCurrentWeather();
			this.updateWeatherForecast();
		}.bind(this), this.updateInterval);
		this.updateCurrentWeather();
		this.updateWeatherForecast();
	}
```

Appendix:
Hardware Images

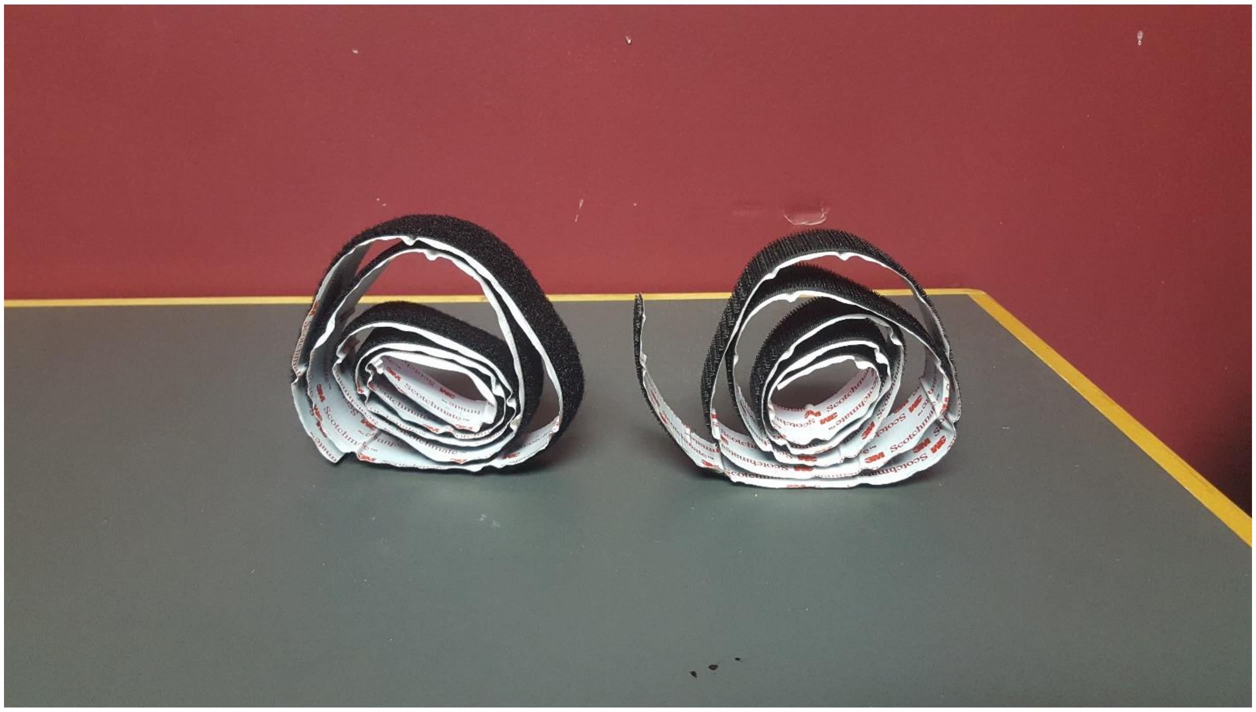
