**Table of Contents**

**1. Introduction…………………………………………………………………...…...3**

**2. Purpose………………………………………………………………………...…...3**

**3. Requirement Specification…………………………………………………...…..3**

**3.1.Functional Requirements…………………………………………...….4**

**3.2. Non-Functional Requirements………………………………………...5**

**4. Scrum Methodology …………………………………………………………......5**

**5. Software….……………………………………………………………………..…8**

**5.1 IOS Application …...………………………………………………...….8**

**5.1.1 Introduction…………………………………………………...8**

**5.1.2 Initial Design………………………………………………….8**

**5.1.3. Conclusion…………………………………………………..10**

**5.2 Android Application……………………………………………………10**

**5.2.1. Introduction…………………………………………………10**

**5.2.2. GUI Architecture ..…………………………………………11**

**5.2.3. Streaming from Twitch TV to Android …………….........15**

## 5.2.4. Twitch TV App……………………………………………....17

**5.2.5. Streaming from Android to Twitch………………………..17**

**5.2.6. Conclusion ..………………………………………………...17**

**5.3 Website Application...……………………………………………...….18**

**5.3.1. Introduction ..……………………………………………….18**

**5.3.2. Initial Design Features ..………………………….…...…..18**

**5.3.3. Conclusion………………………………………….………..21**

**5.4 Back End System……………………………………………………..**

**6. Hardware ..…………………………………………………………….………... 21**

**7. Software Testing…………………………………………………………..……. 23**

**7.1 IOS.…………………………………………………………..………... 23**

**7.2 Android ..…………………………………………………..………….. 24**

**7.2.1. Test plan ..………………………………….….…..…….... 24**

**7.2.2. Test result .…………………………….………........……. 25**

**7.3 Website ……………………………………………………….……..**

**8. Conclusion……………………………………………………………………**

**9. Appendix………………………………………………………………………**

**9.1 Contribution………………………………………………………...**

**9.2 Test cases……………………………………………………..…….**

**9.2.1 Website Test Cases……………………………………...**

**9.2.2 Android Test Cases……………………………………...**

**9.2.3 IOS Test Cases…………………………………………..**

# 1. Introduction

When baking, the cook usually stands in front of the oven and checks the food regularly by looking through the oven windows. Team Fruit Cakes will improve this experience with the Pie Checker. This device can be attached to the oven and it will monitor the food, and relay specific information to any of the devices supported by the Pie Checker. This allows the baker to enjoy the freedom of movement while waiting for their food to bake. Team Fruitcakes supposed that the PieChecker can be used in the real life, it should not be only a school project.

The Piechecker consists of a Raspberry pi, camera and wifi which are contained in a case that the team produced. the probe for measuring food temperature is planned as removeable part. Team Fruitcakes orders some parts of the PieChecker from China apart from the Raspberry pi camera and WiPi which are offered from the department.

About the idea of the software part, Piechecker sends a live video stream to the webpage via a server and the baker can check the food video and temperature on the website or mobile application. The applications follows the flow and interface which is design by a team’s member. The previous documentation in the project such as the project idea and requirement specification, are reference document of the project idea and plan.

Team Fruitcakes uses the scrum methodology to manage this project. Github is the tool being used to store all the code and results for the project. The scrum master’s main job is to update the burndown chart and keep communication between the product owner and scrum team.

The project tasks are created in Trello.com which is the tool that the team’s member can pick up their own tasks in each sprint. In this semester project (spring 2014), we, team Fruitcakes emphasizes on both the scrum methodology and the product development. and we hope that our agile scrum knowledge and technical skill on the application development are improved.

# 2. Purpose

* The team should understand the scrum methodology which is used to manage the project.
* Application is connected to the PieChecker through the internet.
* The Video streaming and the temperature are displayed on the application or website by getting informations from PieChecker.

# 3. Requirement specification

PI checker, the essential product with the basic functionalities of users login, online broadcasting and image storing for the user. It can send constantly updated images through the home internet to specific devices such as a computer or mobile phone.

We assume that the baker has a wireless network available and a smartphone with Android, IOS operating system or internet browser. However, we have certain product constraints such as the following:

* PieChecker must be in a splash proof box.
* It must operate remotely via wireless network
* PieChecker must be able to perform timer and temperature alarms even if the server connection is lost.

The requirements consist of functional and nonfunctional requirements that the project should fulfill.

## \*\*\*3.1.Functional Requirements:

The user wants to:

* Register an account
* Connect a PieChecker to his / her account
* Connect an Android/IOS phone to his / her account
* Attach the PieChecker to the oven
* Connect the PieChecker to the WiFi network
* Set a time or temperature alarm
* Place probe into pie
* Use a recipe from the database
* Checking current step from the whole baking process
* See previous baking sessions
* View the live stream
* Share photos
* Change settings
* View temperature changing diagram

## 3.2. Non-Functional Requirements:

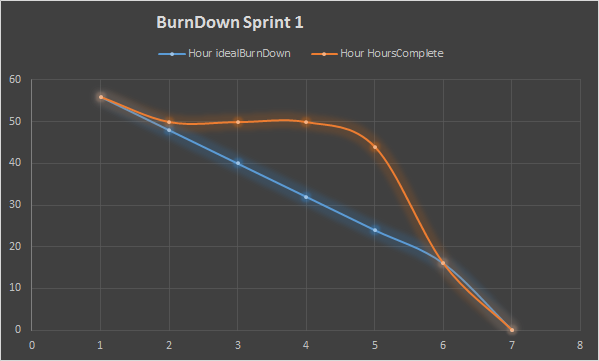
* The system must update the temperature with less than 5 secs interval.
* The system must update the image with less than 5 secs interval.
* The alarm must be given when the temperature is reached, both on the PieChecker and in the web interface.
* The timer alarm must be given on both the PieChecker and the web interface at the same time.

