**ECE-4773 Electrical Engineering Capstone Design**

**Electrical and Computer Engineering**

**University of Oklahoma**

**Norman Campus**

OU Rugby Score Board

Software Design Document

**April 24th, 2019**

# Abstract

The objective of this software is to establish a connection between the user (phone) and the hardware (LED Score Board display). The Apache server is installed and stored inside Raspberry Pi. The user interfaces with the server through an HTML web-page and PHP code to update the values in the database using SQL. Then a Python script is kept running in a loop to retrieve data from the database and outputs the data to the LED display (Score, Time). The software can be run on any device that is able to connect to a Wi-Fi network and open a web-page, for example: Laptop, Android and IOS.

Table of Contents

[Abstract i](#_Toc530561315)

[Table of Contents ii](#_Toc530561316)

[List of Tables iii](#_Toc530561317)

[List of Figures iv](#_Toc530561318)

[Problem Description 5](#_Toc530561319)

[Analysis 5](#_Toc530561320)

[Program Version 6](#_Toc530561321)

[Client-base 6](#_Toc530561322)

[Programs 7](#_Toc530561323)

[1. MySQL: 7](#_Toc530561324)

[2. PHP: 8](#_Toc530561325)

[3. Python: 8](#_Toc530561326)

[4. Teensy: 8](#_Toc530561327)

[Instructions 8](#_Toc530561328)

[1. Logins and Passwords: 8](#_Toc530561329)

[2. Connect WIFI network: 8](#_Toc530561330)

[3. Controlling Score Board: 8](#_Toc530561331)

[4. Powering ON/OFF: 8](#_Toc530561332)

[Troubleshoot 9](#_Toc530561333)

[1. Entering Monitoring mode: 9](#_Toc530561334)

[2. Reinstall the OS: 9](#_Toc530561335)

[3. Reinstall Other Software: 9](#_Toc530561336)

**List of Tables**

Table 1 Bluetooth vs. Wi-Fi 5

Table 2 OURugby Database and tables 7

**List of Figures**

Figure 1 Overall Flow Diagram 5

Figure 2 Welcome Page 6

Figure 3 Spectator View 6

Figure 4 Operator View 7

Figure 5 Etcher Interface 9

Figure 6 Raspberry Pi Terminal Location 10

Figure 7 phpMyAdmin page 10

Figure 8 phpMyAdmin - Change Password 11

Figure 9 PiRun.sh - RunScoreBoard.py - ScoreBoardTeensy Folder 12

Figure 10 Arduino Interface - Verify - Upload 12

# 

# Problem Description

The OU Rugby team needs a new scoreboard. The scoreboard is needed to be controlled remotely using mobile phone (Android and IOS).

# Analysis

To control the board remotely, two connection protocols are to be evaluated at the beginning: Bluetooth and Wi-Fi.

Table 1 Bluetooth vs. Wi-Fi

|  |  |  |
| --- | --- | --- |
| Characteristics | **Bluetooth** | **Wi-Fi** |
| **Range** | Short ( <5m) | Long (>10m) |
| **Component** | Less | Additional router |
| **Price** | Cheaper | More expensive |
| **Complexity** | App-based  (Android **vs** IOS) | Web-based  (Android **and** IOS) |

According to Table 1, Wi-Fi has more advantages over Bluetooth due to the effective range of the connection and the compatible of the programs and the device the program is running on. Since both Android and IOS devices can read a web-page, it is more convenient to write a web-based so both devices can read instead of writing an app on one platform and convert it to the other platform later. In order to operate a web-page, there are two primary bases to be focused on: Client-base and Server-base. The client consists of any elements that helps interfacing with the user. The server consists of any elements that helps manipulating the inputs from user, storing the inputs from user and interfacing with the hardware. The client-base will include the codes for HTML, CSS, and JavaScript. The server-base will include the codes for PHP, SQL and Python. All elements and their relationship are shown in the following figure.

