By Joshua Brown

Abstract

Junior Bible Quiz (JBQ) is fun way for kids to get interested in memorizing the bible by competing against other teams. The world of JBQ has been around for about 50 years. Even though smartphones and other mobile devices have been around for many years now, most of the these competitions are largely done on paper still. QuizMate is a system designed to modernize JBQ by digitizing the process of scorekeeping and communication at these meets.

Table of Contents

Statement of Work

QuizMate will be developed by Joshua Brown until there is a working prototype. A working prototype will be declared when the product can host a quiz meet successfully. Once the working prototype is complete the product will be turned over to the Assemblies of God Church (AoG) IT department to complete and deploy nationally.

Statement of Problem

Currently, most quizzing districts across the country use paper score sheets to keep track of a match. The paper sheets are sometimes hard to read, math errors are common, and the details of a match cannot be aggregated by a computer. Awards ceremonies in the past have taken as long as 3 hours to get awards out to the quizzers. Most campuses do not yet have Wifi capabilities. Most teams do not have a budget to purchase laptops to use for a new scoring system. There is very little tech support in the community to support many different machines for a meet.

Statement of Goals

- The new system shall allow scorekeepers to score directly from the room.
- The new system shall allow for scores to be feed into the main reporting system live.
- The new system shall work off of cell towers.
- The new system shall be designed to have a scoreboard to give awards to the quizzers.
- The new system shall be web based and hardware agnostic.
- The new system shall be developed privately as to not confuse ownership of the product once complete.

Statement of Scope

The scope of this work is to create a product that can be turned over to the main IT department at AoG. The product was to serve the Arizona District of quizzing for Junior Bible Quiz (JBQ), but be able to be used in any district across the country, assuming there own install of the product.

List of Roles

Joshua Brown (Volunteer) = JB

Kenzie Brown (Volunteer) = KB

Heather Brown (Volunteer) = HB

Faith Brown (Volunteer) = FB

Jesse Baumgartner (Arizona District Central Region Coordinator) = JeB

Kathy Allen (Former District Central Coordinator) = KA

Scott Berkey (National JBQ President) = SB

Todd Smith (Volunteer) = TS

Sherrie Muscari (Arizona District Central Region Assistant Coordinator)= SM

Ryan Burbank (Arizona District South Region Coordinator)= RB

Chris Royer (Volunteer) = CR

Vernon Miller (Volunteer Joined November 2015) = VM

Developers	JB, VM
Testers	JB, KB, HB, JeB, KA, TS, CR, VM
Stakeholders	SM, RB, JeB, SB, KA
Project Manager	JB

Risk Management

- Hours required to complete project may be too high and therefore the project will be abandoned.
- The system may not operate with consistent data channels over the cell towers.
- The system may require coding that beyond the talent levels of the volunteer programmers. This may require outside help compromising the independence goal.

- A competitive system may be built in the time it takes to produce the system therefore causing the project team to abandon development.
- Tools required for the development of the system may prove to be too costly.

Cost Management

- The Arizona District office has agreed to pay for domain name services for two domain names.
- The Arizona District office has agreed to pay for web hosting for one website.
- All labor will be volunteer.
- All software will be open-source and therefore free.
- All required tools will be open-source or donated by the volunteers

Cost Itemized

Description	Qty	Annual Cost
Domain name registration for AZJBQ.org.	1	18.17
Domain name registration for AZJBQ.com.	1	15.17
Web hosting (Volunteered)	1	0
Total		**33.34**

Project Team Coordination Plan

Communication

A joint team gmail account will facilitate all communication between team members and the quiz community. The members of the leadership counsel will be each be responsible for checking the account for new message relevant to them. Google Doc's will be used to communicate development information, testing plans, passwords, database schema, and other relevant information.

DevOps Plan

Development Platform

Early stages

The earliest development began as a virtual machine on my home PC. I installed VirtualBox on my Windows Vista computer and installed Centos as a VM to run the web server for the application. I installed Netbeans with PHP plugins to provide the IDE platform that I needed. I installed Netbeans on my Vista computer and configured my IDE to save via SFTP to the Centos server.

Post Prototype

After getting a viable prototype running we made the decision to move the production code to the Godaddy server. Netbeans on my Vista computer is still the main programing platform. The only change was to point the SFTP target to the FTP server at Godaddy.

The Graphic Icon

The graphic work was done by Jesse Baumgartner. He originally did 4 images, and he asked which one I thought was the best. I said, "let's show them to the quizzers and let them pick." So, at the first meet of the year Jesse asked the kids to cheer for the one they liked the most. This image is the one they picked.

Post Godaddy

Performance issues at Godaddy prompted a change in the hosting provider. With some help from the new hosting provider we moved the code from the Godaddy server to ValleyNerds.com. At this time we were still using the same Netbeans software for development.

Post System Crash

Following a crash of my home system, I installed the Mint Linux operating system. After trying to install Netbeans on my new OS, I found that Java was the only language that had been ported to Mint. I decided to move the code to an online IDE called C9.io. A free SaaS product for development. It has an experimental SFTP mount feature that works most of the time. The account and usage for this project falls under the free tier of C9 or Cloud9.

The Plan

Development and documentation

The plan was to develop a prototype and have the JBQ leaders test and provide feedback on functionality. Main basic functions of the system were considered a given set of features and now documentation was needed. As requests for new features and enhancements came in I recorded those in a to-do list style log. As many of them started to come in I would set a priority number to rank them. I further classify the work to be done as a bug or desireable feature. As the development began to consume my time I needed to rely more and more on testers. To convey to the testers what was needed to be tested and what could be ignored, I added columns into the feature tracking sheet to tell the testers whether something needed to be tested or not and on what devices. In the beginning it was indicated that many different devices behaved so differently that we needed to test many of them. However, we didn't own all the different devices and so I put columns into the spreadsheet to have testers check the code on specific devices. Those testers with obscure devices would test those first and mark them off on the list or write notes of failures. I tried to document much of the system, such as databases, tables, functions, modules, etc., but I found that it took way too long to document effectively and the info was changing frequently. I also used PHPMyAdmin to document the databases and tables, but without an automatic routine to do this documentation it just becomes too laborsome. I spent some time looking for something that would document the code for me, but never had any luck.

Godaddy mistake

I need to make a special note here about Godaddy hosting. When we first picked a hosting provider we went with Godaddy, because it was well known and local. We put the code there and ran the system for the first year from there. The code was largely experimental through the first meet. By the second meet we felt it was ready to try out in the live rooms. We used paper as a fall back plan. In the first meet 6 out of 12 rooms scored perfectly, but the other 6 rooms didn't score well at all. At that meet there was no WiFi available. At the third meet we had Wifi, and we're in the basement at the facility. Since, we were in the basement we had no cell reception at all. While down there we had a horrible time. The system was extremely slow.

We went home and started testing to see if we could replicate the conditions. We tested many different conditions, but could not replicate the problem. I spent about 6 months working on the issue. I assumed that the issue was my code, so I ripped it apart many different ways to trying to isolate the problem. I asked for help from many of my engineering friends and we finally determined that it was any cell phone or tablet device sharing the same WiFi, connection. We ran tests and found that as little as two of those devices on one WiFi would cause the site to run slowly for both devices. However, we could put 20 or 30 laptops on the same WiFi and had no issues.

A friend of mine said, "Just copy the code to another hosting provider real fast and check how it performs there." At first I thought there was no way it would make a difference. I reasoned that GoDaddy couldn't be the problem; they host hundreds of thousands of websites. I couldn't be the only one with this problem. At this point I had no other ideas and, "What could it hurt?" So, with the help of a friend who owns the ValleyNerds.com hosting company, we moved the code to his server and tested from there. We no longer had any problems. So, we immediately moved the entire site and have not had any problems with performance since.

Code

The code is written in PHP, Javascript, and HTML. MySQL provides the database backend. MySQLi modules are used to interface with the databases.

Build

Code is changed on the test files and simply saved to the server. The server maintains test areas and therefore all the code is immediately built on the main server.

Test

Testing is done in the testing area on the main server. Testers must authenticate as a tester to be allowed in those areas. This distinguishes, administrator functionality from the testing process. A tester is granted access to the development spreadsheet. From there they will know what features to test and which to ignore. This spreadsheet also has test scenarios that need to be ran. Since we have so many testers it was common for them to be testing the same features. For this reason a tester is supposed to mark off each scenario as they go. If the tester finds a bug they are to report it in the spreadsheet next to the feature that is not working as expected. I take that feedback and adjust code and notify testers of the next round of testing. The testers are given two weeks for the first round of testing and then another week following any bug repair work.

Release/Deploy/Operate

Once the code has been tested and is ready to be released to production. I take a spreadsheet that tracks the deployment. As I work on different modules I update the code deployment spreadsheet so that I know if that code needs to be promoted to the training area as well as the production areas. As I overwrite the code I mark off that module as I go. Once those modules have been updated they are officially in production and ready to be used.

Effort Hours Summary

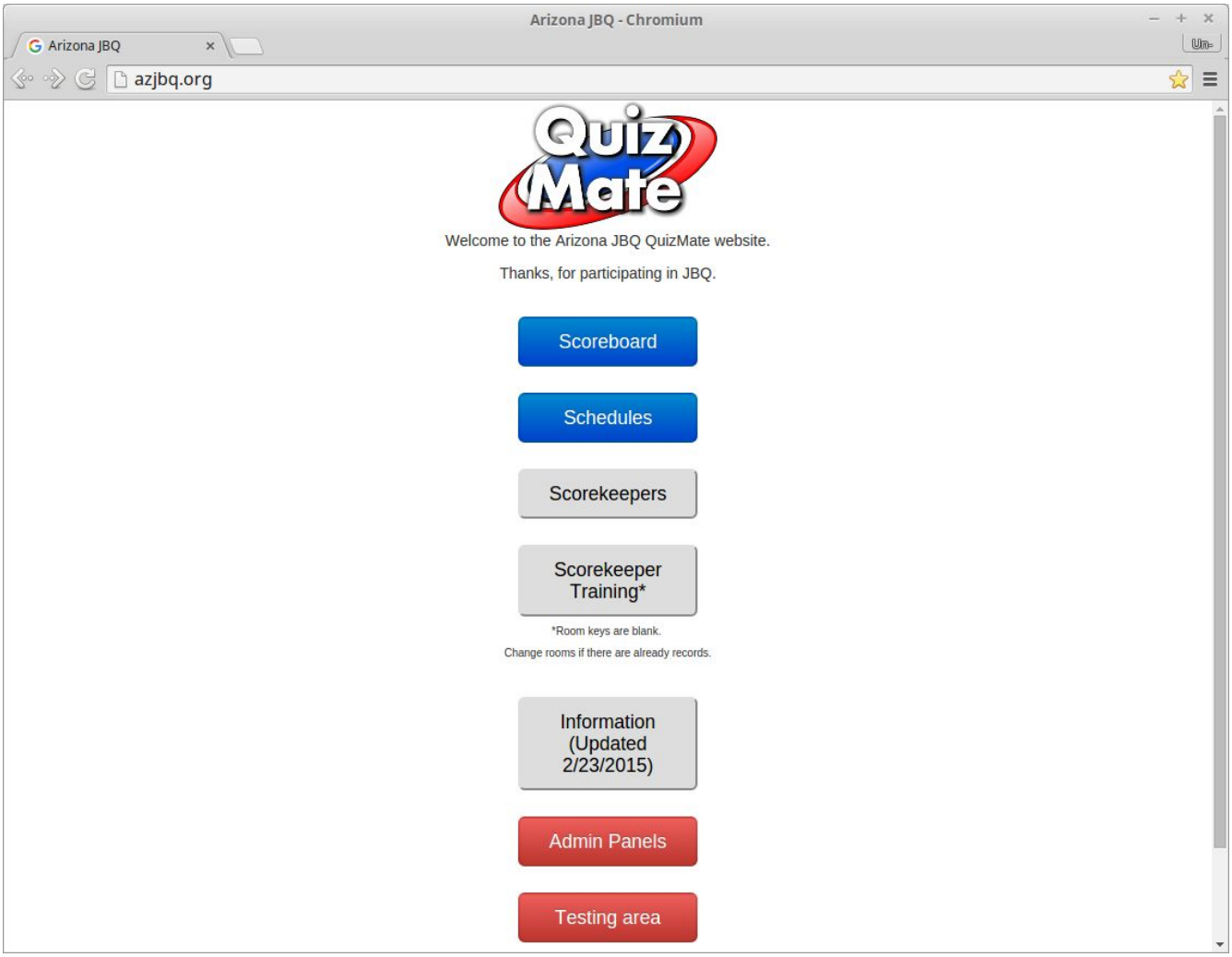
Person	**Effort Weeks**	**Avg Hrs/wk**	**Development Hours**	**Testing Hours**
Joshua Brown	156	10	1040	520
Kenzie Brown	100	4	20	380
Heather Brown	50	2	0	100
Faith Brown	60	2	0	120
Jesse Baumgartner	60	2	0	120
Kathy Allen	30	1	0	30
Scott Berkey	10	1	0	10
Todd Smith	30	1	0	30
Vernon Miller	4	2	8	0
Total	**500**	**25**	**1068**	**1310**

Project Work

Screen Shots

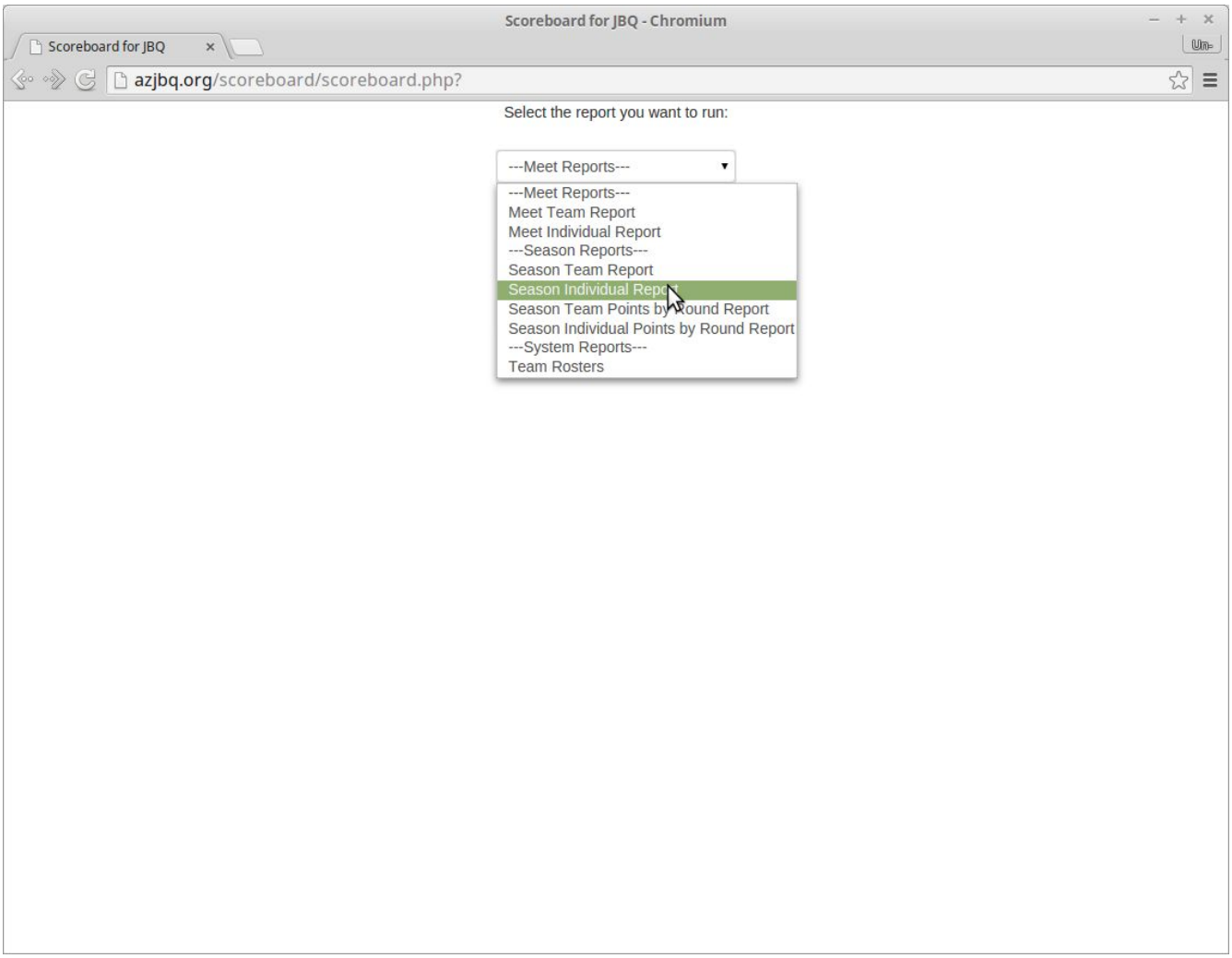
Main screen

(Anyone on the Internet can view)