# 4. Scrum Methodology

Team Fruitcakes used the scrum methodology to organise this project. The Product owner is a second year supervisor and one team member is the scrummaster. Each team member decides their own task for each sprint, as the team has a sprint review for 14 days. In addition to the sprint review, the sprint planning and retrospective are also done during the sprint review session. The selected tasks must be finished within each sprint period. Trello is a tool that is used to manage assigning and tracking tasks in each sprint.

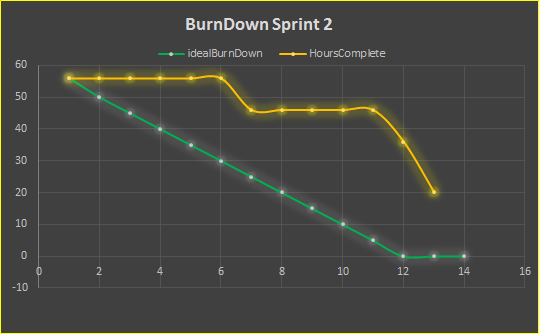
The burndown chart is used to compare ideal task remaining and actual task remaining. The scrum master creates the burndown chart in each sprint as following:

Sprint review 1(Feb 26-Mar 5, 2014):



It took 7 days for the first sprint and the points for each product backlog was denoted as working hour. The product backlogs in the first sprint are mostly about documentation, researching and initial design for the project. Refering to figure 4-1, the team Fruitcakes finished all tasks in the sprint, even the the actual hour complete doesn’t the same line as the ideal burndown.

Sprint review 2(Mar 5-19, 2014):



We started to take 14 days sprint from the second sprint in figure 4-2, total points are 56 and assigned a bit more technical work in the product backlogs for example setting up database, add login function to the android application. Most of the product backlogs are done, it’s just few tasks are not done in the sprint. We got comment from supervisor that the tasks should be individual, each task must be defined the criteria and the points should be denoted by man day instead. Since, the product backlogs were assigned to the team, we decided improve our sprint in the next sprint, referring to the comment.

Sprint Review 3 (Mar 19 - April 2, 2014) :

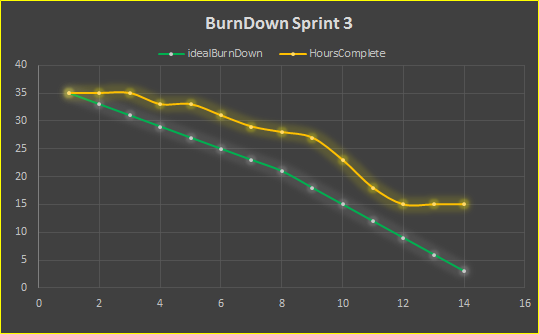
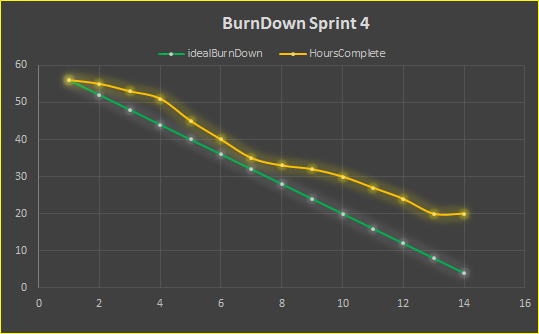


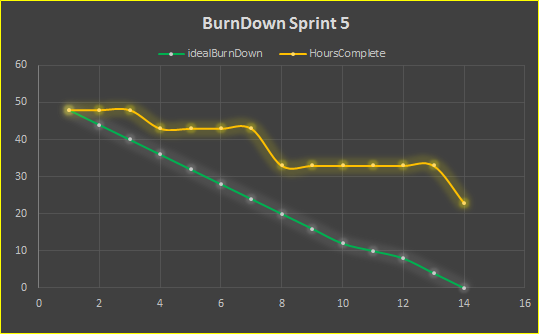
Figure 4-3 shows the burndown chart for the third sprint. The points for the product backlogs were assigned by manday. The points is 35 points for entire sprint, 14 days and there are a few tasks are not done. The period we doing sprint 1-3, is the same period as studying other courses and exam period so it’s not much points assigned to the sprints.

Sprint Review 4 (April 2 -16, 2014):



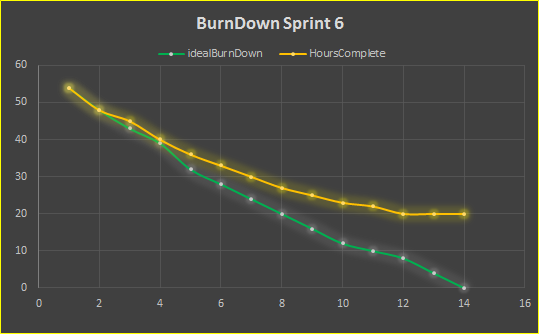
The fourth sprint is shown in figure 4-4, the hour completed is not in the same line as ideal burndown. We assign more points into the sprint since we were done with the other courses. the unfinished tasks from the previous sprint were moved to this sprint. The points for this sprint are 56. However, there are some tasks are not finished, then we moved the tasks to the next sprint.