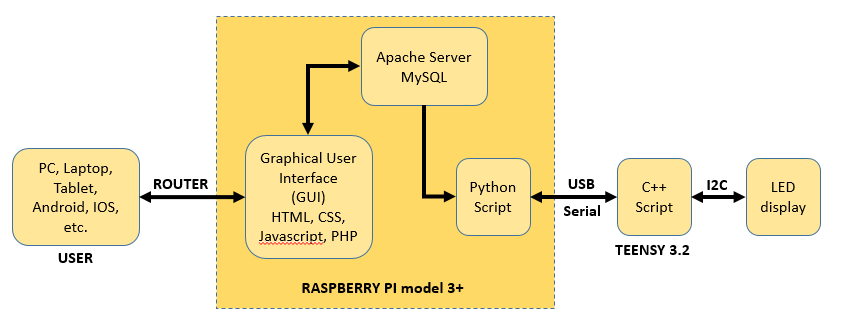


Figure 1 Overall Flow Diagram

# Program Version

* Apache 2.4.25 (Raspbian)
* PHP 7
* Python 3
* Arduino IDE 1.8.7
* Teensyduino 1.44

# Client-base

The GUI of the web-page is given below. The structure or frame of the web-page is written in HTML using CSS as styling. JavaScript will let the score be modified when any of the increment buttons are pressed. Also the JavaScript allows the appearance of the GUI to change depending on the user input. For example, the corresponding period will light up when it is chosen or the “Play” button will automatically change to “Pause” button.

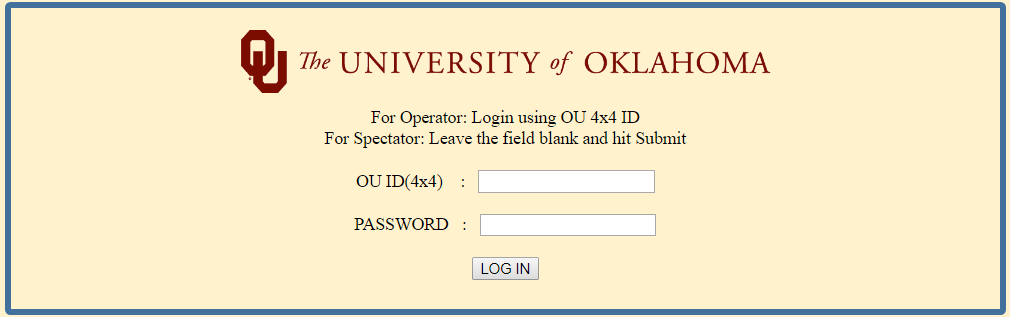


Figure 2 Welcome Page

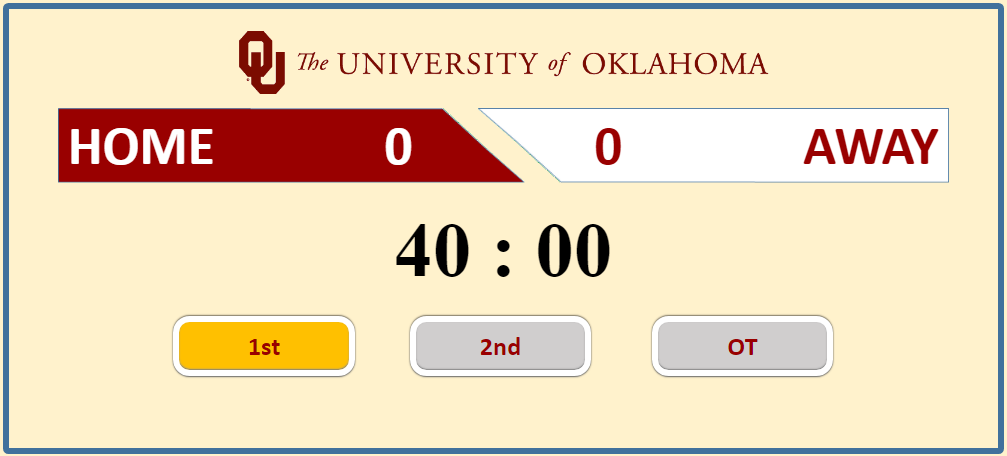


Figure 3 Spectator View



Figure 4 Operator View

# Programs

1. **MySQL**:

Whenever the user gives the inputs or the clock is running, all the inputs are saved and updated to the server. In order to store all the inputs, a database is created under MySQL server called “OURugby”. Inside the database, there are 3 tables: Score, Time and Button. The “Score” table stores the score values of the home and away team. The “Time” table stores the current time of the current period in minute and second so they can be called at any time. The “Button” table stores the current state of the program or the last button pushed.