Figure 1: Scotch Tape Velcro

Figure 2: Portable Battery Pack

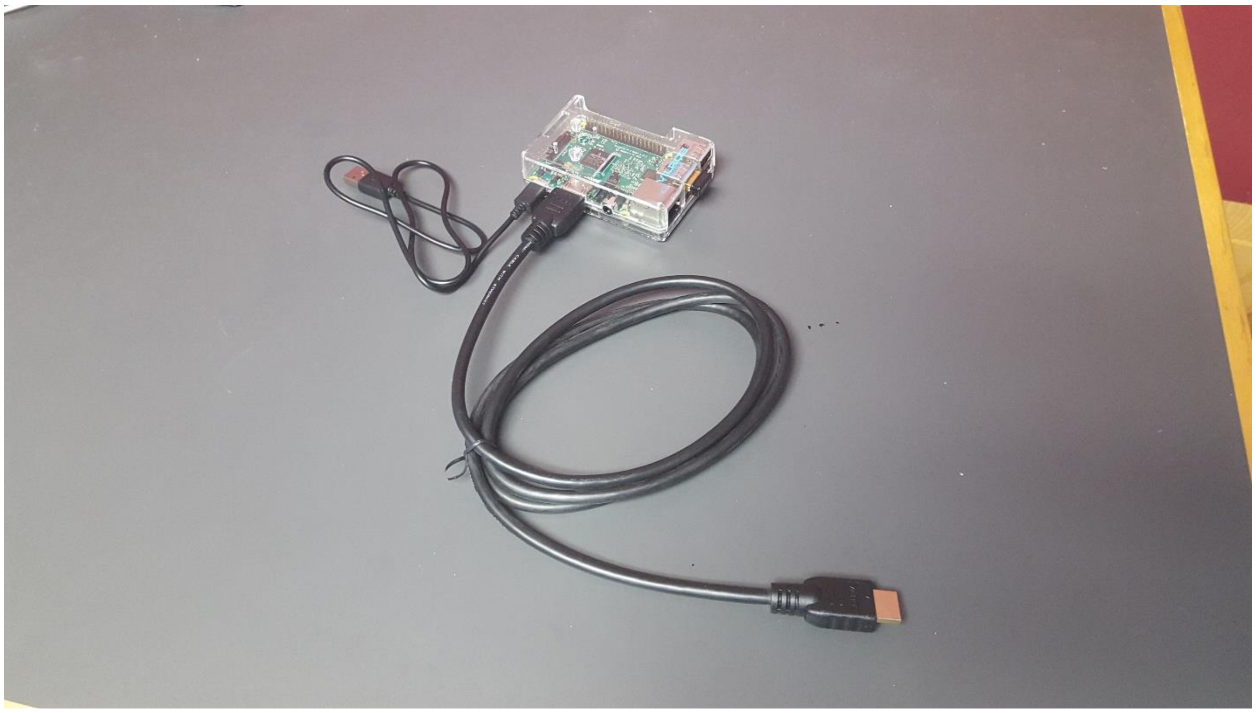

Figure 3: Raspberry PI Controller

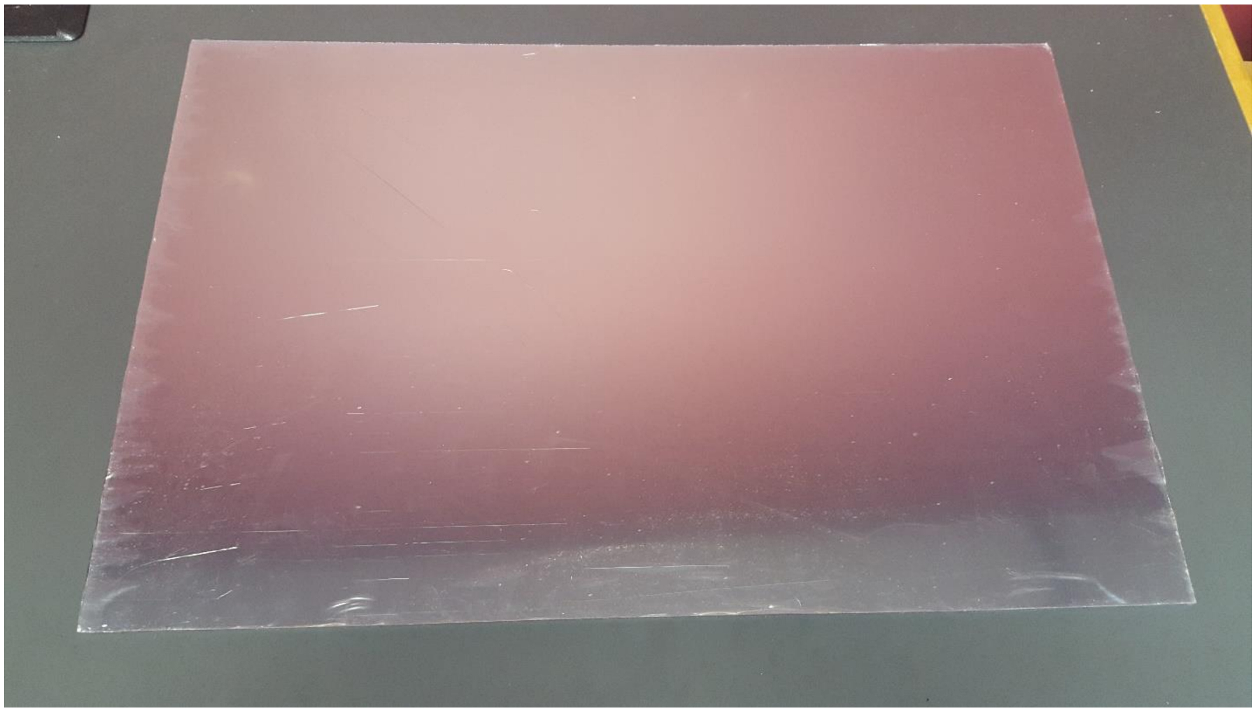

Figure 4: Two Way Mirror with covering attached.