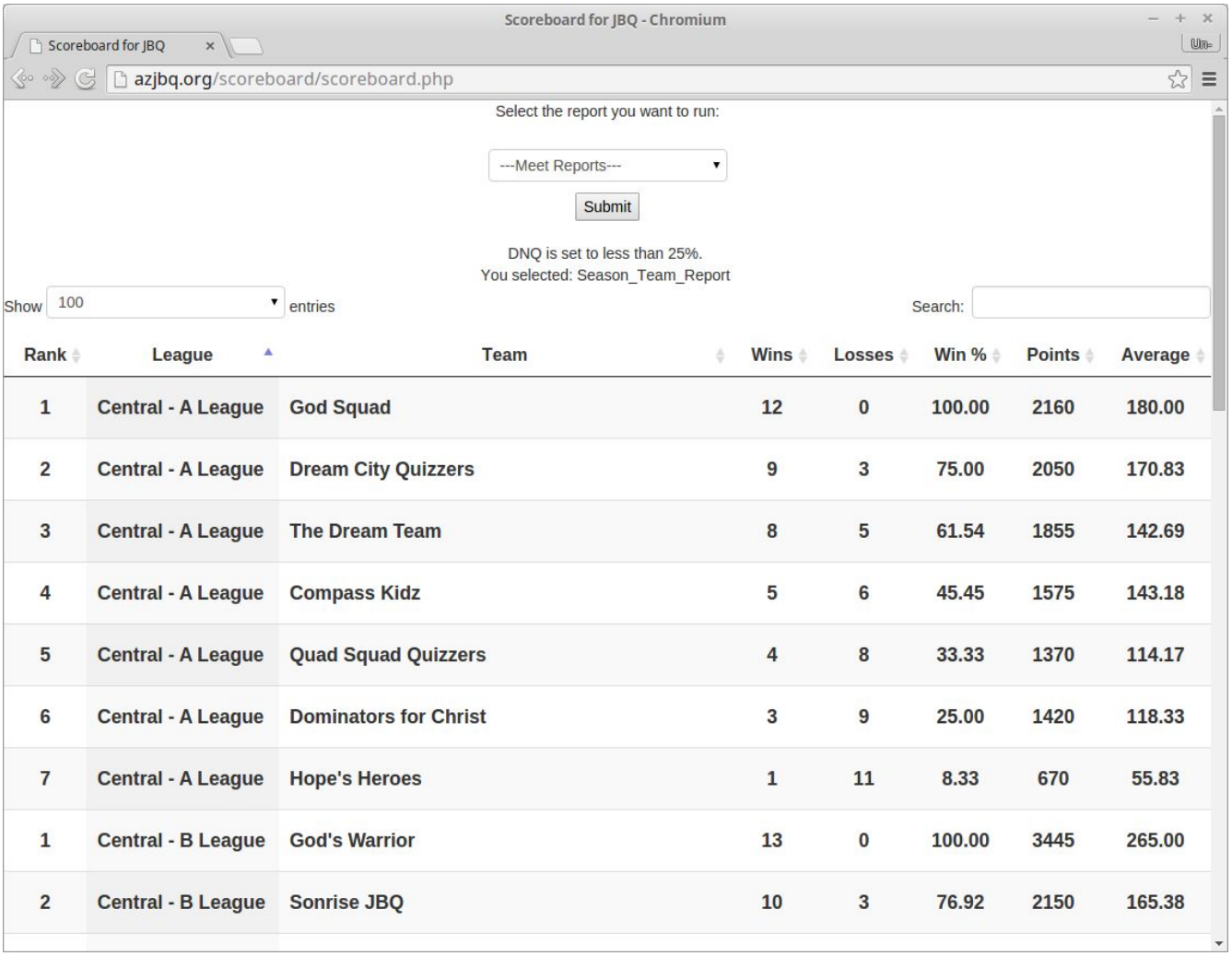
Scoreboard (Selecting a report)

(Must be at least a spectator to view)



Scoreboard (Selecting the Season_Team_Report report)

(Must be at least a spectator to view)

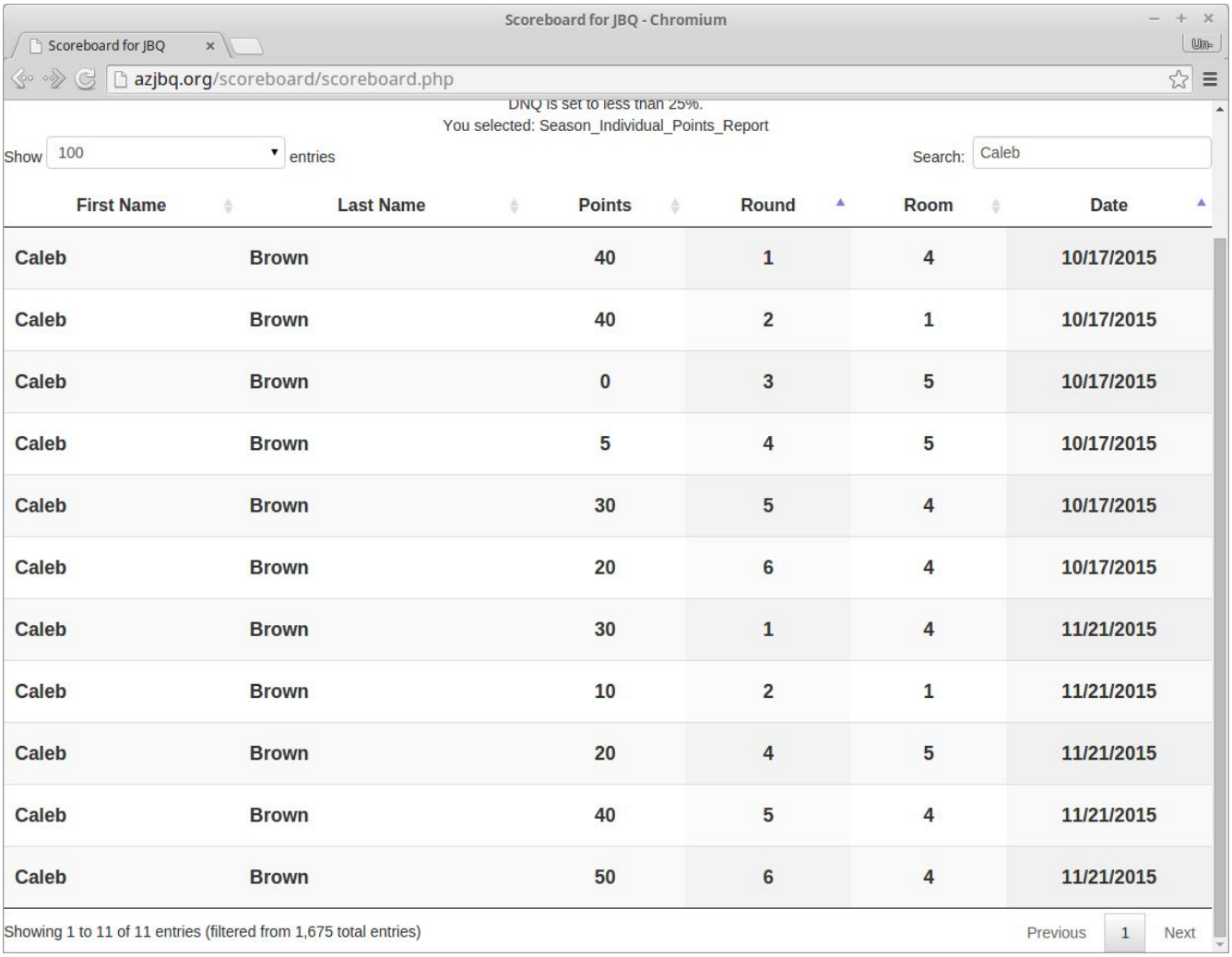


Rank	League	Team	Wins	Losses	Win %	Points	Average
1	Central - A League	God Squad	12	0	100.00	2160	180.00
2	Central - A League	Dream City Quizzers	9	3	75.00	2050	170.83
3	Central - A League	The Dream Team	8	5	61.54	1855	142.69
4	Central - A League	Compass Kidz	5	6	45.45	1575	143.18
5	Central - A League	Quad Squad Quizzers	4	8	33.33	1370	114.17
6	Central - A League	Dominators for Christ	3	9	25.00	1420	118.33
7	Central - A League	Hope's Heroes	1	11	8.33	670	55.83
1	Central - B League	God's Warrior	13	0	100.00	3445	265.00
2	Central - B League	Sonrise JBQ	10	3	76.92	2150	165.38

Scoreboard (Selecting the Season_Individual_Points_Report report)

(Must be at least a spectator to view)

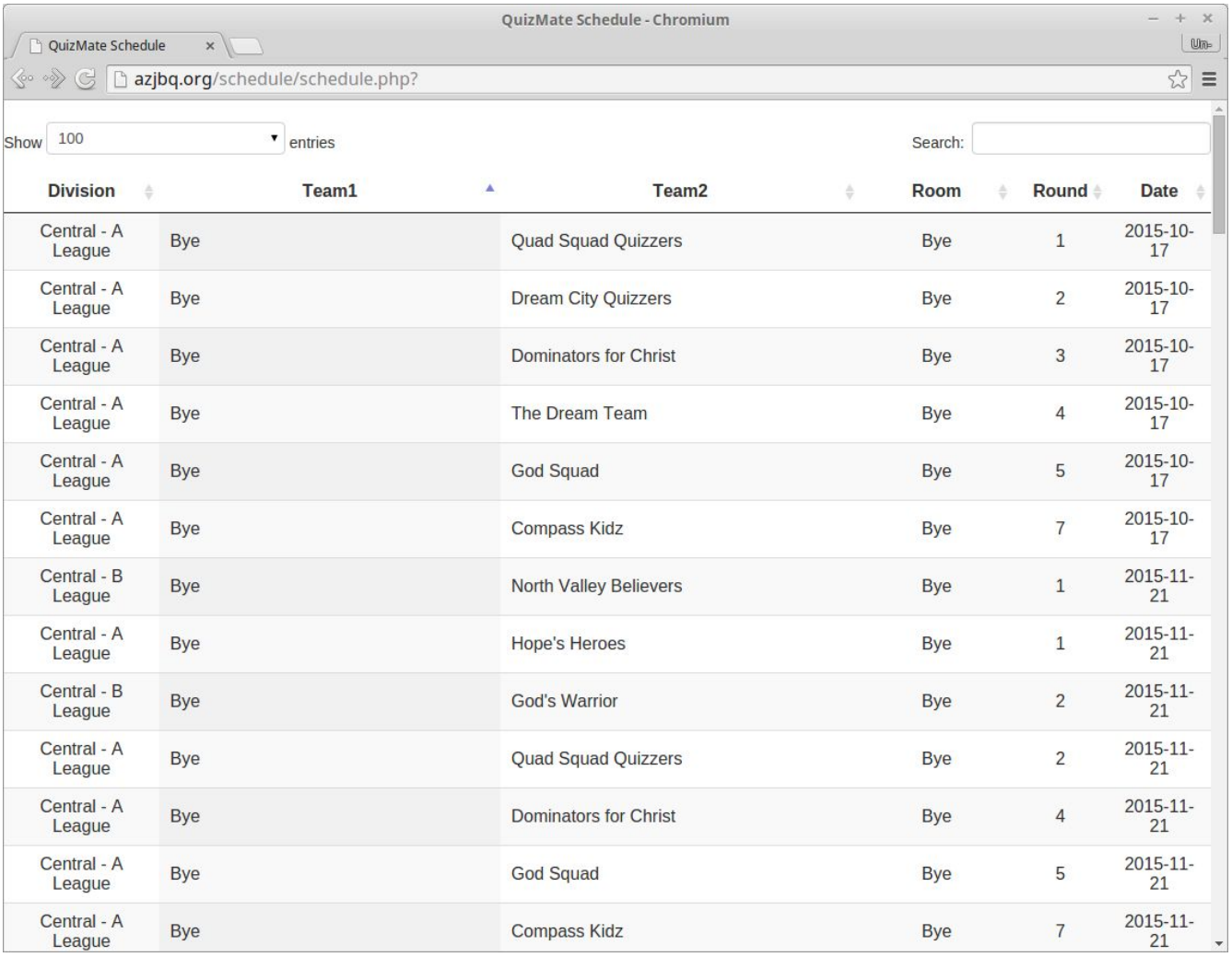
*I added "Caleb" as a filter and sorted on Date and Round. The filters are Javascript and run solely on the client device. As you can see below the records have been filtered down to 11 from 1,675.



Schedules (Selected from the main page)

(Must be at least a spectator to view)

*The same datatables engine is used for building the lists for displaying. With the filtering capabilities, teams can filter to just see their own team. Whereas officials in a room can filter to see just their room or Coordinators can filter to see their Divisions. This is a powerful view that can be filtered and sorted anyway the



Division	Team1	Team2	Room	Round	Date
Central - A League	Bye	Quad Squad Quizzers	Bye	1	2015-10-17
Central - A League	Bye	Dream City Quizzers	Bye	2	2015-10-17
Central - A League	Bye	Dominators for Christ	Bye	3	2015-10-17
Central - A League	Bye	The Dream Team	Bye	4	2015-10-17
Central - A League	Bye	God Squad	Bye	5	2015-10-17
Central - A League	Bye	Compass Kidz	Bye	7	2015-10-17
Central - B League	Bye	North Valley Believers	Bye	1	2015-11-21
Central - A League	Bye	Hope's Heroes	Bye	1	2015-11-21
Central - B League	Bye	God's Warrior	Bye	2	2015-11-21
Central - A League	Bye	Quad Squad Quizzers	Bye	2	2015-11-21
Central - A League	Bye	Dominators for Christ	Bye	4	2015-11-21
Central - A League	Bye	God Squad	Bye	5	2015-11-21
Central - A League	Bye	Compass Kidz	Bye	7	2015-11-21

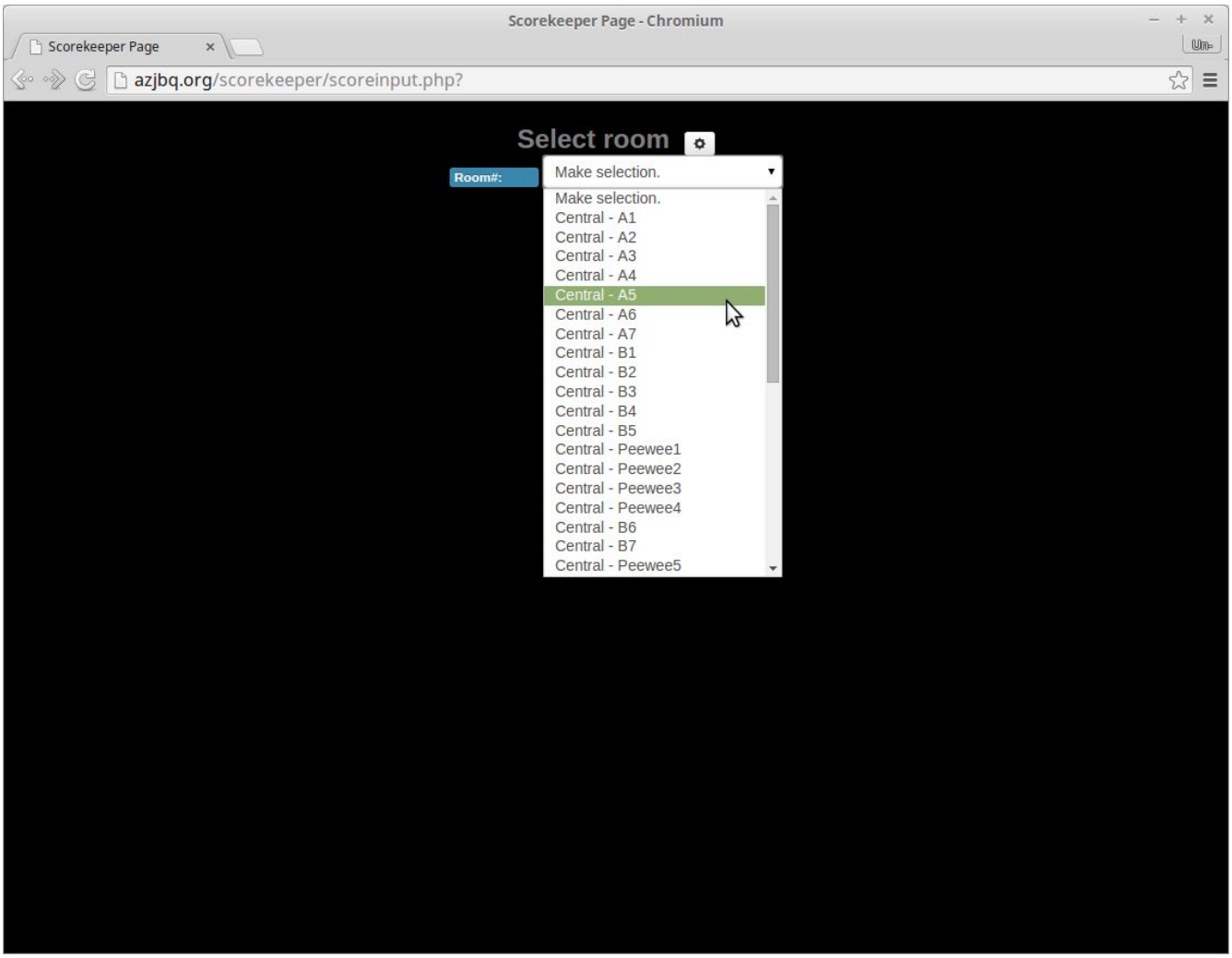
consumer wants.