Sprint Review 5 (April 16 - 30, 2014):



The fifth sprint in figure 4-5, since one of our member was really sick, we got big gap between the ideal burndown and hour completed. Some tasks in the sprint had to wait for the other task done, then it is quite many tasks are not finished in the sprint. most of the team’s member spent much time on the researching.

Sprint review 6 (May 1- 14, 2014):



In the sixth sprint - figure 4-6, Android videis decided to not finished since we realise that it need longer time to fix video streaming in android application. 54 points were assigned to this sprint. there are some tasks that we really need to finished it at the final week of the project. The website, IOS video streaming and hardware had to be fixed at the last week after this sprint.

Each task in a sprint is decided to points depending on the difficulty and the task owner’s skill level, team Fruitcakes defined a manday (8 hours) as a point. Referring to the charts above, most of the tasks in each sprint are done. The unfinished tasks from previous sprints are moved to the current sprint. The main cause of unfinished tasks is technical debt, uncontrollable situations. For instance, an examination held during a sprint takes time away from the tasks. In addition, unfinished relevant tasks and the sickness of team members are also technical debt.

The scrum meetings did not occur every day, but the tasks have been followed up on by the scrum master.

**Conclusion**

Integrating scrum into an already busy schedule was not easy. Assignments from other courses would regularly cause technical debt. Lack of experience meant that estimates of how long tasks would take had little chance of being accurate. While we should, in theory, have had daily scrum meetings this was not practical due to irregular schedules and no assigned workspace at the university. The use of a second year student as the product owner made sense initially but seeing them only every two weeks made everyones job harder. They couldn’t be expected to maintain control of the product but then no one in the group had overall control either. There was clear conflict between what a scrum member was supposed to do and what we actually had to do.

In conclusion, while scrum is proven to work in the right workplace it is almost impossible to use it correctly in our student environment.

**System overview**

# 5. Application

## 5.1. IOS

### **5.1.1 Introduction**:

This application has been made to be used by a PieChecker user. The IOS application is designed to run on iPhones, making it remotely easier to keep control over your bakings and as a another choice of viewing, alongside the website. The application uses a live streaming website as a source. The application is created using Xcode 5.0 and Adobe Photoshop CS5 for visuals.

### 5.1.2 Initial Design:

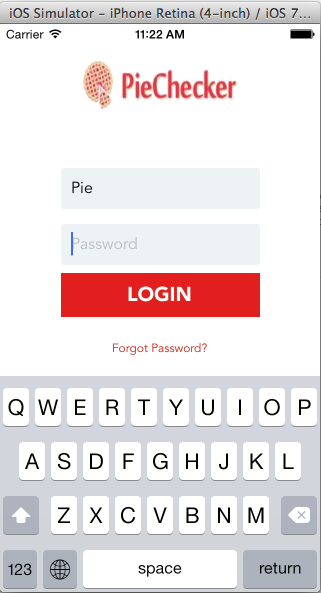
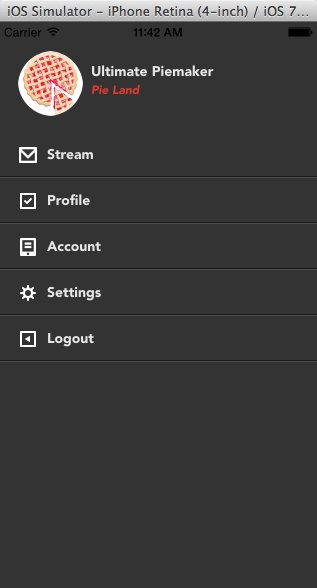
The application is designed to have four different pages to navigate through, including the Login Page. These are easy to navigate through via the sidebar, by just sliding or clicking the menu icon.

**Login Page:**

When the application is opened, this page is the first thing that is seen, from here they can login with their username and password.

**Sidebar:**

From this section they can navigate through the applications different pages.

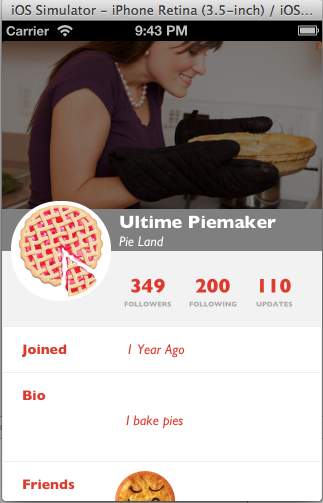


**Profile Page:**

This is the users profile page when information about themselves can be written and use a profile picture, some features such as followers, following, etc may be deleted but that is to see.

**Video Page:**

Still in progress to get a working stream.



### 5.1.3. Conclusion:

The application is not yet finished but so far there have been quite a few of obstacles to face. Since this is the first time programming an application, this was a bit difficult. The biggest issue so far has been trying to fix the livestream, some services don’t allow live streaming outside of their own application. Even though a live streaming service may allow it, it is difficult for a beginner to find a way.

## 5.2. Android Application

### 5.2.1. Introduction

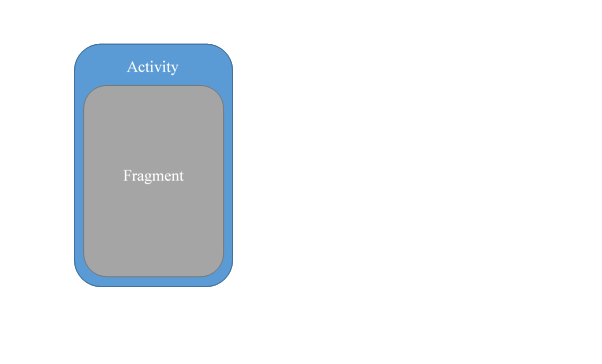
The purpose of the android application was to replicate all the functionality of the PieChecker website. The application is designed to function on smart phones running the android operating system and the architecture is designed in such a way that the GUI will be adaptable to tablet size screens.