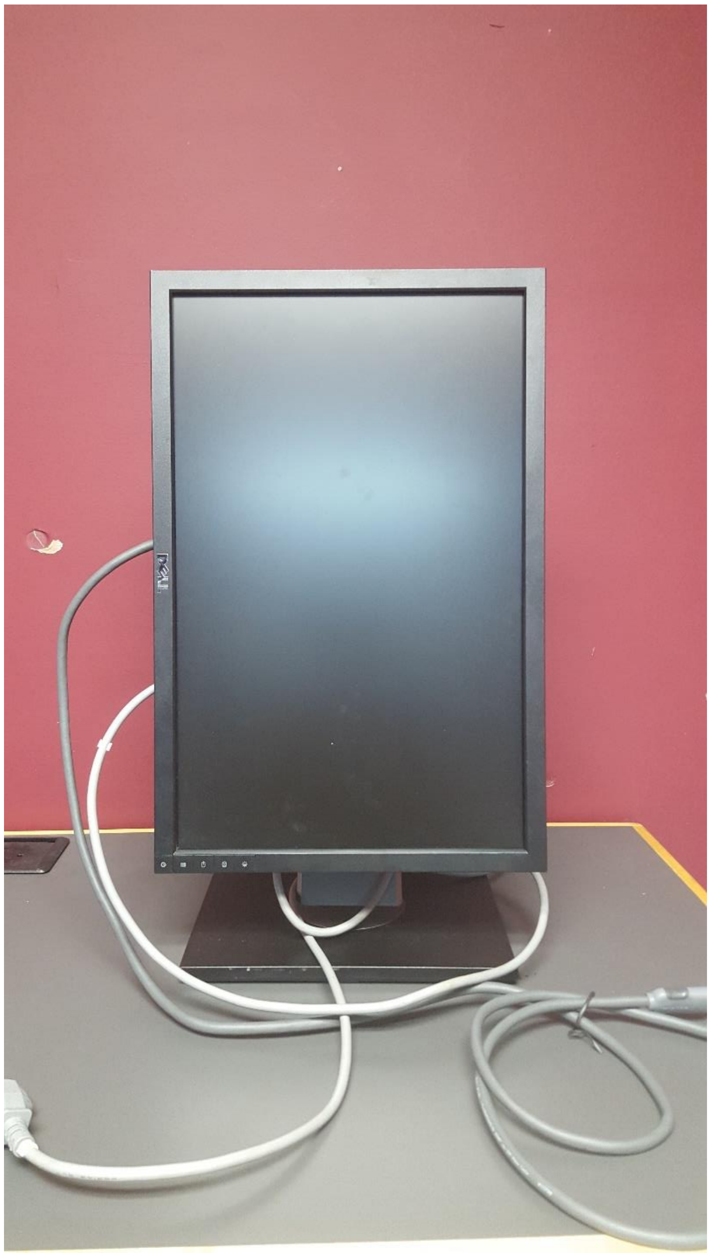

Figure 5: Monitor

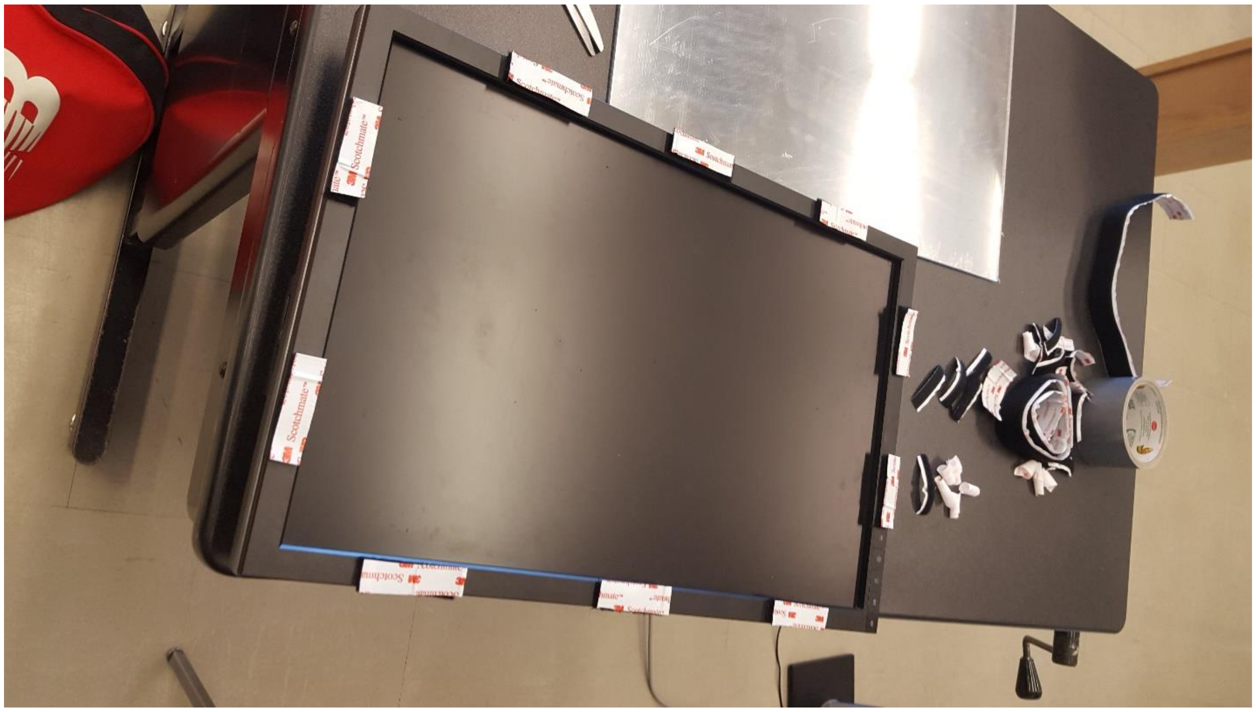
Figure 6: Monitor with scotch tape Velcro straps connected