Scoreinput (Selecting a room)

(Must be at least a scorekeeper to view)

*Scorekeepers are shown only portions of the page until they have set all the options and therefore cookies on their devices. This is the first step of that configuration where they must select a room to scorekeep in.

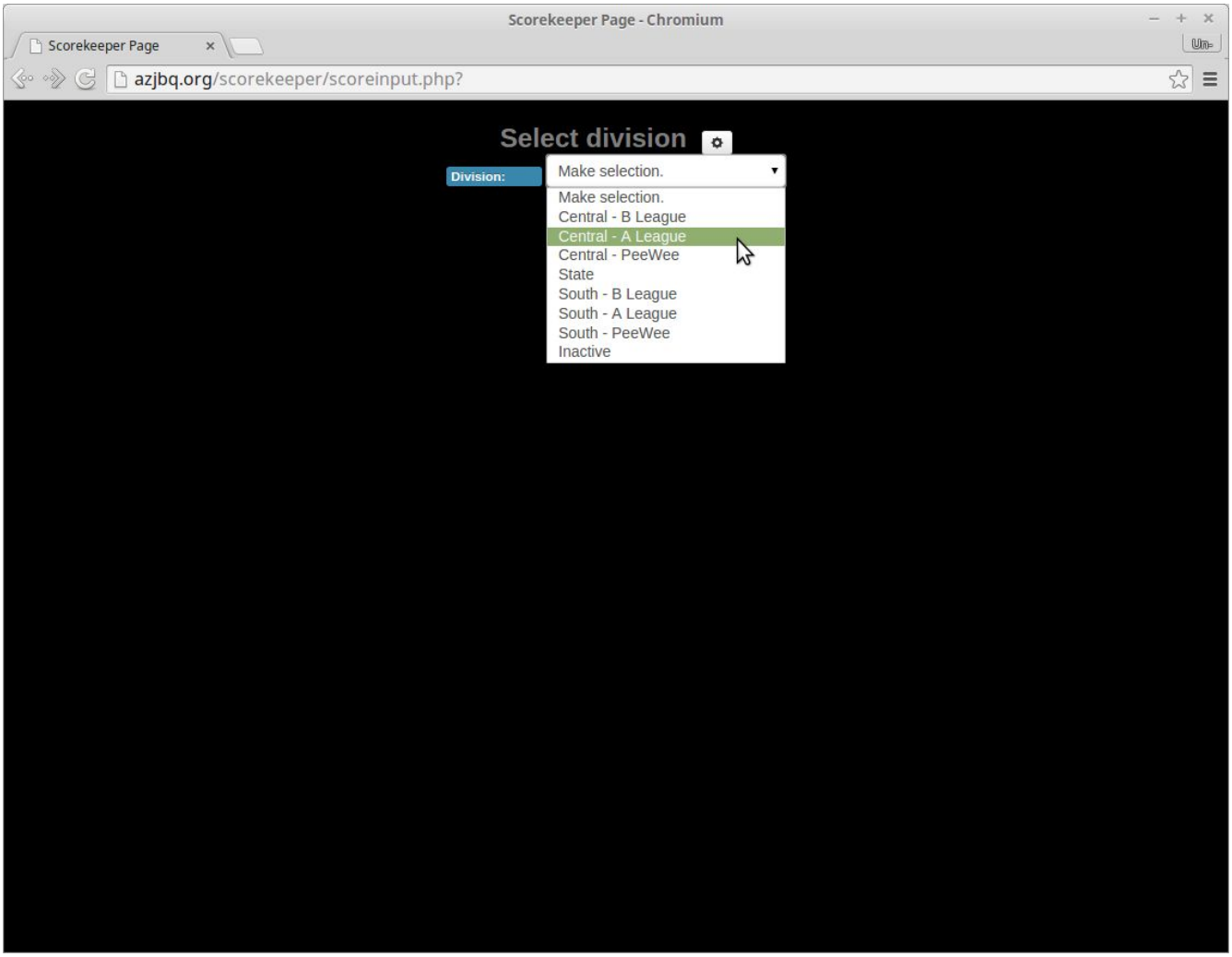
**Why the black screen. Well many people forget to bring chargers for their devices so having the black screens actually helps to save a lot of battery usage over the day.



Scoreinput (Selecting division)

(Must be at least a scorekeeper to view)

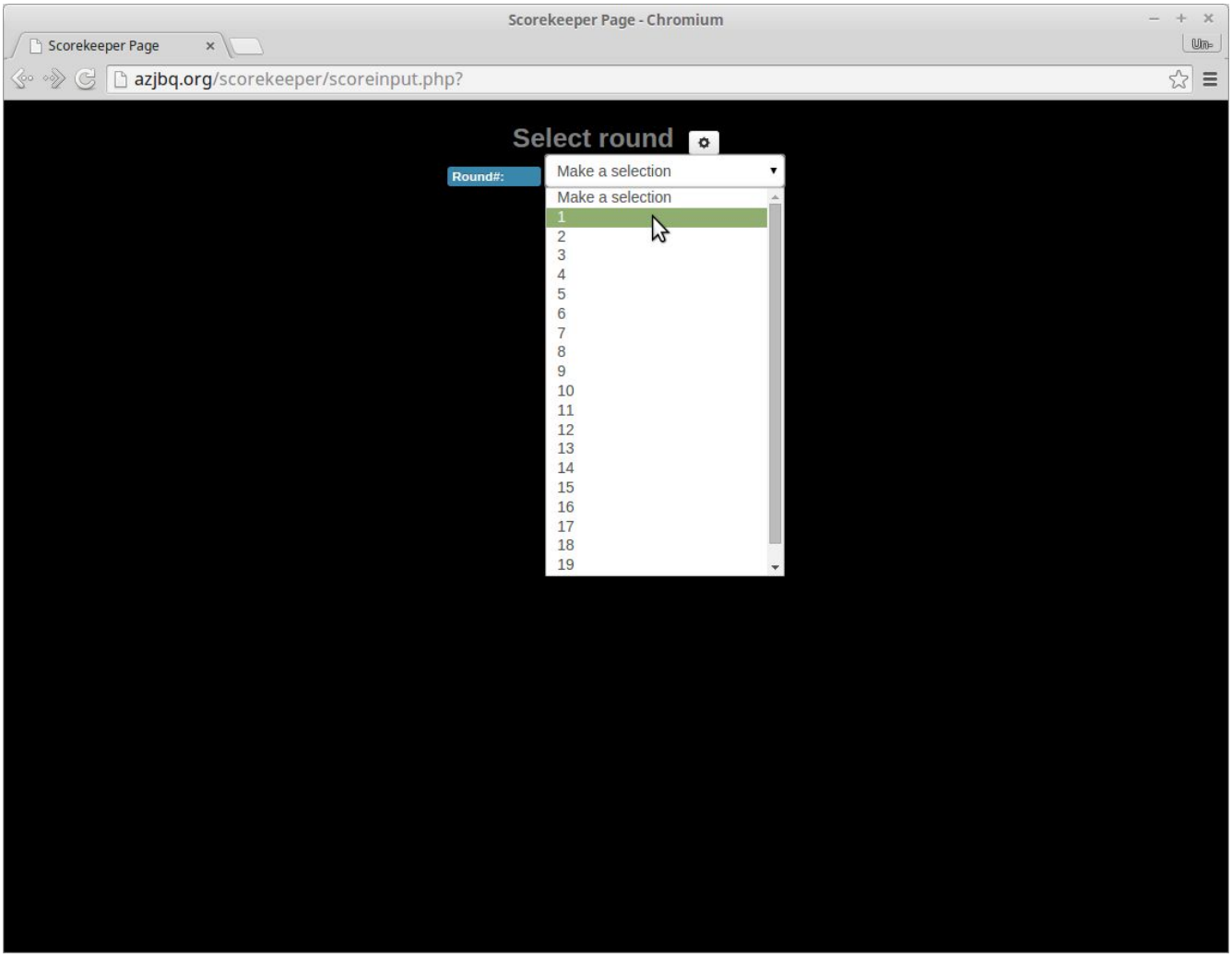
*This is the second step where they must select a division to scorekeep in.



Scoreinput (Selecting the round)

(Must be at least a scorekeeper to view)

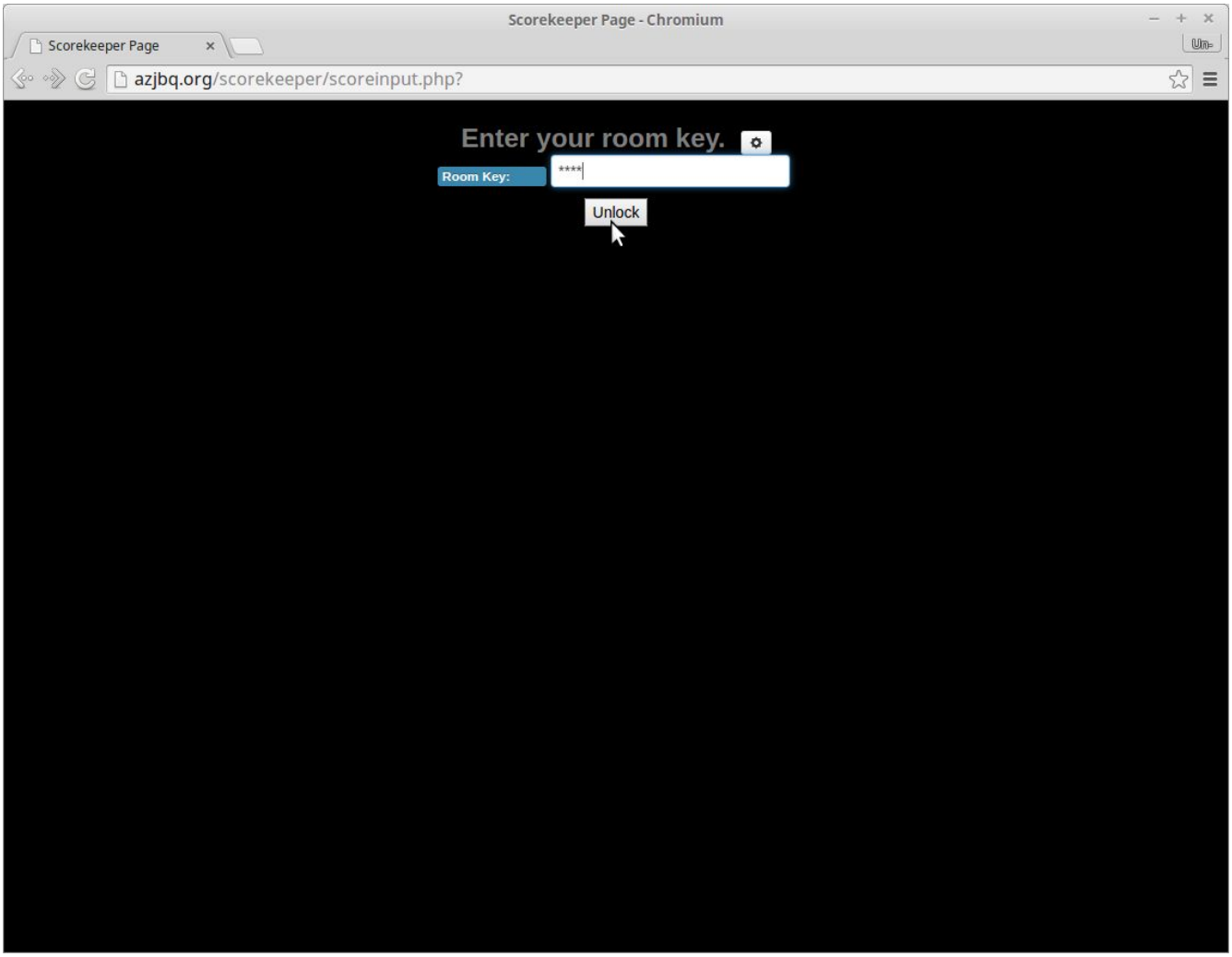
*This is the third step where they must select a starting round.



Scoreinput (Entering the room key)

(Must be at least a scorekeeper to view)

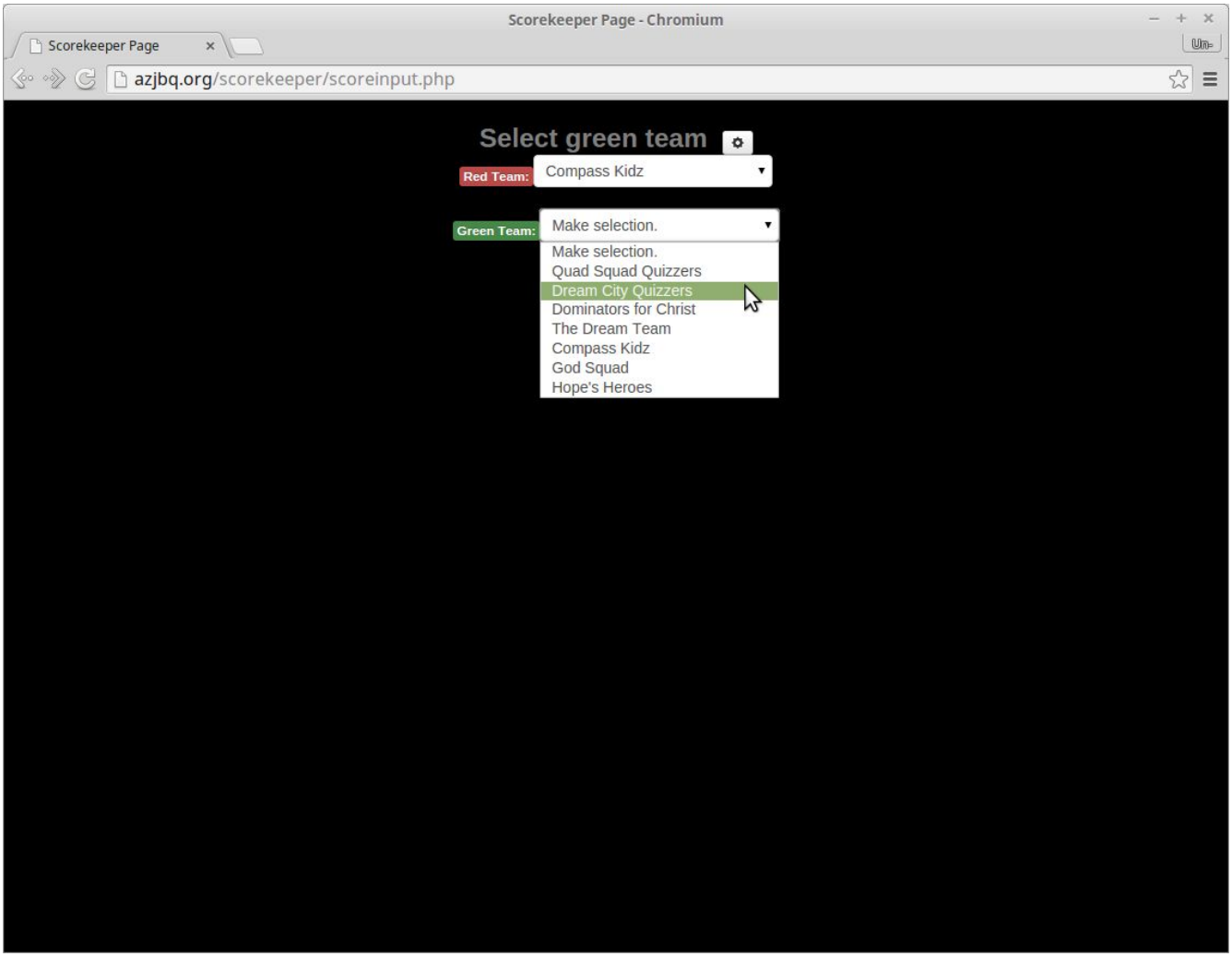
*This is the fourth step where they must put in a room key. This protects each match from having a scorekeeper pick the wrong room. Each scorekeeper is given a room key from the coordinator.