Table 2 OURugby Database and tables

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Score** | | | | | | | |
| **ID** | | **TEAM** | | **POINT** | |  | |
| 1 | | Home | | 0 | |  | |
| 2 | | Away | | 0 | |  | |
|  | | | | | | | |
| **Time** | | | | | | | |
| **ID** | **PERIOD** | | **MINUTE** | | **SECOND** | | **STATUS** |
| 1 | 1st | | 40 | | 0 | | 1 |
| 2 | 2nd | | 40 | | 0 | | 0 |
| 3 | OT | | 10 | | 0 | | 0 |
|  | | | | | | | |
| **Button** | | | | | | | |
| **ID** | | **NAME** | | **STATUS** | |  | |
| 1 | | Play | | 0 | |  | |
| 2 | | Pause | | 0 | |  | |
| 3 | | Reset | | 0 | |  | |
| 4 | | New | | 1 | |  | |
| **Operator** | | | | | | | |
| **ID** | | **LOGIN** | | **PASSPHRASE** | |  | |
| 1 | | admin | | OURugbyTeam | |  | |
| 2 | |  | |  | |  | |

**NOTE**: The Raspberry Pi Apache Server is **CASE-SENSITIVE**.

1. **PHP**:

At the beginning of the program or when someone hits “Refresh”, the page will automatically return to the last page it was on. The PHP codes will connect to the MySQL server and pull all the scores off the server and output to web-page. The codes will also retrieve the last active period and the corresponding time left. Every time the user changes the score or the clock changes, PHP will update the database with new values.

1. **Python**:

A python script is used to gather data from the server including Score and Time. Then the information is sent to Teensy microprocessor serially using a USB port.

1. **Teensy**:

The teensy uses C++ compiler. The script takes information from the Raspberry Pi and converts to output to the LEDs combinations as binary numbers. These numbers are sent to the I/O chip expander using I2C protocol in order to display the numbers on the LEDs.

# Instructions

1. **Logins and Passwords**:

Raspberry Pi Device Login Info

Username: pi

Password: OURugbyTeam

Raspberry Pi phpmyadmin Login Info

Username: root

Password: OURugbyTeam

Raspberry Pi WIFI networks Login Info

Network Names: OURugby, OURugby1, OURugby3

Password: Rugby1974

Raspberry Pi default webpage address

192.168.1.127 (if connected to OURugby)

192.168.1.245 (if connected to OURugby1)

192.168.1.252 (if connected to OURugby3)

Raspberry Pi default local server address

127.0.0.1

Operator Mode default admin Login info

Username: admin

Password: OURugbyTeam

1. **Connect WIFI network**:

Using the credentials above to connect into Raspberry Pi WIFI network. Open the Raspberry Pi default webpage address to check connection. If the welcome page is opened, the device is connected to the network

1. **Controlling Score Board**:

On the welcome page, log in using the default admin info or an OU 4x4 that has been granted access.

Refer to Figure 3, using corresponding button to add/subtract score of the Home or Away Team. Refer to scoring instruction at the bottom of the page for Score Definition.

1. **Powering ON/OFF**:

Turn on the system by plugging the main power plug into outlet. Wait for about 30 seconds for the system to start up. Turn off the system by taking the main power plug out of the outlet.

# Troubleshoot

1. **Reinstall the OS**:

In case the SD card is corrupted and the Raspberry Pi won’t boot properly, the OS and all the programs must be reinstalled. Things to have before performing the reinstallation:

* Raspbian stretch with desktop and recommended software (OS image) (<https://www.raspberrypi.org/downloads/raspbian/>)
* Etcher (OS Image installer) (<https://www.balena.io/etcher/>)
* SD card reader

Download the Raspbian stretch and Etcher Software. Unzip the Raspbian stretch file.