All testing was carried out using a Samsung Galaxy S4 running Android version 4.4.2 (Kitkat). All code was created using Eclipse with the Android SDK.

### 5.2.2. GUI Architecture

The UI framework uses a Model-View-Controller (MVC) pattern. The view is responsible for what is seen on the screen. The controller is responsible for reacting to external events such as keystrokes, screen taps or phone calls. The model is the core of the application. The controller updates the model, the model invalidates the view and causes it to be redrawn if necessary.

As the application should run on smart phones and tablets it utilises fragments within the view.



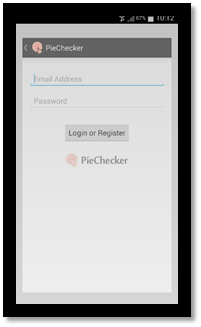
The activity represents the view of the screen and has no components. The activity is needed to hold the fragment. Each view of the GUI has its own fragment and they are swapped as needed. For example the home and video pages on the website are reproduced in two separate fragments for the android app.

**Splash Screen**



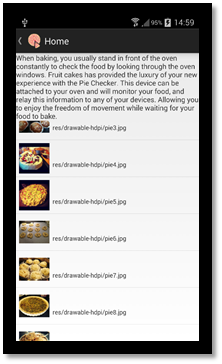
The splash screen is the first thing the user sees when starting the app. It is run in its own thread to allow other processes to be performed at the same time. The original intention was to allow the app to retrieve any necessary data at this point.

**Login Screen**



The login screen in its current state was designed to work with a version of the website no longer in use. The code takes the entered email address and posts it to the URL. This would trigger the account system to send an email to the user with instructions on registering.

**Home Screen**



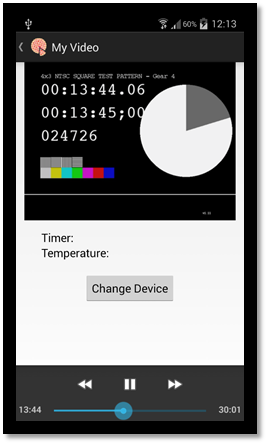
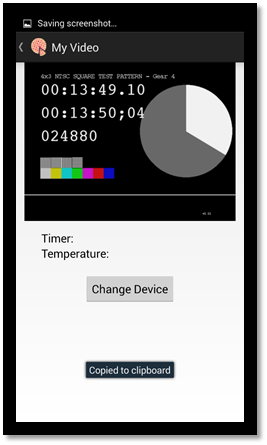
The home screen demonstrates the use of a list view containing more than one object. The images shown are held in the resources of the app. In a real app the images would be held in a remote database. To improve performance of the app the images would be fetched once, scaled to an appropriate size and held in a database on the smart phone.

**Gallery Screen**



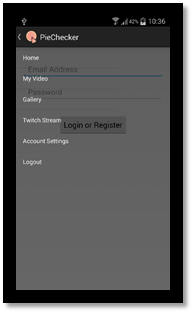
The gallery screen is used to demonstrate how multiple images can be shown on a small device. The view can be scrolled if there are more images than will fit on the screen. As with the home page the images are held on the device but would normally be on a remote database. The app would need to be modified to perform scaling on any downloaded images before attempting to display them.

**My Video Screen**



The video screen use a video view in the top half of the screen and place holders for the temperature, timer and a button that would be used for changing the device being viewed. When the video view is pressed a controller appears in the bottom of the screen. The video in the images is a sample m3u8 stream from Apple. For the first demonstration the app was shown playing an mp4 file that was stored locally on the phones SD card.

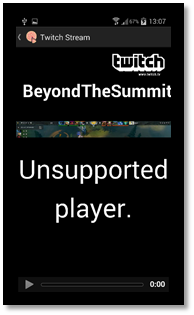
**Navigation**



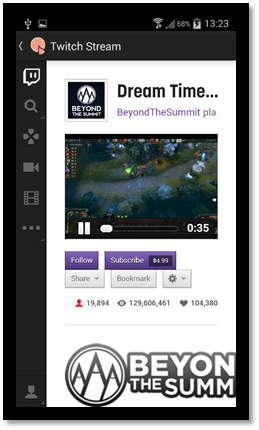
The user navigates through the app using a drawer layout with a list. When the user selects an item in the list the app displays the appropriate fragment. The menu is displayed by either swiping from the left across the screen or pressing the arrow key in the top left of the image.

**Streaming from Twitch TV to Android**

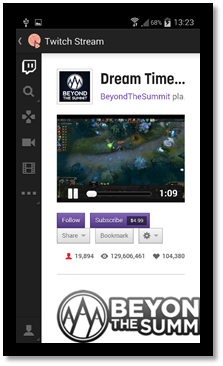
Streaming from Twitch is buggy at best and unworkable at worst. Twitch uses SWF file format which is only playable using Flash player. Flash is no longer supported by Android and Adobe no longer have support to allow Flash to play on Android. Twitch also uses HLS, or HTTP Live Stream, for most of there streams. While HLS is technically supported on Android it is extremely limited. It should, in theory, be possible to show a HLS stream in a web view. When this was attempted the image below would be shown.



The app was also testing showing the whole page in a web view.



The image above shows that the page in a web view appears to work. The audio started at 0 seconds but the video does not proceed beyond the initial frame. The image below shows the same stream at a later time. The video has still not changed.



It should be noted that at no point did the stream play any advertisements and when the app did not have focus the audio continued to play.