Scoreinput (Setting the teams)

(Must be at least a scorekeeper to view)

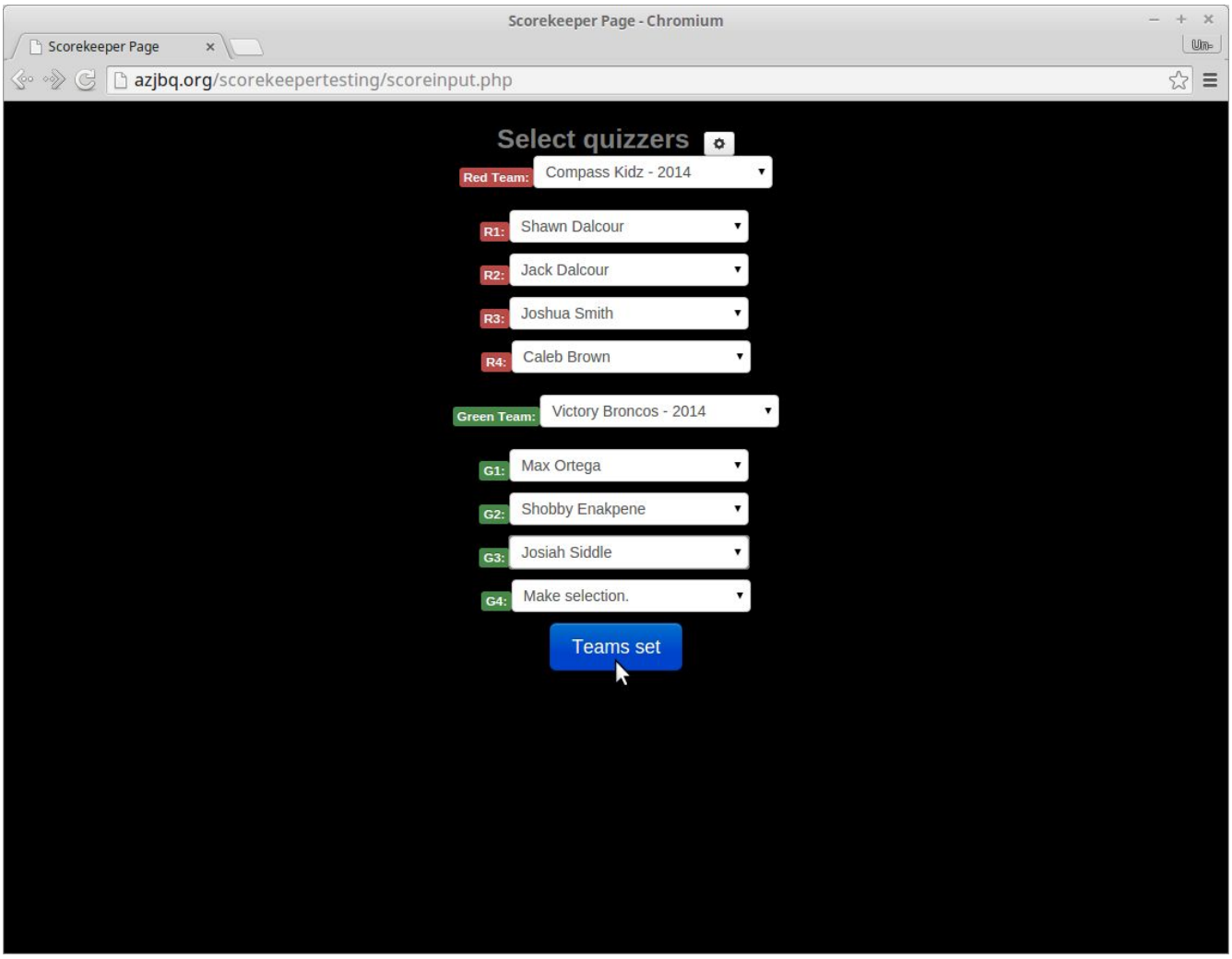
*This is the fifth step where they must set the teams according to the color they decide to sit at.



Scoreinput (Setting the quizzers)

(Must be at least a scorekeeper to view)

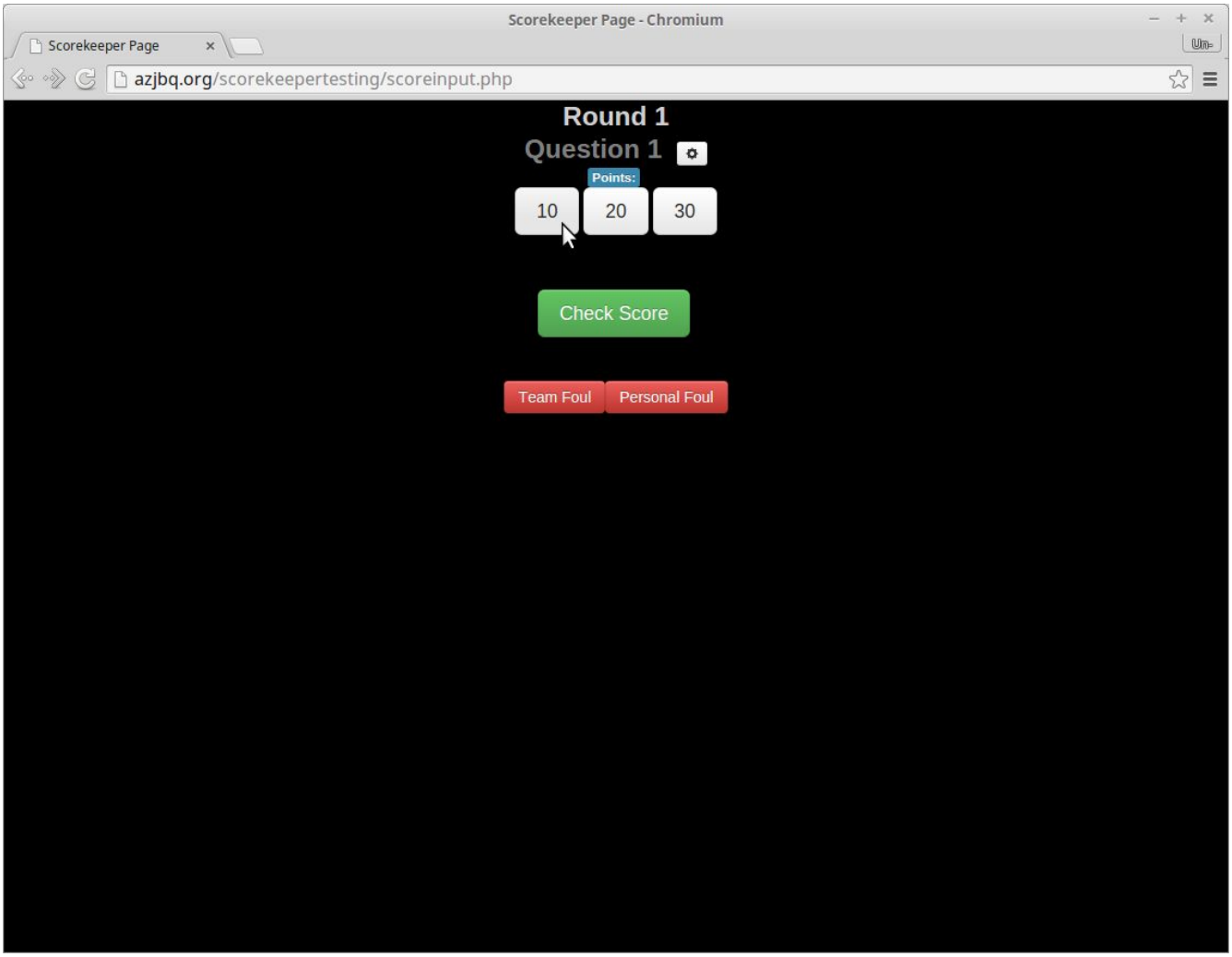
*This is the sixth step where they must set the quizzers according to the chair they decide to sit in. I changed the screenshots to the testing area so for demo purposes, so that we wouldn't be adding scores to the production database.



Scoreinput (Scoring - picking the point value)

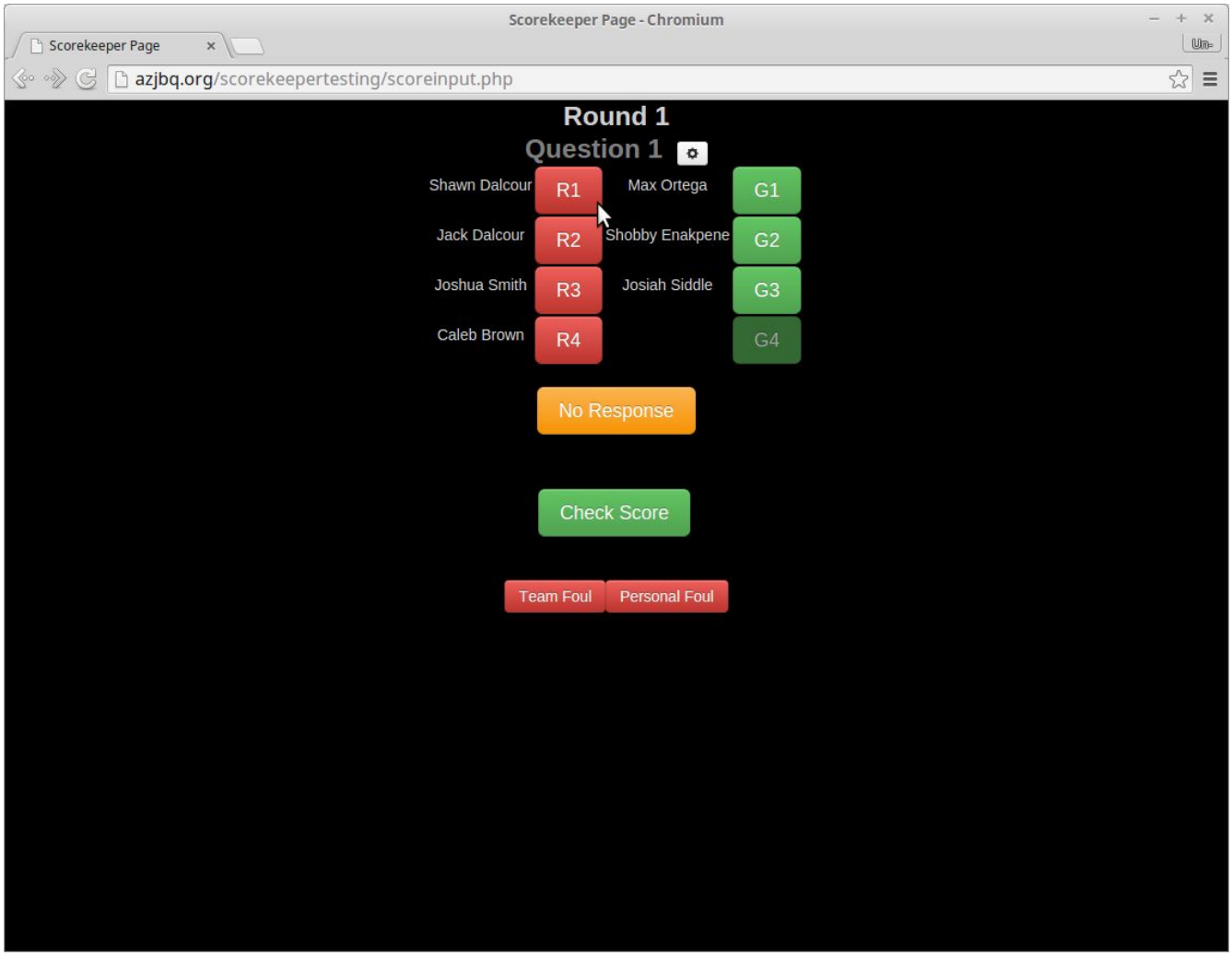
(Must be at least a scorekeeper to view)

*The match is now set and the scorekeeper can begin to score the match. Each question is worth 10, 20, or 30 points.



Scoreinput (Scoring - picking the quizzer)

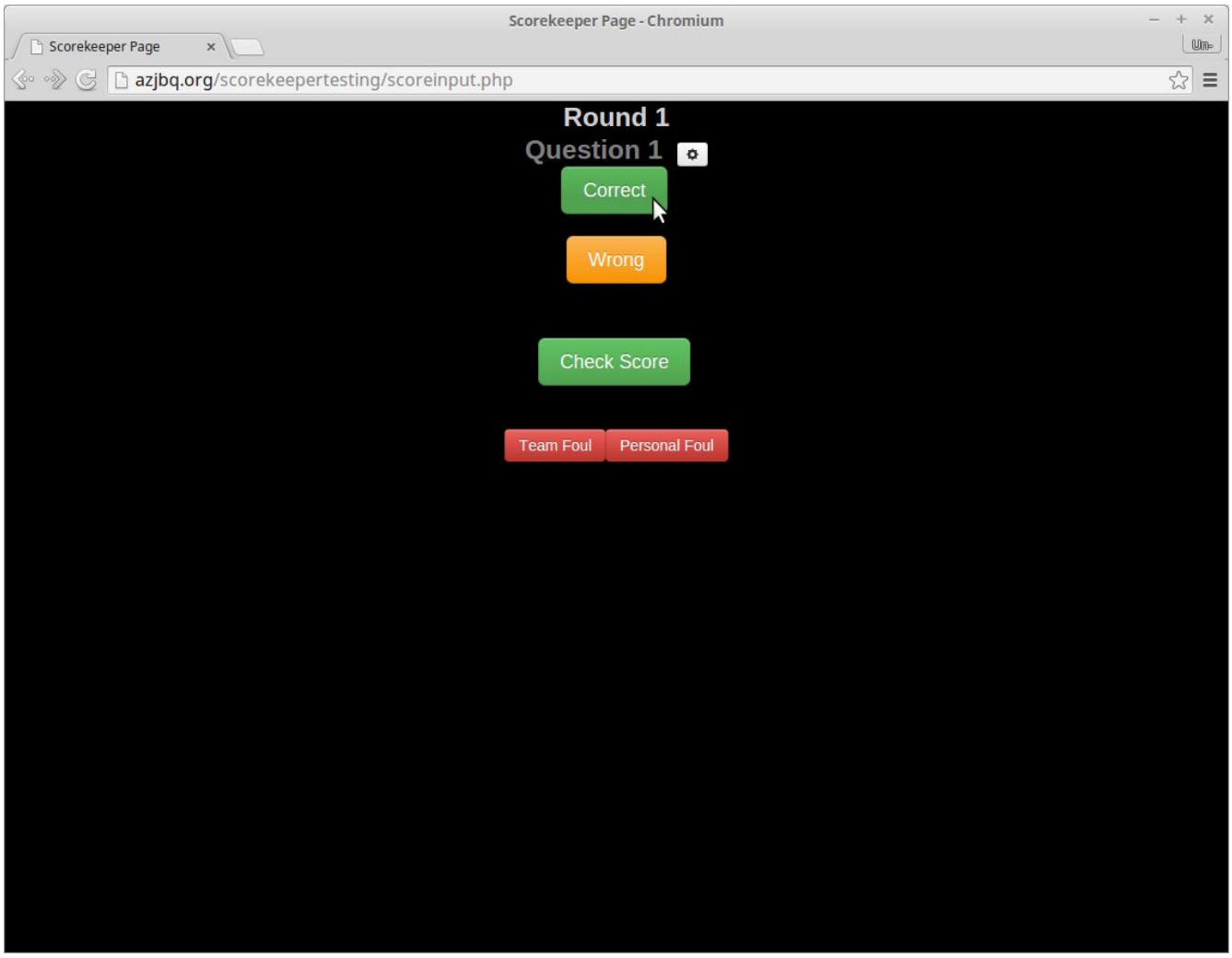
(Must be at least a scorekeeper to view)
*Active quizzers are shown and the inactive chairs are faded as you can see below. The scorekeeper picks the quizzer that buzzed in. If no quizzer buzzes in then the scorekeeper would hit the "No Response" button