Install the Etcher software. Then Open the Etcher Software.

Insert the SD card into SD card reader and plug the SD card reader into the device which has Etcher software

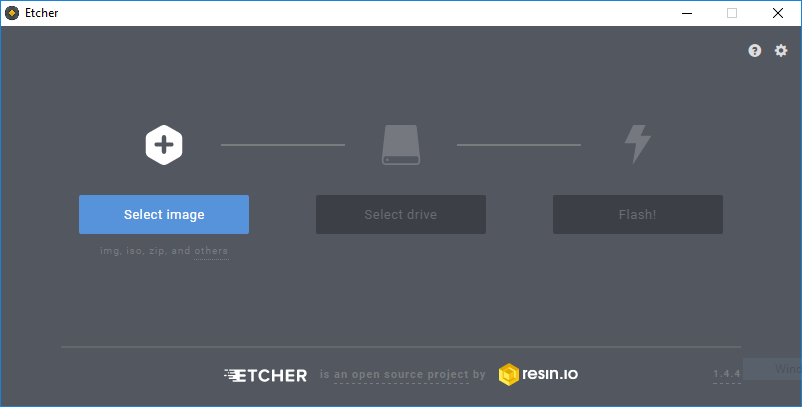


Figure 5 Etcher Interface

Click “Select Image” and locate the image file in the unzipped Raspbian Stretch folder.

Click “select Drive” and select the drive of the SD card.

Hit “Flash” and wait for the process to finish.

Safely remove the SD card when it is finish.

1. **Reinstall Other Software**:

Refer to the Program Version section for a list of programs needed to control the Scoreboard. Using the Raspberry Pi Terminal to install the software. See the README.txt for more explicit instructions.

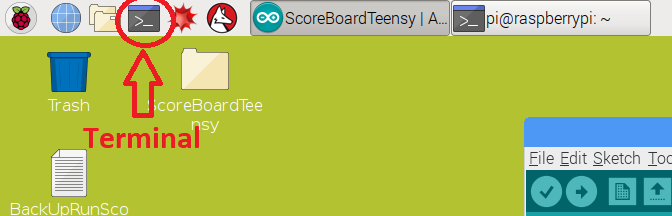


Figure 6 Raspberry Pi Terminal Location

Once all the software are installed. Open the browser and goes to 127.0.0.1/phpmyadmin.

Log in with username “root”.