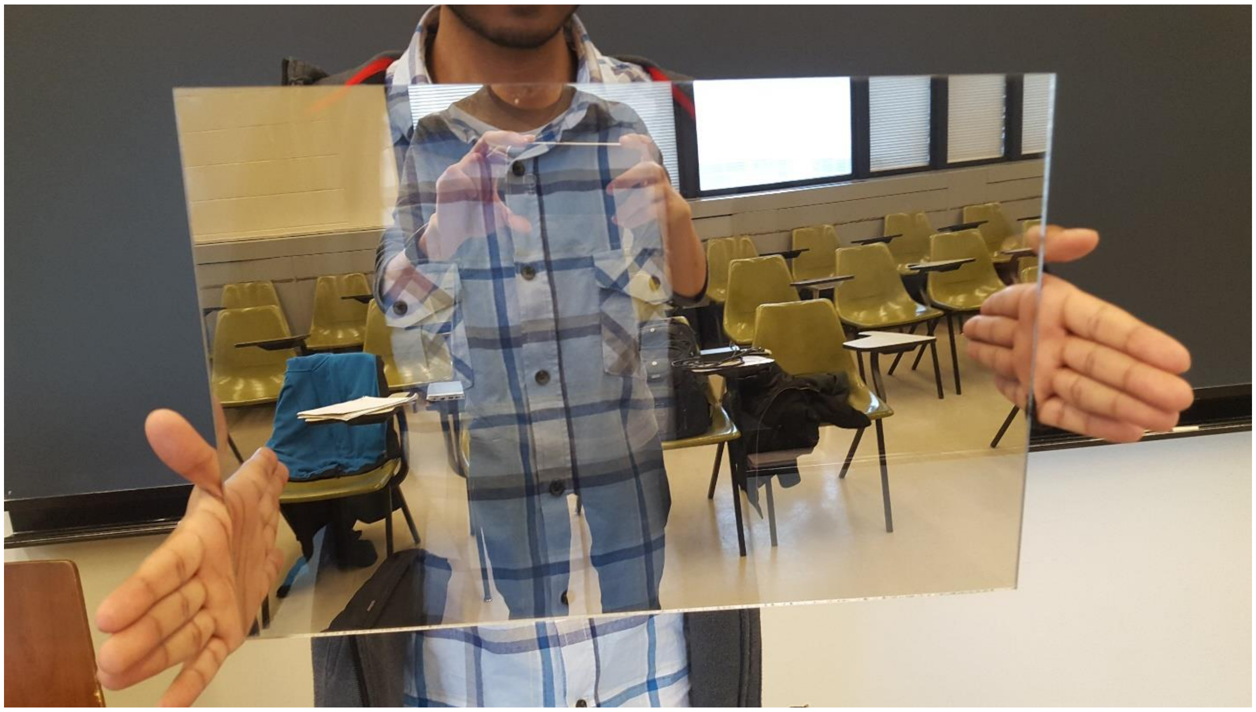
Figure 7: Two Way Mirror

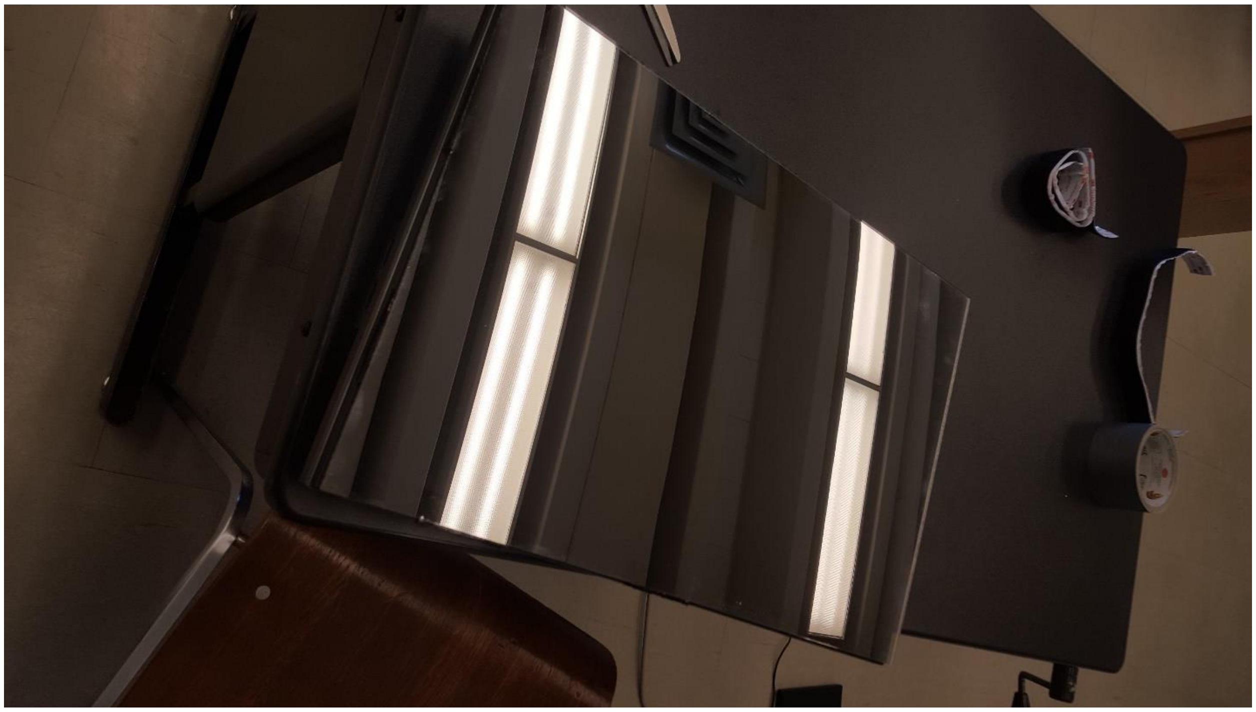

Figure 8: Mirror connected to the monitor