**Twitch TV App**

Twitch TV has its own app that works on Android phones. It is currently free to use and allows the user to login and watch streams.

**Streaming from Android to Twitch**

Given the lack of experience with Android and streaming in general this was always going to be a difficult task. Twitch TV is in the process of creating an SDK for mobile users but this has yet to cover Android devices. There also exist several apps for streaming across a Wi-Fi network. The most commonly mentioned one, SpyDroid, was tried but it failed to stream any video but this is more likely to be bug related to hardware or using the latest Android version. The end result is that no progress was made here. The only real solution would be to create something completely from scratch and this is far beyond the current capabilities of the team member assigned to this.

**The Code**

When the streaming part of the app not achieved development was halted. The submitted code demonstrates the progress made with Android programming but should not be considered as anything but a demo/testing platform. In its current state the code can built and run on a smartphone with KitKat through Eclipse. The build is not considered stable but it is safe to run as the only permission it requires is internet access. One error to watch out for is rotating the screen while the app is running. This will cause the app to crash. The app will function properly if started in a horizontal or vertical position but should not be rotated. There are other minor bugs but as the app was scrapped they were never fixed.

For instructions on setting up Eclipse to work with Android and for a vast amount of other Android related information visit<http://developer.android.com/training/index.html>.

**Conclusion**

The biggest obstacle to overcome for this part of the project was the lack of experience with Android development. At least 50% of the time spent was on research and tutorials to understand the basics. Late changes to the back end and no database of images have blocked development of correctly implementing the getting and processing of images for display. The change to Twitch as a streaming service has made it difficult to stream to the app without the Twitch app being installed, even with the app the performance is extremely buggy when attempting to watch in a web view. Streaming from a smart phone was also not achieved. At the time of writing there is still no SDK or API for Android from Twitch and despite several days effort no solution was found.

In conclusion, streaming to and from Android is the very first thing that should have been researched. It is extremely unlikely that the app would have been started had the group understood just how little information there was available and how little support there was for Android streaming. On the positive side there was significant knowledge gained while learning about Android development that can be utilised in future projects.

## 5.3. Pie Checker Website

### 5.3.1. Introduction:

The Pie Checker website was created solely for the use of the product. This is how the user will be able to view their food that is baking in their kitchen. The product enables the user to remotely view their food via live streaming video, save images to their gallery and view other user’s shared images. It was created using a combination of HTML, CSS, and Javascript. The design guidelines were made using Photoshop. The fame work, such as the buttons and text, were created using HTML and completed with placement and colors using CSS. The functions were added using Javascript

### 5.3.2. Initial Design Features:

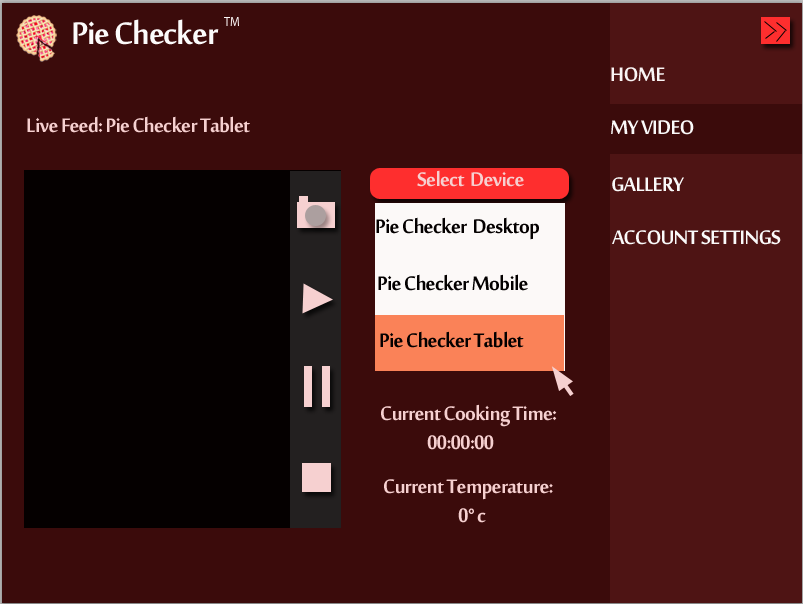
This website contains five different pages in which the user can navigate through via a sidebar menu. Each page link is represented on the menu bar along with an option to logout of the account.

**Login page:** This page is where the user must either login to or register to Piechecker. After registering the user will receive a confirmation email for their account.

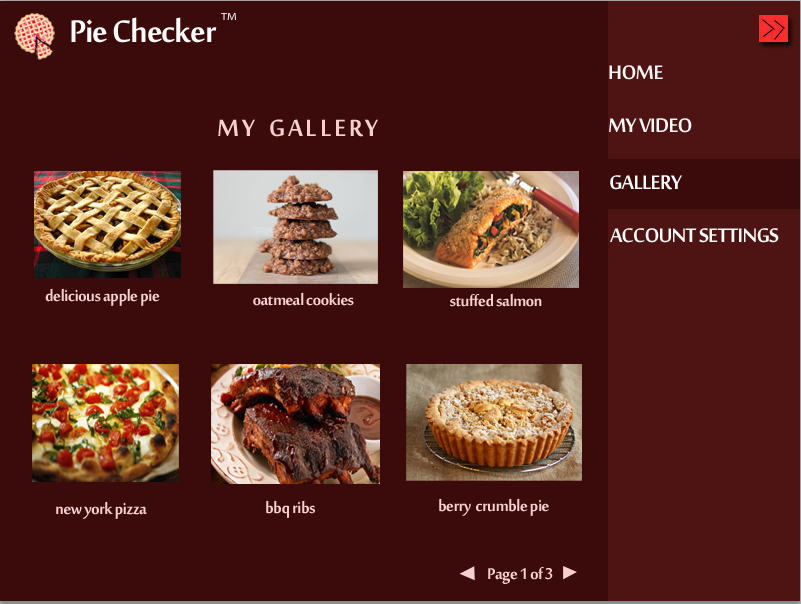
**Home page:** The homepage is where the user can see pictures shared by other users and see a description about the product.