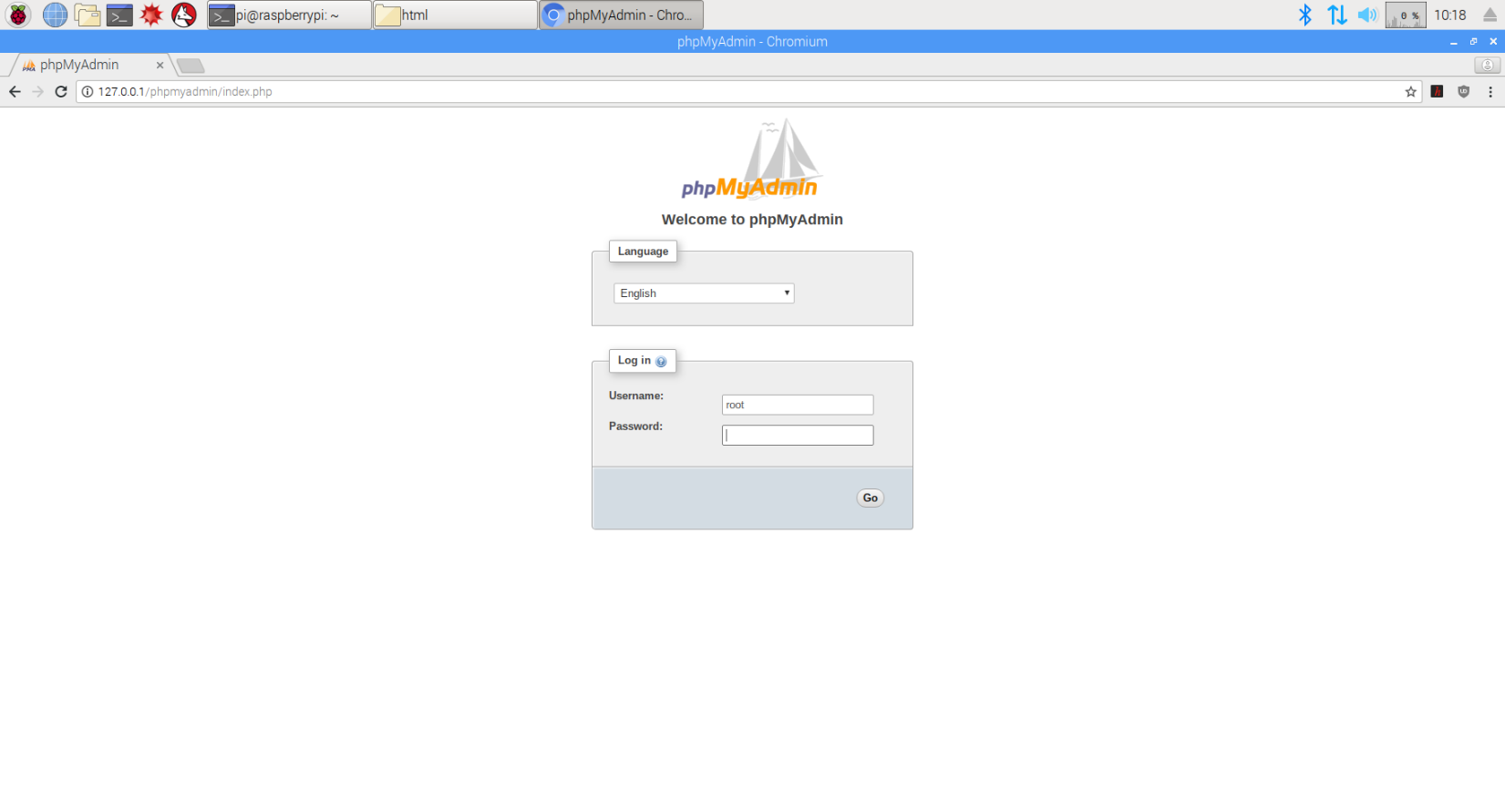


Figure 7 phpMyAdmin page

The first time logging in won’t require password. Next, change the password of the root user to the one indicated above in the Login and Passwords section.