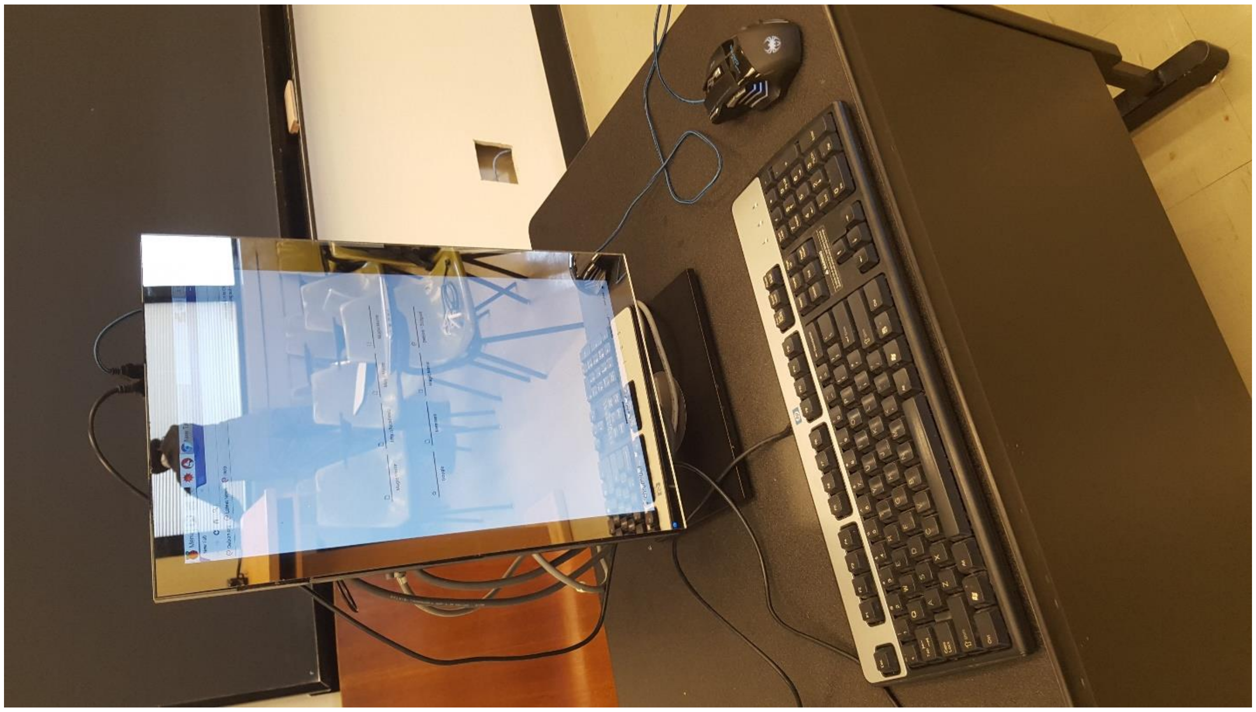

Figure 9: A Keyboard and a Mouse were used to set up the Smart Mirror

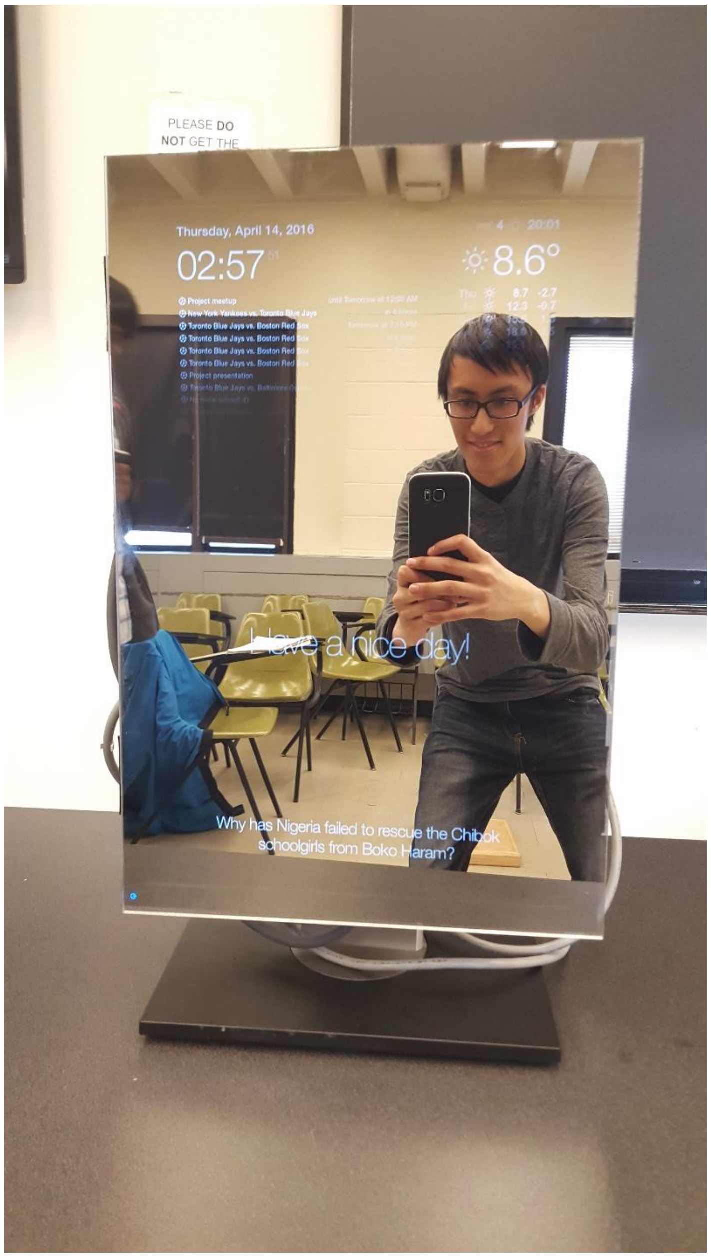
PLEASE DO NOT GET THE
Thursday, April 14, 2016
02:57
8.6°
Project meetup
New York Yankees vs. Toronto Blue Jays
Toronto Blue Jays vs. Boston Red Sox
Project presentation
Have a nice day!