Scoreinput (Scoring - picking the result)

(Must be at least a scorekeeper to view)

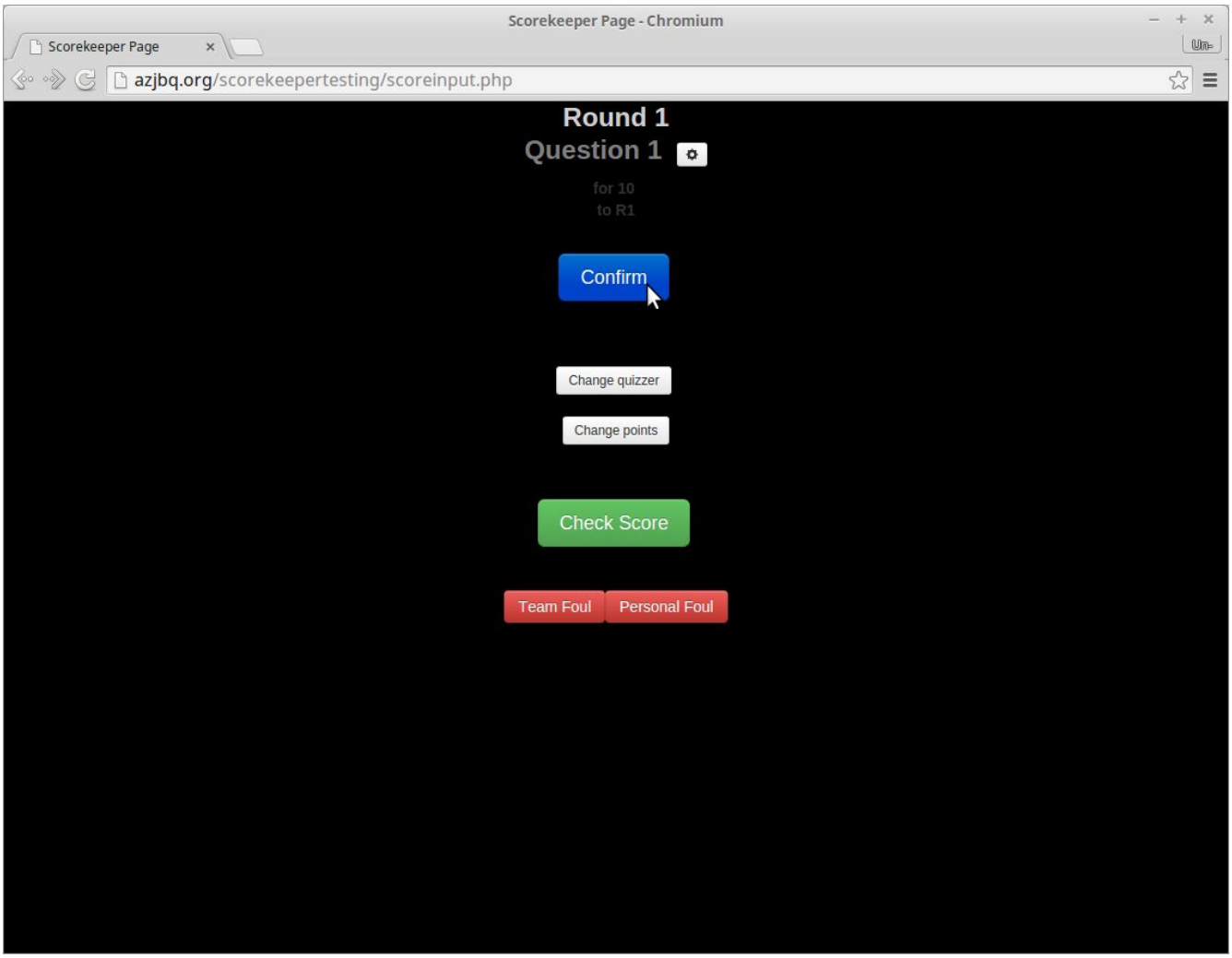
*Scorekeeper selects the result.



Scoreinput (Scoring - confirm)

(Must be at least a scorekeeper to view)

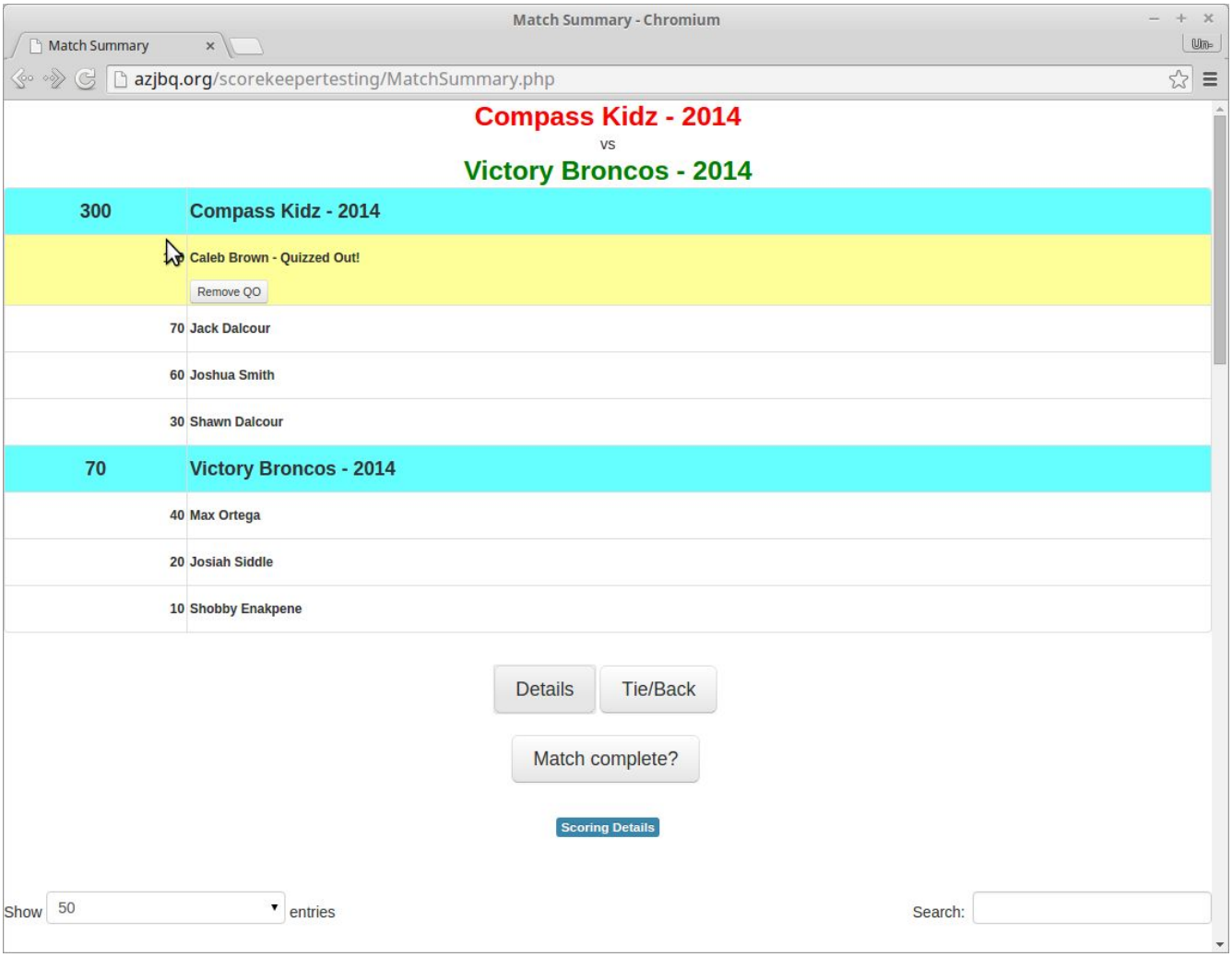
*Scorekeeper confirm or change the result.



Scoreinput (MatchSummary - Checking the summary)

(Must be at least a scorekeeper to view)

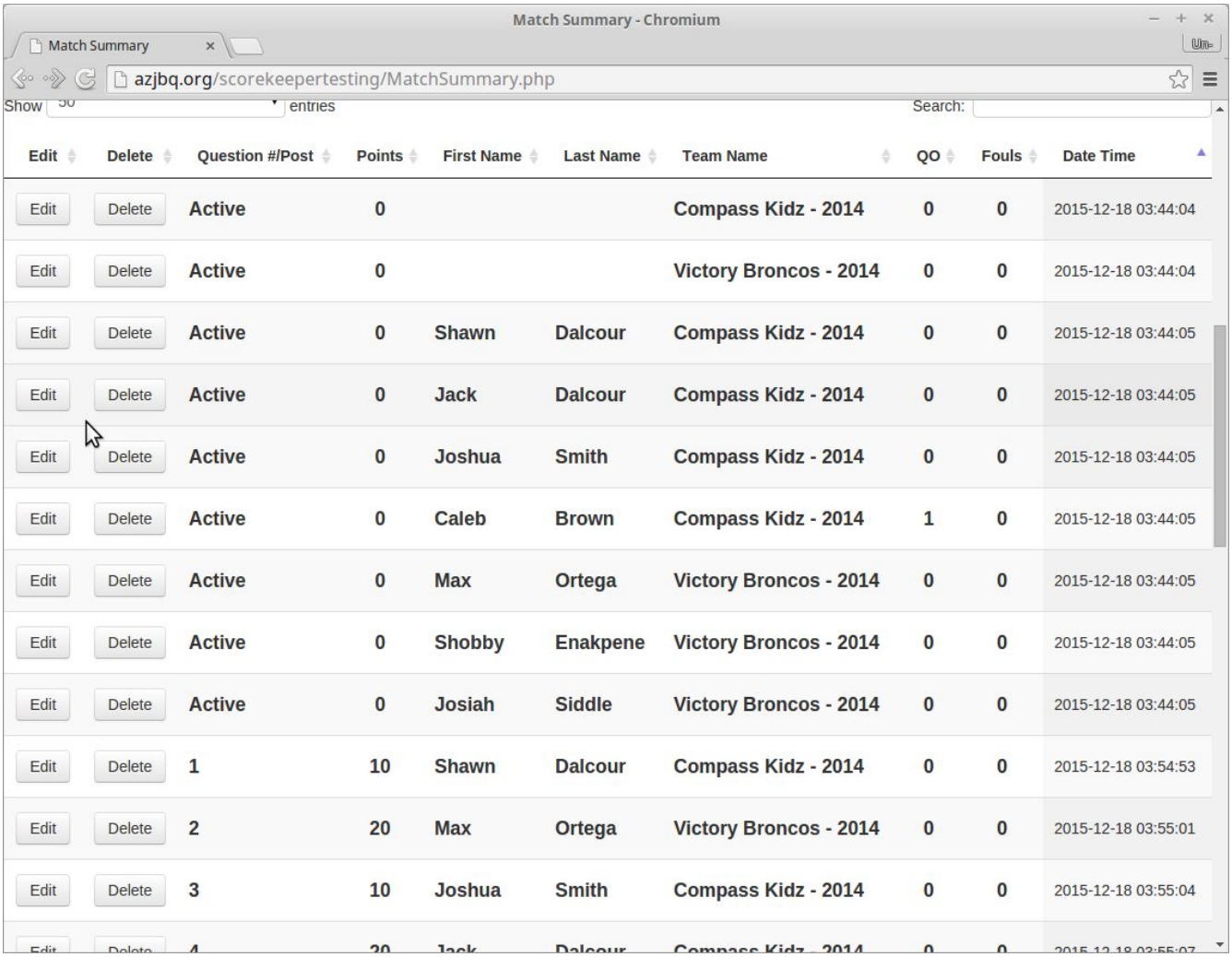
*Scorekeeper reads the results. The MatchSummary page sorts the scores by winning team, and top quizzers by score per team.



Scoreinput (MatchSummary - Checking the details)

(Must be at least a scorekeeper to view)

*Scorekeeper can make changes to scoring based on a mistake or a reversal of a decision. Often times the coaches have recorded the scores differently and scorekeeper can go through each score until they reconcile the scores.

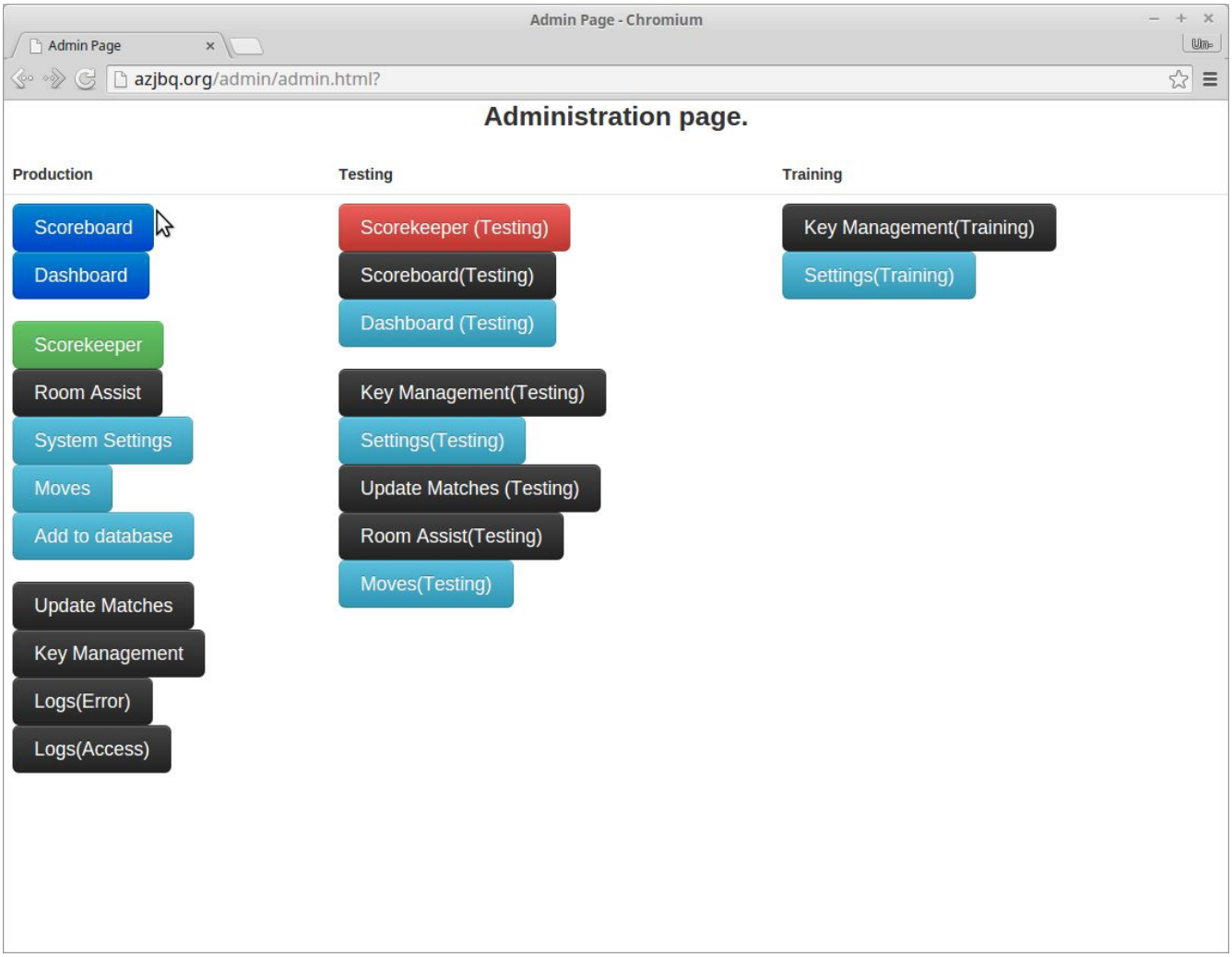


Edit	Delete	Question #/Post	Points	First Name	Last Name	Team Name	QO	Fouls	Date Time
Edit	Delete	Active	0			Compass Kidz - 2014	0	0	2015-12-18 03:44:04
Edit	Delete	Active	0			Victory Broncos - 2014	0	0	2015-12-18 03:44:04
Edit	Delete	Active	0	Shawn	Dalcour	Compass Kidz - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Jack	Dalcour	Compass Kidz - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Joshua	Smith	Compass Kidz - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Caleb	Brown	Compass Kidz - 2014	1	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Max	Ortega	Victory Broncos - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Shobby	Enakpene	Victory Broncos - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	Active	0	Josiah	Siddle	Victory Broncos - 2014	0	0	2015-12-18 03:44:05
Edit	Delete	1	10	Shawn	Dalcour	Compass Kidz - 2014	0	0	2015-12-18 03:54:53
Edit	Delete	2	20	Max	Ortega	Victory Broncos - 2014	0	0	2015-12-18 03:55:01
Edit	Delete	3	10	Joshua	Smith	Compass Kidz - 2014	0	0	2015-12-18 03:55:04

Administration (Admin page)

(Must be at least an admin to view)

*I made this page to help navigate faster to different areas of the system, even if that area is test or training. As you can see each are divided into Production, Testing, and Training.