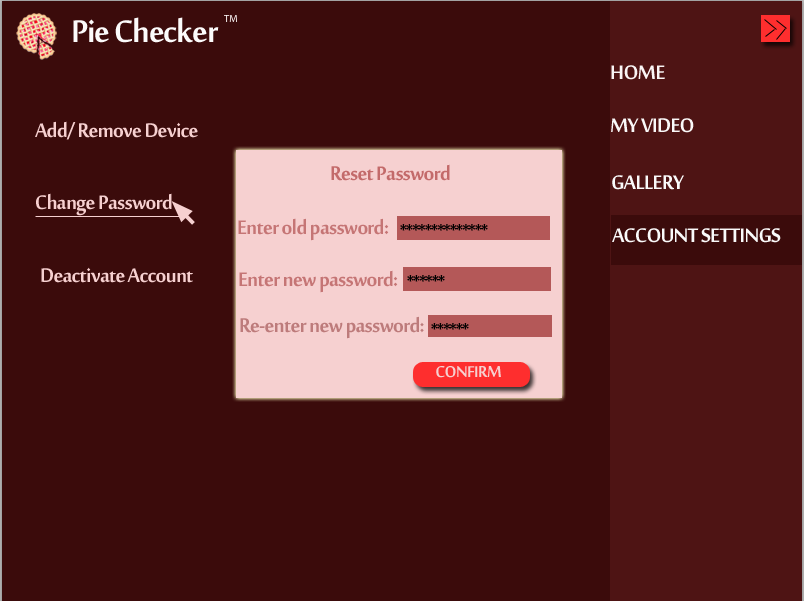
**My Video page:** The my video page is where the user will be able to view their live streaming video. They will be able to pause or stop the video as they desire and have the option to change the view to full screen. Also the user can select which device they wish to see if they have more than one pie registered to their account. On this page, the current cooking temperature and time are displayed for the users convenience.



**Gallery page:** The gallery page is where the user will be able to review their previously stored recordings and thumbnails of the dishes. By simply clicking on an image, the user will be able to view it in a larger form.



**Account Settings page:** The account settings page is where the user will have the option to add or remove Piecheckers from their account, change their password, or even deactivate their account completely.



**Finished Design Features:**

**Initial Login/Register page:** when the user goes to our Piechecker website (<http://piechecker.com>) they will have an option to register via TwitchTV. Our project’s login system will use a TwitchTV account in order to stream and save live videos. after logging in, the user will be redirected to the Piechecker homepage to continue on with using our product.

**Home Page?:**

**Setup Page?:** first page you see when logging in

**About Us Page?:** logged out

**Streamer Page?:** canwatch other user’s streams

**5.3.3. Conclusion:**

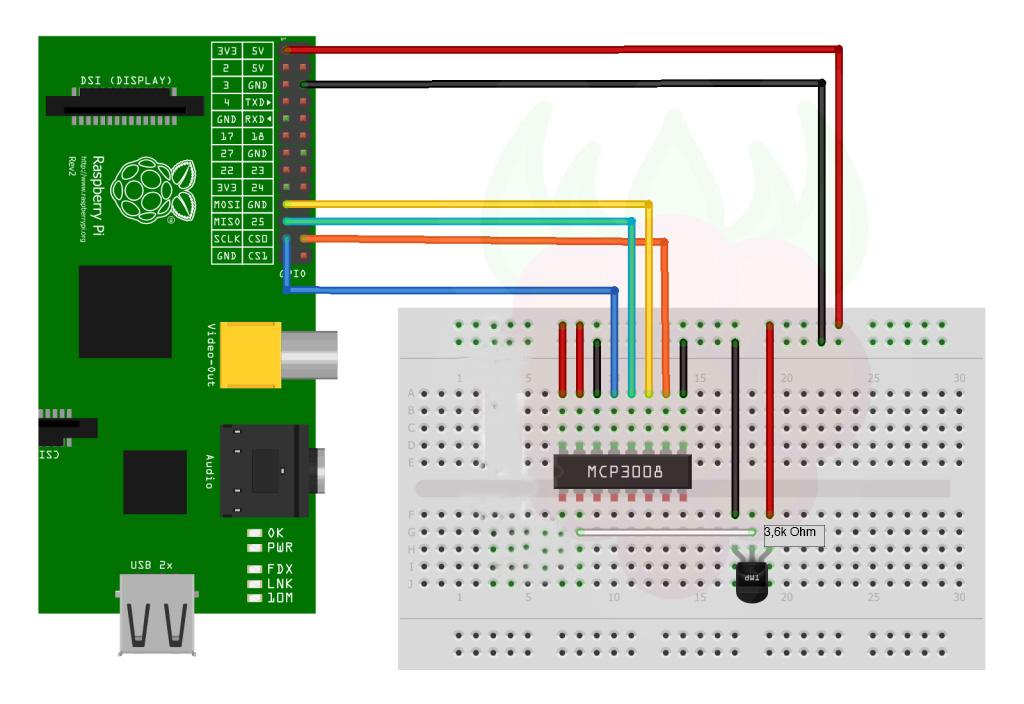
The initial design has many differences compared to the actual product. When developing the website, we faced several challenges to make the pages exactly how we would have liked to have them. One huge aspect of the website that changed was our login system. We realized that if we use our own server and database, the users’ video streaming would accumulate a very hefty cost for us in which we are not willing to pay. Instead of using our own server, we are piggy backing off of a program called “TwitchTV”. by logging into your TwitchTV account, the user will manually redirect the page back to Piechecker.com and our website will recieve all the users information from TwitchTV. By changing our backend system so drastically, we didn’t have enough time to complete our website following the initial design. After all the changes were made, the website looks different but the requirements for functions are still met. It would have been easier to continue using what we had except the login system would not have functioned properly.