Why has Nigeria failed to rescue the Chibok schoolgirls from Boko Haram?

Figure 10: Mirror connected to monitor displaying text.

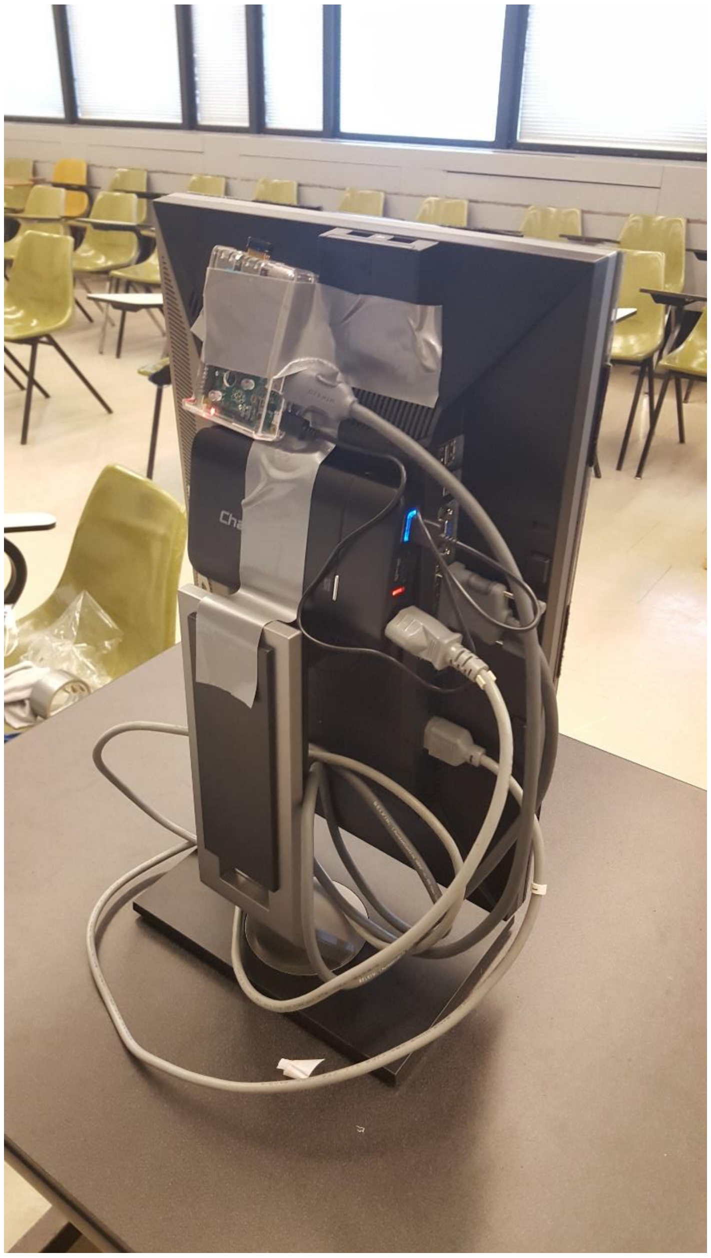

Figure 11: Back side of the Smart Mirror.