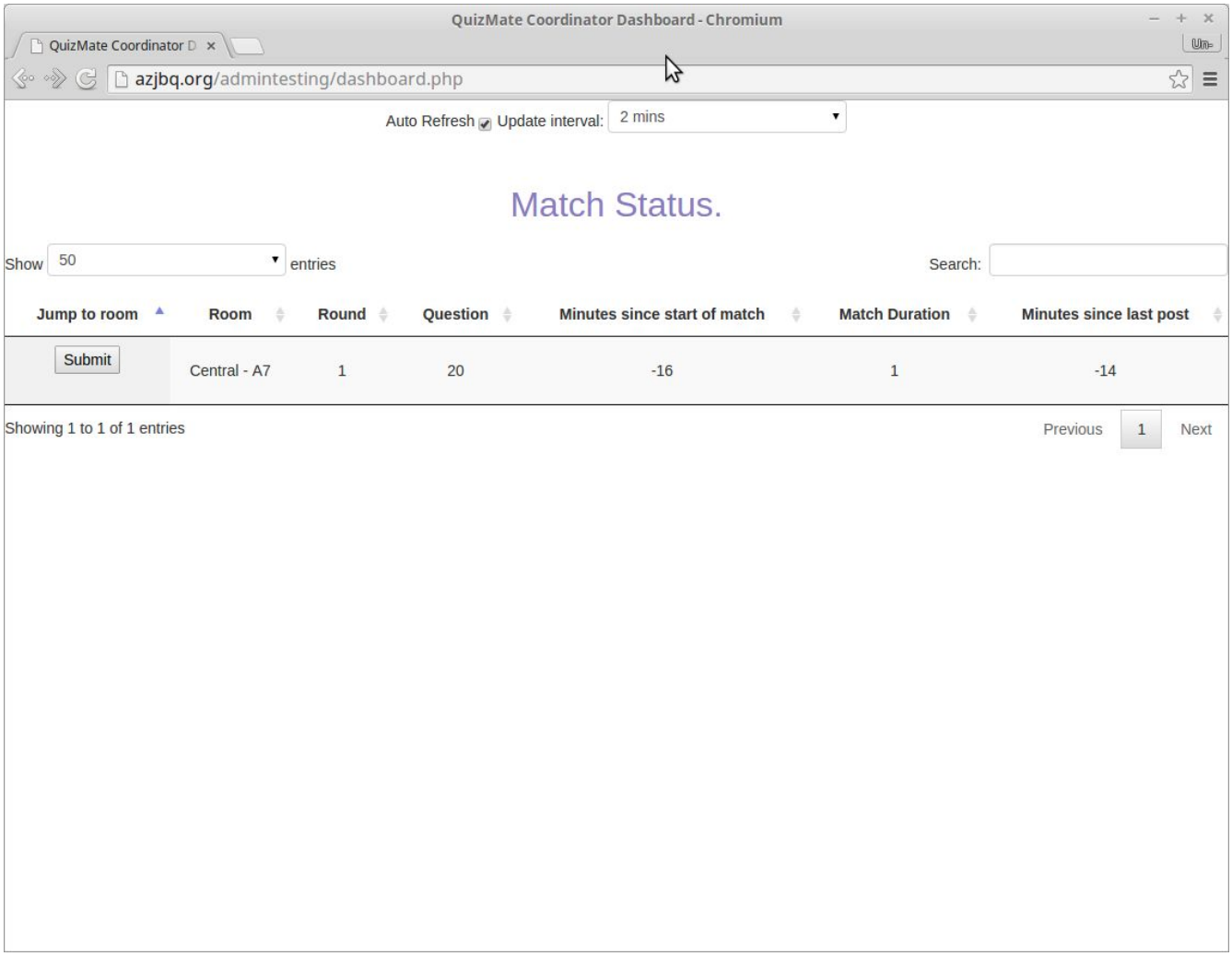


Administration (Dashboard page)

(Must be at least an admin to view)

*This page is designed for the coordinator to watch during a meet. From this page a coordinator can determine if a room is running slow or hung up. They can jump into the MatchSummary page of that match, live, to see which teams are in there and what they might be working on. They can even make repairs to the scores from that page. With this view the coordinator knows which room this is, based on the name and therefore can physically go check on the room if needed.

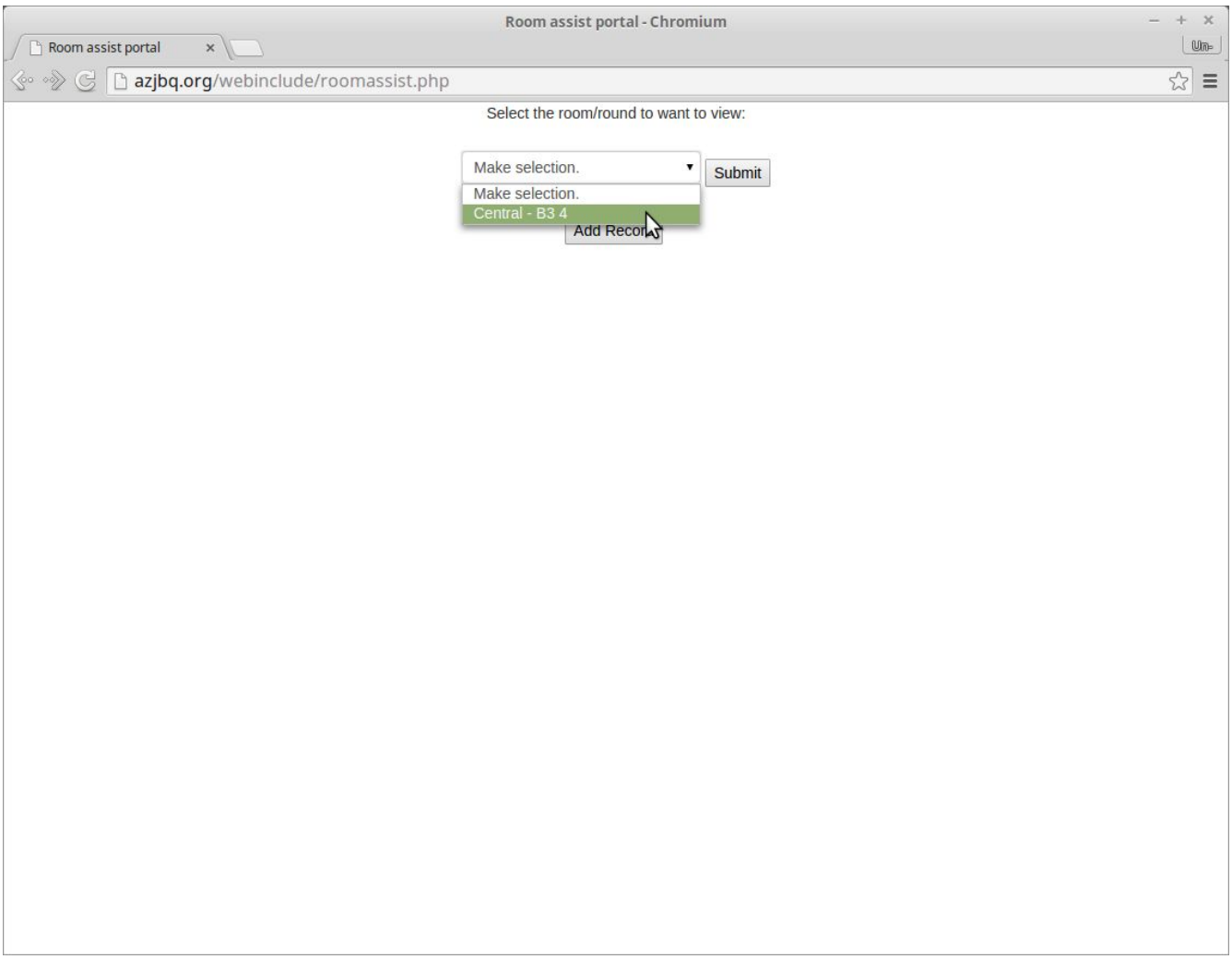
**There are collapsable panels that show errors if the system detects something wrong. For instance if a room is detected that has three teams in it, then a big red panel would appear at the top showing which room is the problem.



Administration (Room Assist Portal - Match Selection)

(Must be at least an admin to view)

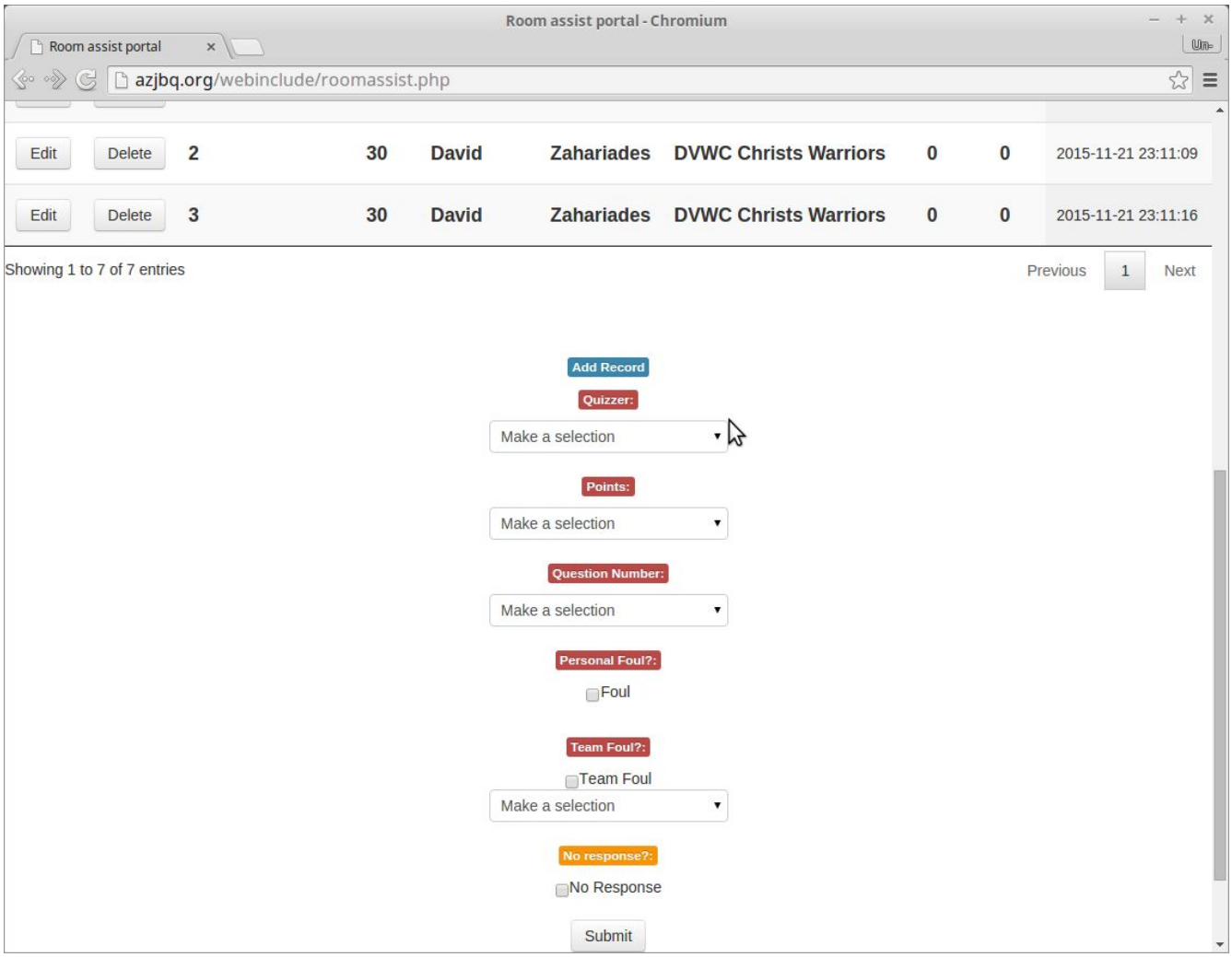
*This page allows an admin or coordinator the chance to jump into a room and assist the scorekeeper. The drop down displays only the active matches. The admin picks the match and hits submit to view the details of the match.



Administration (Room Assist Portal - Details)

(Must be at least an admin to view)

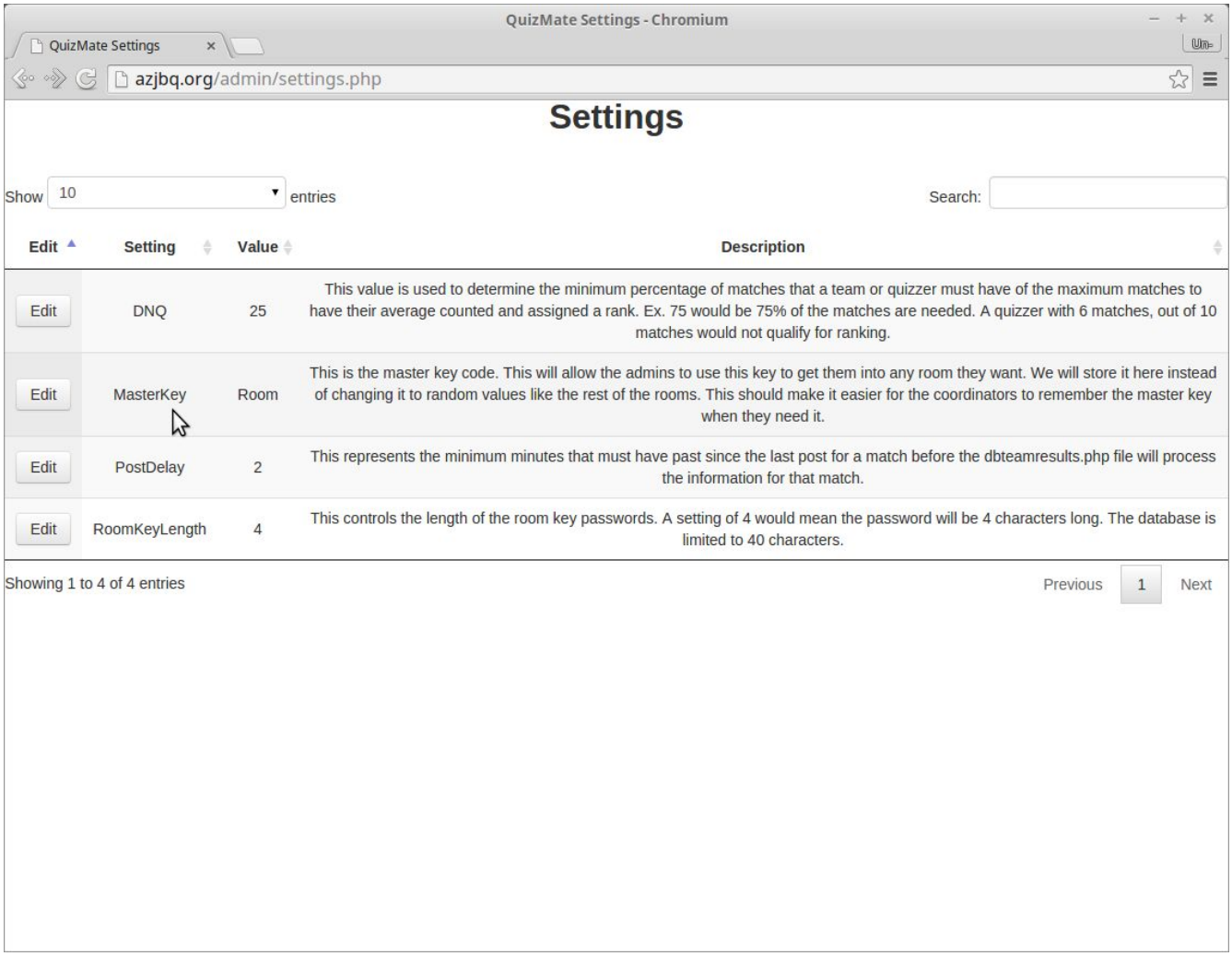
*This is the detailed view of the match that was selected. As you can see the admin can edit or delete specific records. The admin can also add new records to the match with the options at the bottom of the page.