Because we are using TwitchTV more or less as our “back-end”, we can use the SD card for the pie as our database. We store the code and everything the user needs in order for the piechecker to work on the card and the user can download the info from our website. This help us to have a nice and easy installation process.

# 6. Hardware

The measuring of temperature was a big part of our project and to implement it we purchased a number of things:

* Pt-100 thermocouple
* M/F and M/M jumper wire
* Mpc-3008 ADC
* Breadboard
* 3.3 Kohm resistor

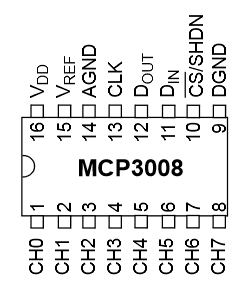
The pt-100 is an analog device and to convert the signal for the rpi we had to purchase the mpc-3008 an analog to digital converter. The mpc-3008 was chosen specifically because it was one of the cheaper and more commonly used adcs. The breadboard was purchased because of the simplicity in creating a breadboard circuit. 

*Figure x: How our circuit appears on a larger breadboard*

Connecting the temperature probe also required work on the software side.

First of all the rpi uses GPIO-pins for connecting devices and to connect our adc and probe we had to change to protocol to SPI luckily rpi supports spi so the change was accomplished by simply changing a config file in rasbian.

To work with the signal on the rpi we used a python 2.7 library called spidev which was the most widespread library for this kind of thing.



*Figure x:Breakdown of the ports on the adc*

Since the probe is a thermocouple the signal you receive from it is it’s resistance which means we that had to convert it into a celsius value using python.

As the team Fruitcakes needs additional parts, we decided to order the parts from China, since it is cheaper cost to buying it from China.

Items bought from china:

* mini speaker
* PT-100
* BerryClip 6 LED add on board
* LED
* clock model DS3231
* Raspberry Pi casing
* Button S1601

During the brainstorming meeting we make the decision about what kind of product we will develop with Raspberry Pi. After some researching on the what type of hardware needed and could be fit with pi. The first purchase was made on a chinese b2c website called “TaoBao” by delivery through DHL during one week. Through this method we solved financial issue and still maintain the quality of product.

## 6.1 Conclusion

The integration of our hardware into the project was started very late compared to the front end development which almost lead to it not being integrated. But during the last weeks of the project much focus was put on the hardware which lead to it being finished. When we ordered the first batch of items for the hardware no one really knew what we had to buy to connect the probe and it was ignored for a long time, luckily the components we needed were readily available from local electronic stores. The accuracy of the temperature probe could have been better if we bought a more expensive adc but accuracy was not really one of the goals so it was ignored. To conclude we had some issues when implementing the hardware but everything was resolved in an almost timely fashion.

# 7. Software Testing

## 7.1. IOS

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Revision Date | Roles | Author |
| 001 | 14/05/2014 | Test Case | Saipirun Sanprom |
| 002 | 22/05/2014 | Interface Testing | Shan Jiang |
| 003 | /05/2014 | Function Testing | Shan Jiang |

**Reference:**

•IEEE 829 - Standard for Test Documentation

(<http://gerrardconsulting.com/tkb/guidelines/ieee829/main.html>)

•IOS\_TestCase

(<https://docs.google.com/spreadsheets/d/1nY6qoaOlndDIiv-fS_s01xPkNtuU5Hf6hn2j19Mw9vA/edit?usp=sharing>)

•Project Requirements Document.

**Introduction:**

This is a high-level overview defining the testing strategy for the Pie Checker IOS application produced by FruitCake. Testing steps were processed base on the IEEE Standard 829, Test Case and Design Document to validate required quality before product release.

**Test Items**

**Items To Be Tested:**

Since the app’s development is not done, and the project package didn’t deliver to tester yet. Following steps are only the plan for testing

**•Streaming video.**

open “My Video” page, choose the device which user is using, click start button to turn streaming on.

**•Buttons’ function related to video,while video is streaming.**

**•Textfield limitation for password Setting page.**

**•New password creating in password Setting page.**

**•Display image in Gallery page.**

**•Display image in Home page.**

**•Sidebar function for all pages.**

## 7.2. Android

### 7.2.1. Test plan

The software test Plan has been done following the software requirement specification document. Team Fruitcakes decided to test interface and system. First, we tested the interface to see if the functions and text field limitations worked correctly. Then, we tested the core functions of the software, for example testing video streaming, login function, password changing, etc. To test the android application, 10 test cases have been created.

The test results should meet criteria that is stated in each test case. Each test case contains the test case name, number, description, pre-condition, action, expected result and the actual result. In addition, it is important that the interface has the correct result from the server and meets the requirement.

Software testing in this project is done as integration testing. The integration testing is any type of software testing that seeks to verify in the interface between components against a software design. Because this software project is not on a large scale, we think this type of testing is suited for the project.