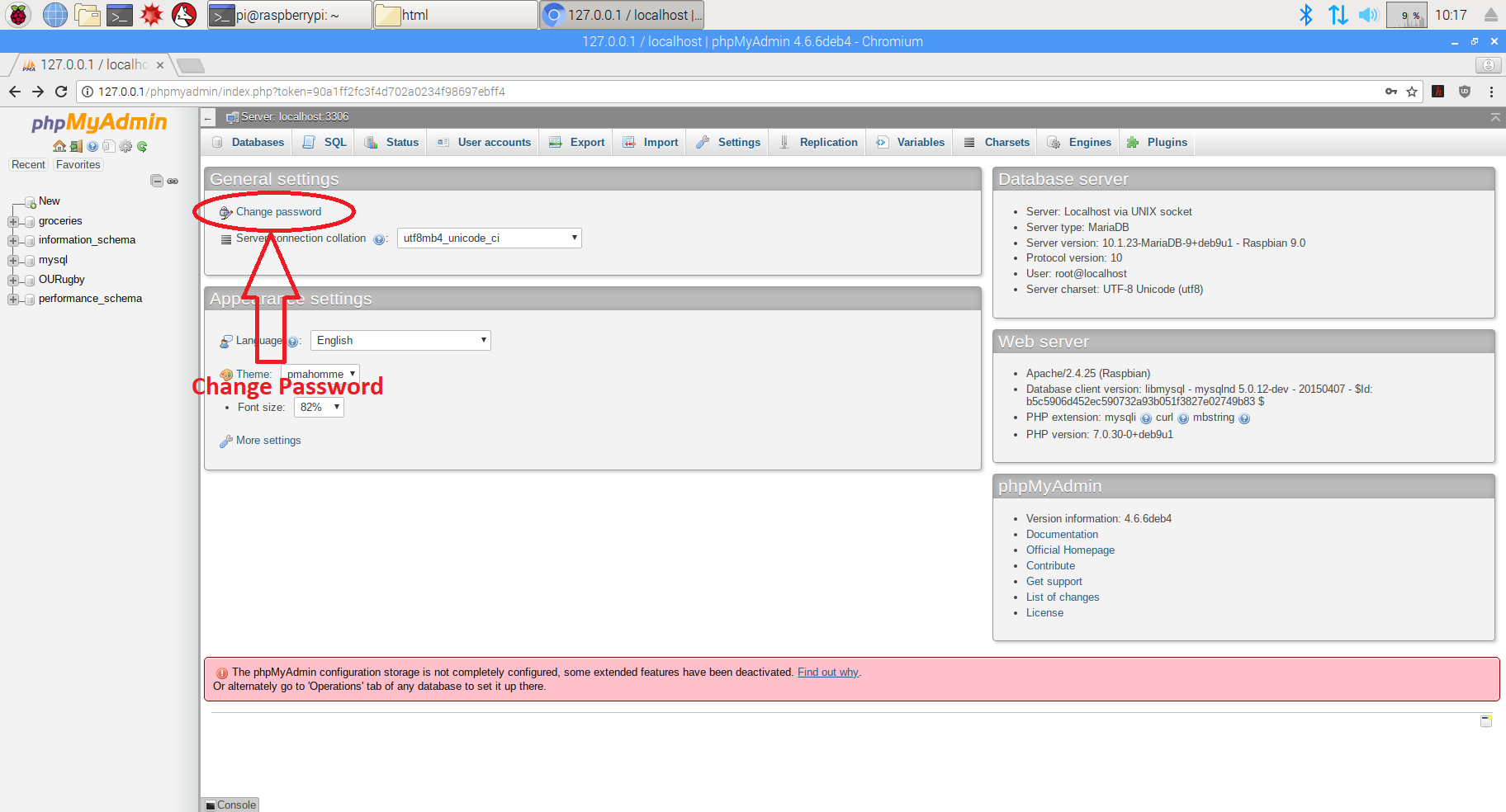


Figure 8 phpMyAdmin - Change Password

Create a database named OURugby. Create 3 tables as indicated in MySQL section.

Refer to “***Other Commands***” document for creating a Start-up script and setting the Static IP address.

Plug in the Back Up thumb drive and locate the Server Folder. Copy every file and folder in the “***Server and Programs***” Folder into the Raspberry Pi folder ***“/var/www/html***”

Copy ***RunScoreBoard.py***, ***PiRun.sh*** and ***ScoreBoardTeensy*** Folder into the Raspberry Pi Desktop.

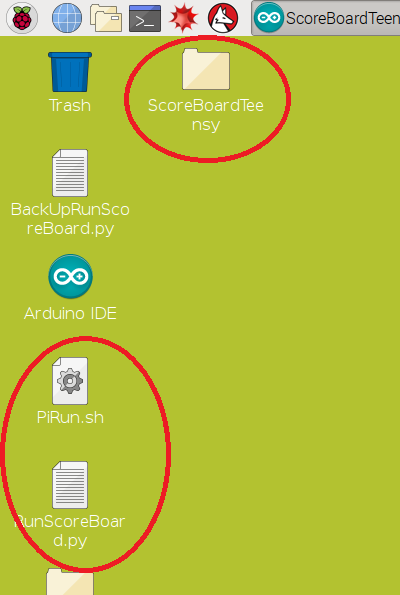


Figure 9 PiRun.sh - RunScoreBoard.py - ScoreBoardTeensy Folder

Run Arduino IDE and Open the file ScoreBoardTeensy.ino inside the ScoreBoardTeensy Folder.

Connect Teensy Board to Raspberry Pi using the USB to microUSB cable.