Administration (Settings)

(Must be at least an admin to view)

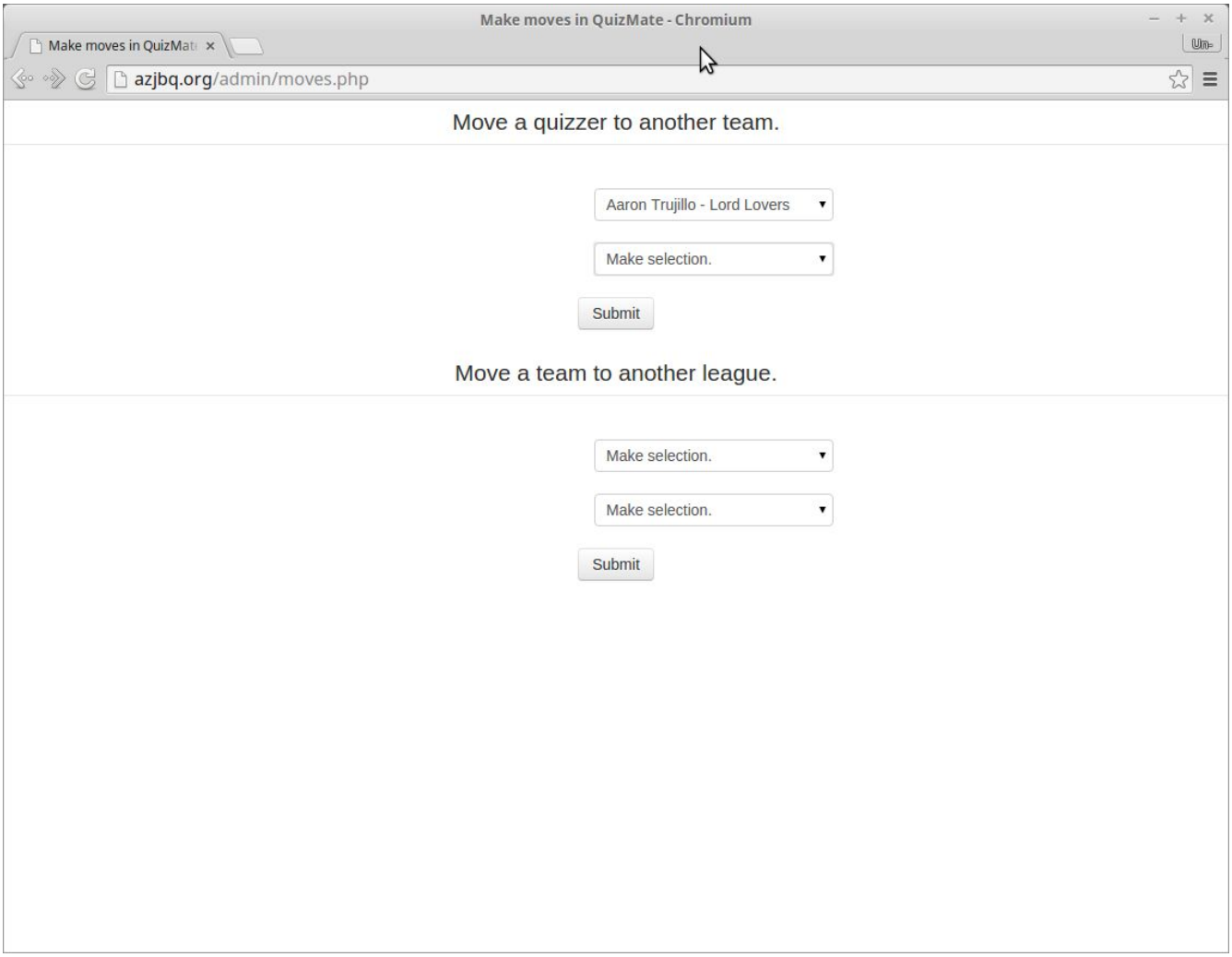
*This page allows for certain aspects of the system to be adjusted safely without affecting code. This was a late add-on and so certain parts of the system are being retrofitted to allow the admin to adjust more aspects of the system from this panel. There is a settings page for each test, training, and production.



Administration (Moves page)

(Must be at least an admin to view)

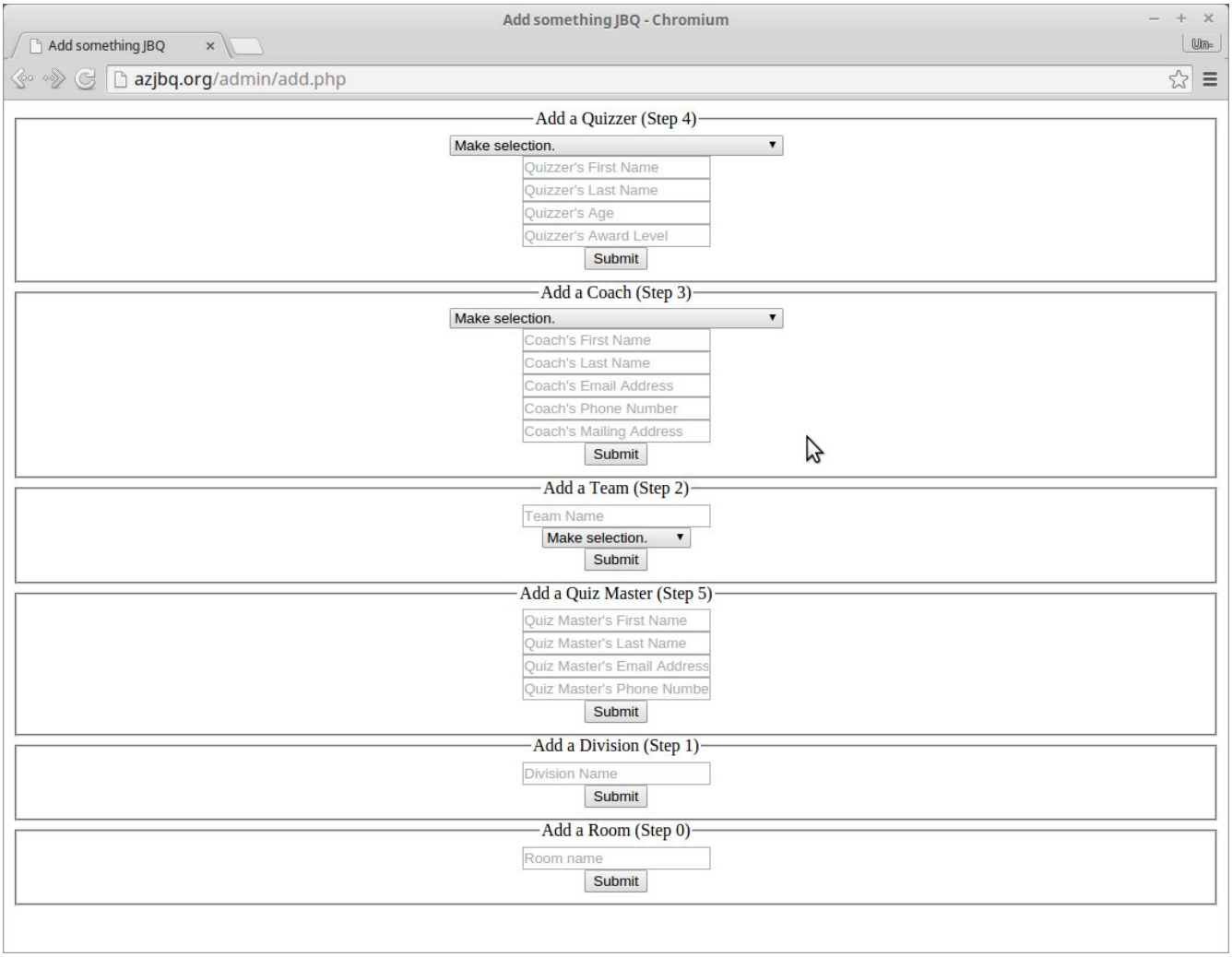
*This allows for moves of quizzers and teams. This actually happens before almost every match. It was very cumbersome to do this work directly in the database tables, so I built this page to make this work much easier.



Administration (Add page)

(Must be at least an admin to view)

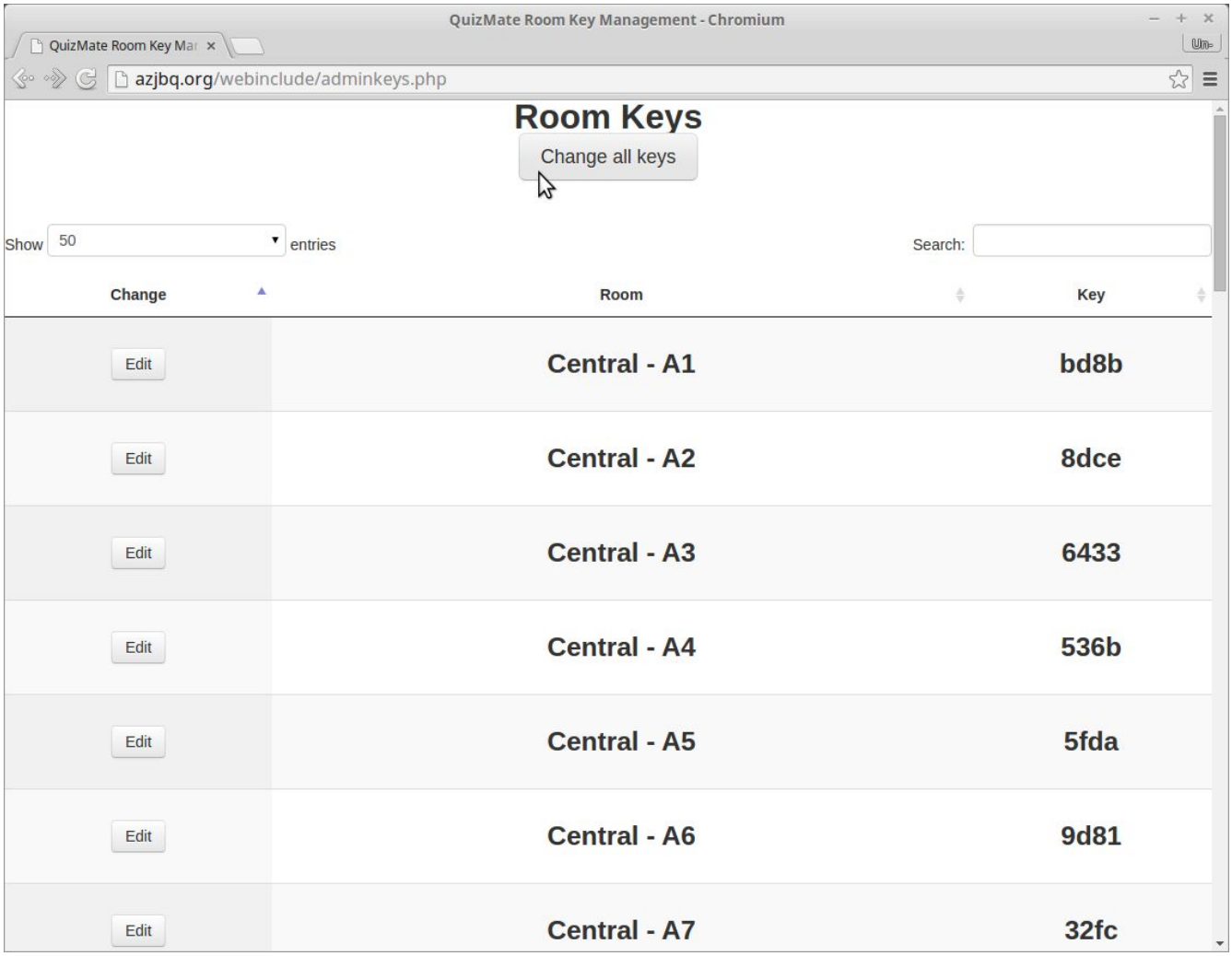
*This page allows for all adds to the system. If the admin needs to add a quizzer or a team or anything, they come here to this page. They're numbered to assist the admin. Since, you can't add a quizzer until their team exists it helps to number them to make this workflow easier.



Administration (Room Keys page)

(Must be at least an admin to view)

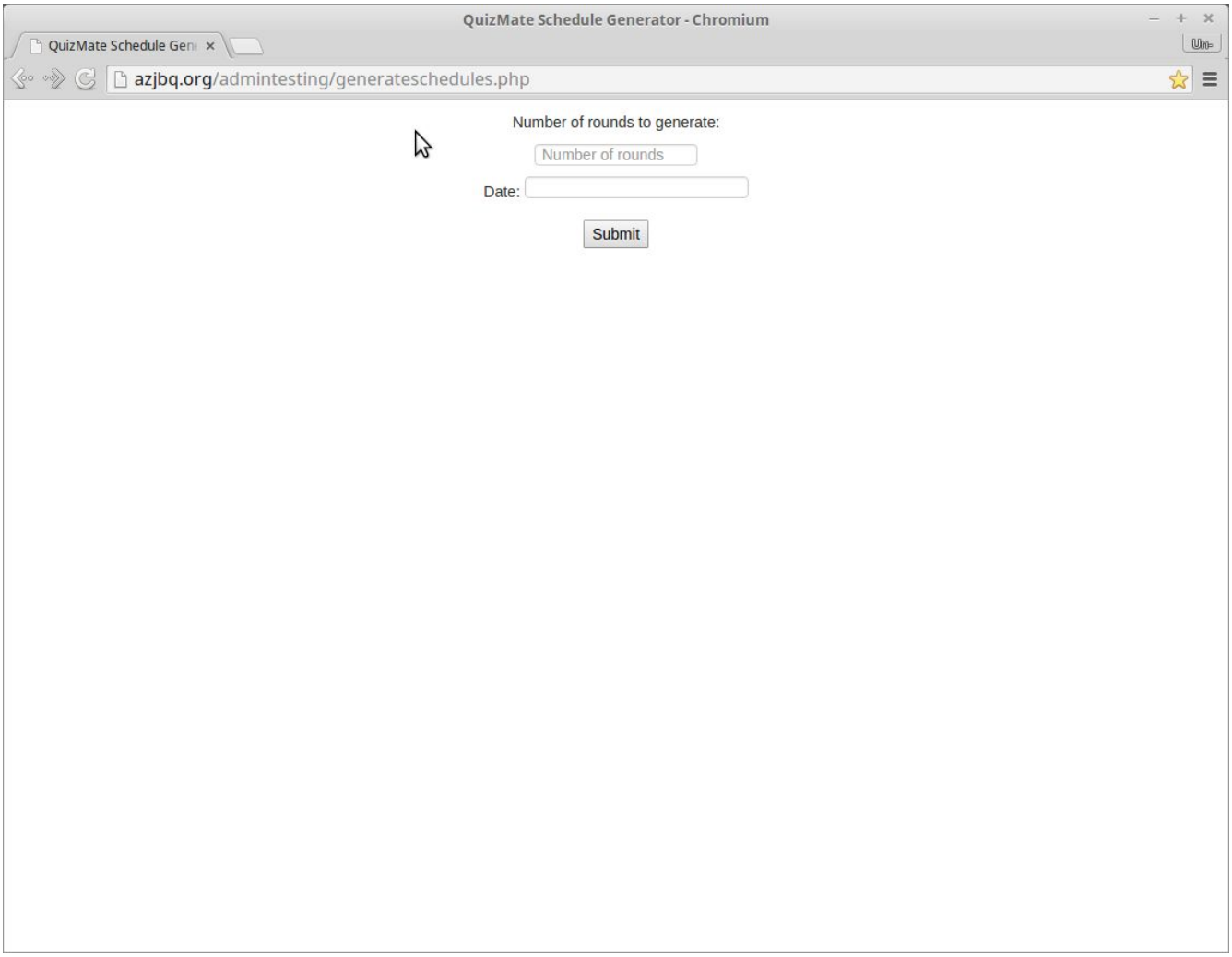
*This page allows the admin to change the room keys. The admin can change them all at once or just one of the rooms, that might be having problems. The code generates random alphanumeric strings that match the length set in the settings table. This table is set to print large so that each line can be cut and put into the room packets for the competition. The scorekeeper gets this piece of paper and enters the key to authorize them to the room.