The tester should be a person other than the ones who created the code. A main reason why the person who writes the code shouldn’t also be the tester is because they could be bias on testing easily. Using a different person is one solution that can solve this problem. Also we assume that the tester is a genuine user of PieChecker.

### 7.2.2. Test result

Since the video streaming which is the functional requirement is not done, we decided to not do the testing because it is not necessary to test other components which are not the core function. However the test cases for Android application have been created. We have learned the testing document and the testing process.

In conclusion, even the software testing is not done, but we have improved our knowledge of software testing, understanding testing documentation, type of software testing and the relation of software testing and requirement specifications.

## 7.3 Website

# 8. Conclusion

**scrum methodology-**

* learned how to use scrum to manage project
  + more meetings would be efficient to help follow up on tasks and to provide extra help to anyone who needs it.
  + scrum master should keep close attention to tasks being completed and such in order to keep the burndown chart up to date.
  + we planned exact number of sprints and tasks out in backlog. however we didn’t decide beforehand which tasks belong to which sprint (order of tasks is very important)
* planned to assign task to each member for each sprint. review every two weeks. however, we found that many problems occur. one task is dependant of another task and if one isnt completed, the other can’t be either thus creating technical debt that accumulates through future sprints. (domino effect)
* reasons for tasks being incomplete are members being sick, traveling, priorities elsewhere, MIA, experience and skill level for certain tasks may vary (need more time to do research)
* scrum master is a great position to keep everyone focused and make sure no outside issues affect the team. however there is no authority in the job. there are no punishments for incomplete work or missing scheduled meetings.
* should manage time more efficiently (plan some cushion time for possible problems that could happen so that you wont need to rush at the end of the development period.)

**overall project-**

* ssd design and final outcome of the project are not exactly the same but the flow follows the design as much as possible.
* some difficulties didn’t allow us to do things a particular way however we have come up with solutions that still allow us to reach the goal for the project.
* some minor features we initially decided on didn’t make it to the final product due to time management or priority level wasn’t significant.
* could not reach all requirements and some requirements were met but not in the way we planned.
* realize the design and the use cases are very important tools in developing the project and should be fully understood before beginning.
* give priority to each task so things are done in order. we had the problem of doing tasks we wanted then realizing another task had to be finished before this one could function.

**technical development-**

* our project had so many platforms to focus on. we realized when it was too late that we needed to find a different program to satisfy all of them. we also needed more people to help in certain areas than expected. one platform would have been suffice.
* entire backend of project had to change during the last two weeks of allotted time.
* a lot of time had to be spent researching how to use Angular Javascript, CSS and HTML before we could actually start building our website. then realized Angular was a bit too advanced for our skill level so we had to push forward using only HTML, CSS, and Javascript.

# 9. Appendix

## 9.1 Contribution

**Luuk van Egeraat:**

**Marcus Näslund:**

* Created a function in gallery page
* Purchased electronic components
* Created first circuit for the temperature probe
* Wrote some hardware documentation

**Michael Warne:**

* Android development
* Android documentation
* Voice over for video presentation

**Philip Malm:**

* iOS development
* iOS documentation
* Brought screen and cables for the pi
* Test plan for website testing

**Saipirun Sanprom:**

* Documentation:
  + Design Document
  + Final documentation(documentation plan, introduction, testing)
  + documentation integration
  + burndown charts
  + sprint review document
* Creating test cases.
* Creating initial website interface and adding functions to component.
* Creating function for the components in the previous designed website.
* a part of video presentation making.

**Shan Jiang:**

* Hardware:
  + Purchase part component from China
  + Combine casing with holder
* Website:
  + HTML5&CSS3 development for initial website
  + Implement for JQuery
* Presentation:
  + Record videos
  + Editing videos
  + Product Casing
* Documentation:
  + Test plan for IOS app

**Nicole Musco:**

* Documentation:
  + Final documentation (Website Application, Testing, Conclusion)
  + Design Document
* Creating design and user flow for applications and website
  + visual design using photoshop tool
* Creating initial website interface (video page, unused login page)
* Contributed in the making of the presentation video ( writing the script, recording ideas, recording, creating photoshop images for video clip)

**9.2 Test Cases**

**9.2.1 Website Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TestCase: Stream Video on the webpage** |  |  |  |  |
| **Test Case#: WP001** |  | **Test Case Name: Stream video on the web page** |  |  |
| **Design by: Saipirun Sanprom** |  | **Design date: 6 May 2014** |  |  |
| **Tester:** | **Philip Malm** |  |  |  |
| **Test date:** |  |  |  |  |
| **Description: Test streaming video showing in web page** |  |  |  |  |
|  |  |  |  |  |
| **Pre-Condition:** |  |  |  |  |
| **The pieChecker is on and standby in front of the oven.** |  |  |  |  |
| **The baker opened web page and already log in to the homepage as initial state.** |  |  |  |  |
| **the pieChecker connect to internet through WIFI** |  |  |  |  |
|  |  |  |  |  |
| **Step** | **Action** | **Expected Result** | **Pass/Fail** | **comment** |
| **1** | **Click "My VIDEO" button** | **The system change to display the MY VIDEO Page.** |  |  |
| **2** | **Point mouse at "Select Device" drop down menu** | **Drop down menu is shown under the word.** |  |  |
| **3** | **Click mouse on wanted device** | **the video is shown on screen from the selected device.** |  |  |
| **4** | **check time counter** | **the time counter should start in the same time as video starting** |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**9.2.2 Android Test Cases**

**9.2.3 IOS Test Cases**