Click Verify. Once it is verified. Click Upload to upload the program into Teensy Board.

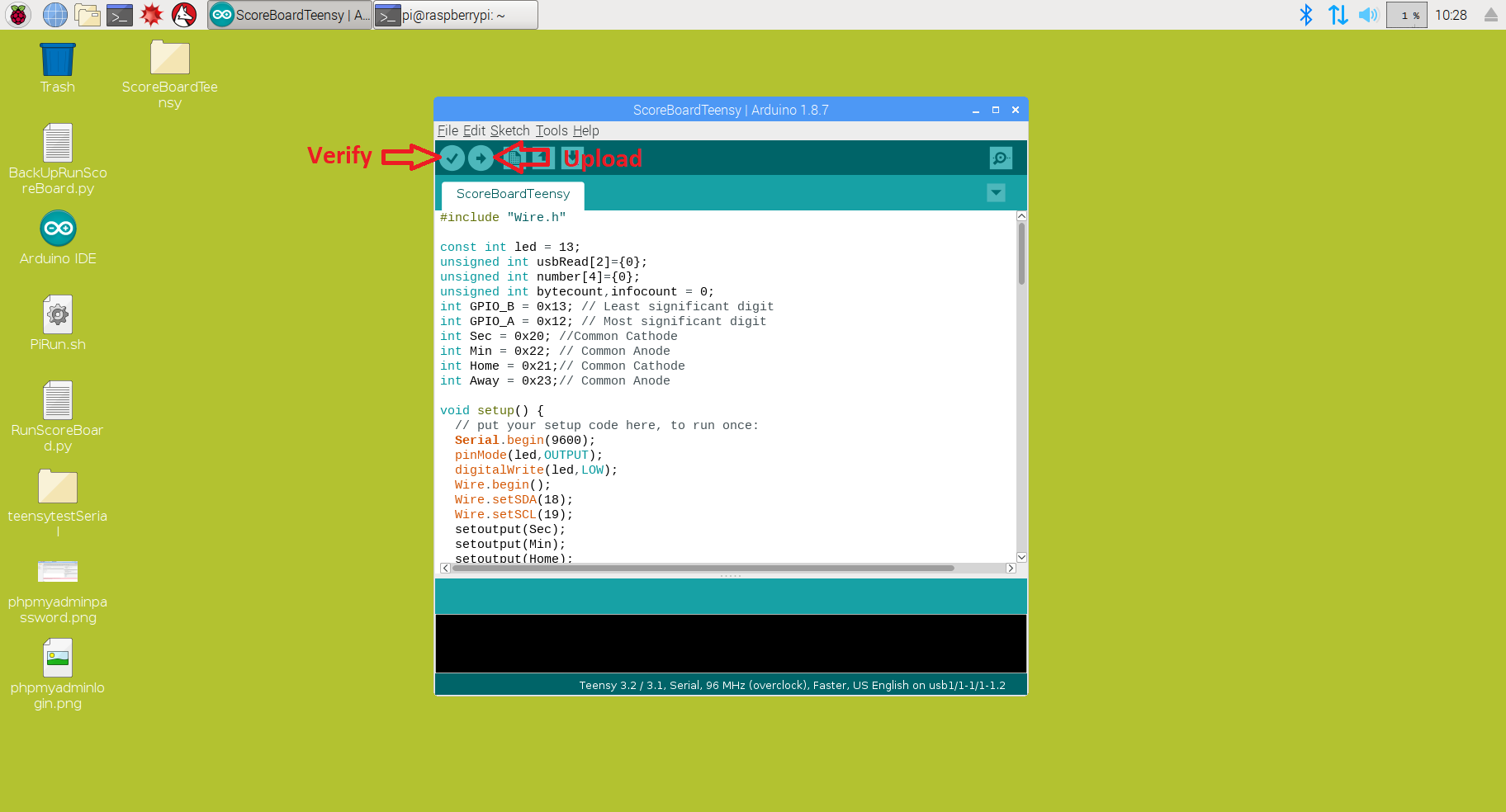


Figure 10 Arduino Interface - Verify - Upload