Administration (Schedule Generator page)

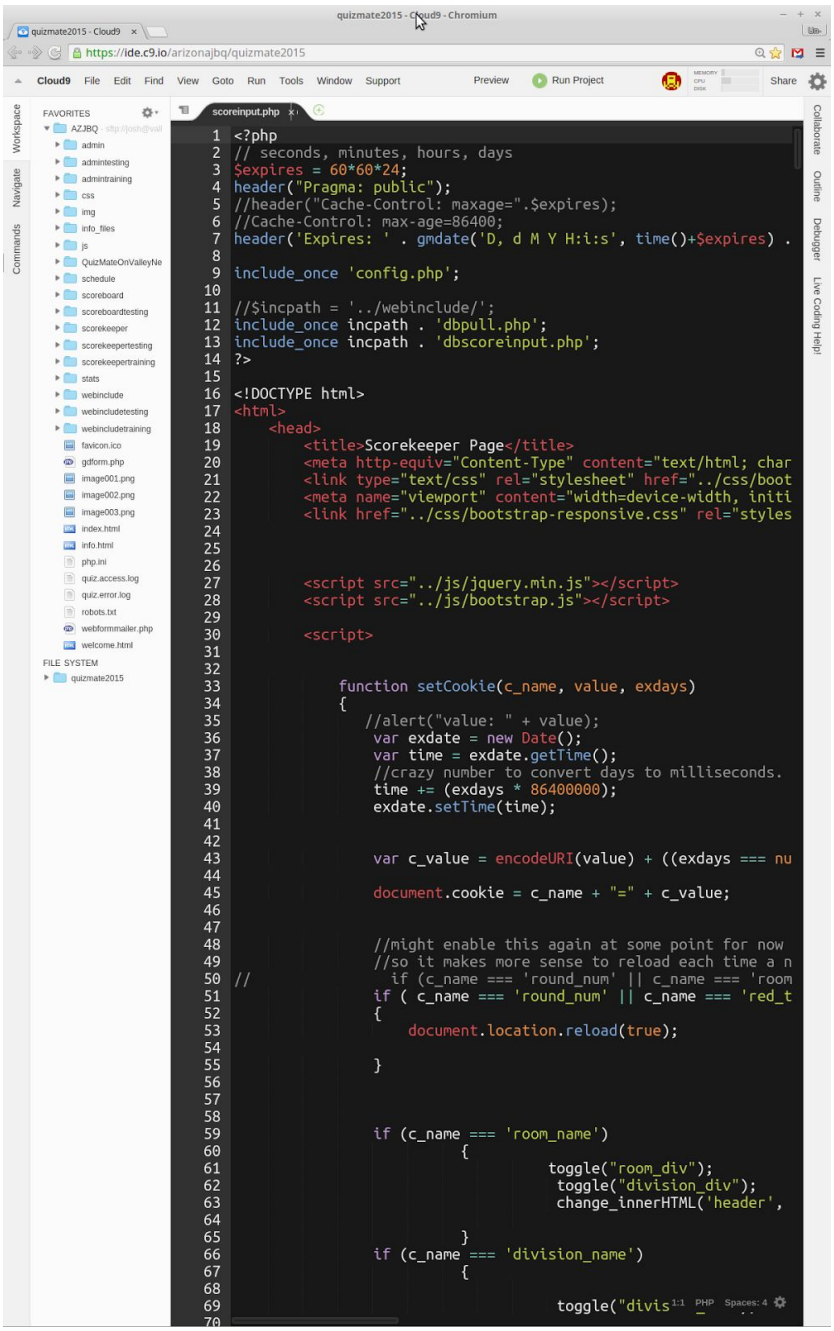
(Must be at least an admin to view)

*This page is designed to allow admins to quickly schedule a meet. This code picks up all the active teams; schedules them based on all the past meets for the year; and sets down a schedule for the next meet. This is some very complicated scheduling as it takes into account how many times a certain matchup has occurred this season and tries to balance out those matchups. In this way the scheduling tries to make it fair by not forcing one team to go against the hardest team all the time. This is important for the Arizona District since we give out trophies for the season record it's important to make it as fair as possible across all the meets and not start out from scratch each time.



IDE Screen Shots

This is a screenshot of the web based IDE platform called Cloud9. This allows me to code from any machine and even if my system crashes again, I know that I will be able to code right away from another computer.

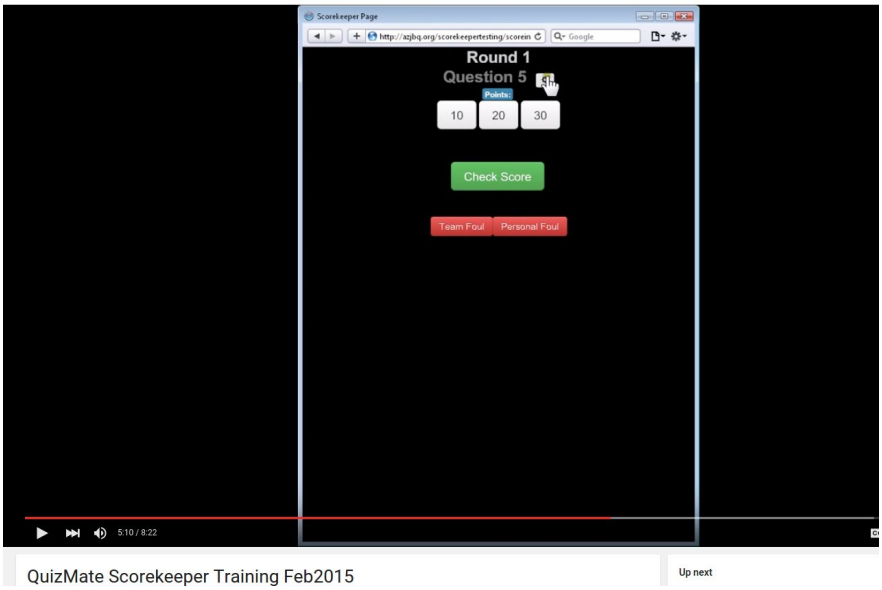



```
<?php
// seconds, minutes, hours, days
$expires = 60*60*24;
header("Pragma: public");
//header("Cache-Control: maxage=".$expires);
//Cache-Control: max-age=86400;
header('Expires: ' . gmdate('D, d M Y H:i:s', time()+$expires) .

include_once 'config.php';

//$incpath = '../webinclude/';
include_once incpath . 'dbpull.php';
include_once incpath . 'dbscoreinput.php';
?>

<!DOCTYPE html>
<html>
    <head>
        <title>Scorekeeper Page</title>
        <meta http-equiv="Content-Type" content="text/html; char
        <link type="text/css" rel="stylesheet" href="../css/boot
        <meta name="viewport" content="width=device-width, initi
        <link href="../css/bootstrap-responsive.css" rel="styles



        <script src="../js/jquery.min.js"></script>
        <script src="../js/bootstrap.js"></script>

        <script>


            function setCookie(c_name, value, exdays)
            {
               //alert("value: " + value);
                var exdate = new Date();
                var time = exdate.getTime();
                //crazy number to convert days to milliseconds.
                time += (exdays * 86400000);
                exdate.setTime(time);


                var c_value = encodeURI(value) + ((exdays === nu

                document.cookie = c_name + "=" + c_value;


                //might enable this again at some point for now
                //so it makes more sense to reload each time a n
//                if (c_name === 'round_num' || c_name === 'room
                if ( c_name === 'round_num' || c_name === 'red_t
                {
                    document.location.reload(true);

                }



                if (c_name === 'room_name')
                        {
                                toggle("room_div");
                                 toggle("division_div");
                                 change_innerHTML('header',

                        }
                if (c_name === 'division_name')
                        {

                                toggle("divis
```

Demo Videos

QuizMate Scorekeeper Training Feb2015

https://www.youtube.com/watch?v=KaxQeD0ZDOk&feature=youtu.be

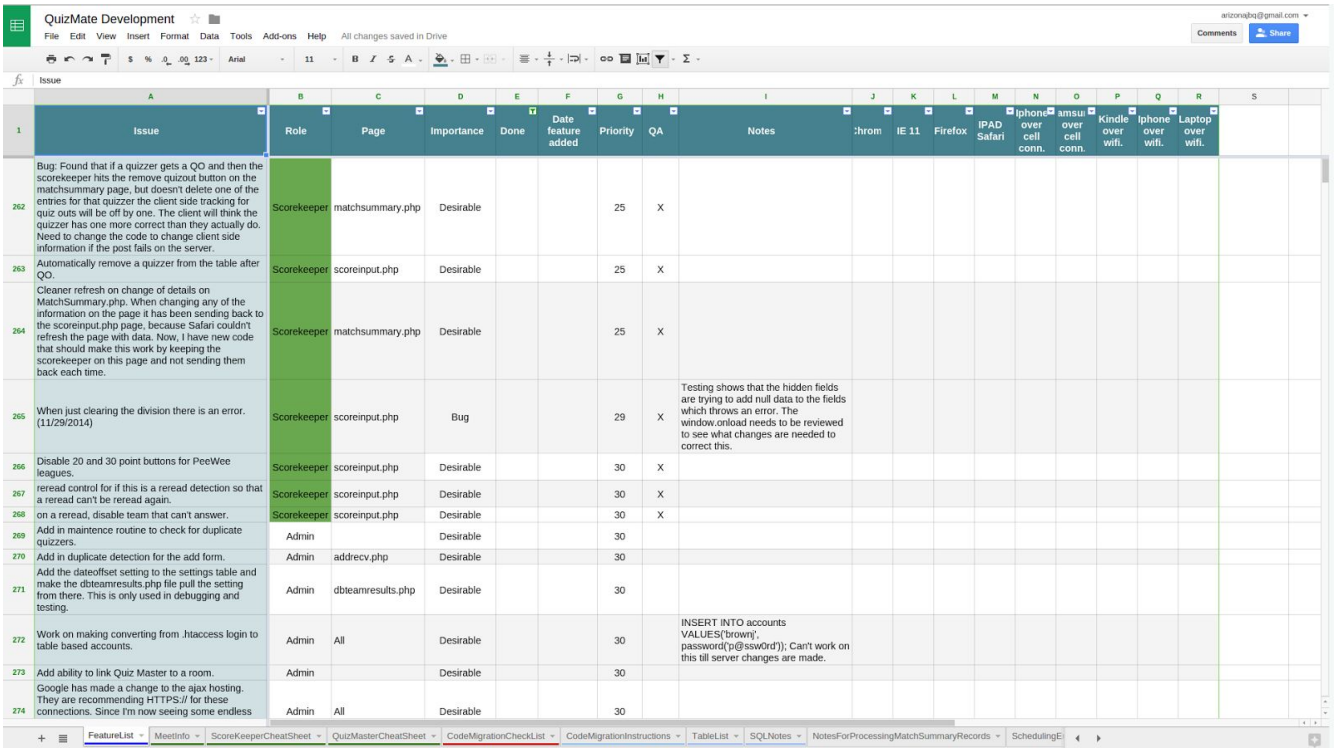


Project Files

Development Spreadsheet

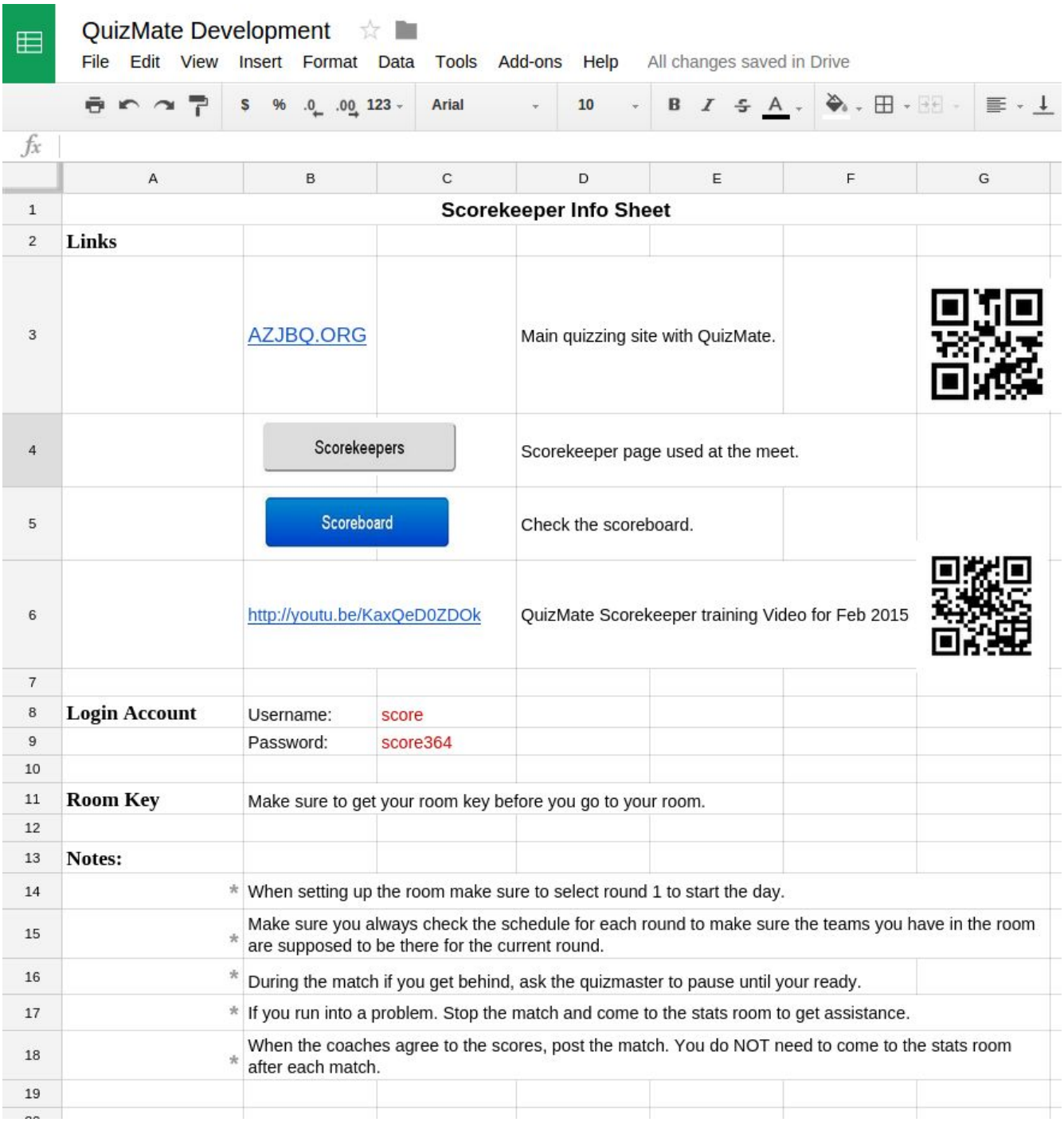
This is the spreadsheet that I keep track of features and bugs, and communicate with testers back and forth.
Columns:

1. Issue - The description of the feature, bug, or test scenario.
2. Role - The role that is expected to interact with the change.
3. Page - This just gives me an idea of what file to change.
4. Importance - This for sorting out the required elements from the desirable features.
5. Done - This is marked when I complete the changes. This also helps the testers to know which elements are ready for testing.
6. Date feature added - This is just to mark down when I've completed the work. This helps when troubleshooting to know what were changed recently and couple responsible for the issues.
7. Priority -
8. QA
9. Notes
10. Chrome,IE11,Firefox,iPad Safari,iPhone over cell conn.,Samsung over cell conn.,Kindle over WiFi.,iPhone over WiFi.,Laptop over WiFi.



Scorekeeper Cheatsheet

This is the paper that we print for each scorekeeper to assist them in getting started for the day. The first QR code helps the scorekeeper get to the site quicker. The second QR code helps them get to the training video.



QuizMate Development

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

	A	B	C	D	E	F	G
1				**Scorekeeper Info Sheet**			
2	**Links**						
3		AZJBQ.ORG		Main quizzing site with QuizMate.			
4		Scorekeepers		Scorekeeper page used at the meet.			
5		Scoreboard		Check the scoreboard.			
6		http://youtu.be/KaxQeD0ZDOk		QuizMate Scorekeeper training Video for Feb 2015			
7							
8	**Login Account**	Username:	score				
9		Password:	score364				
10							
11	**Room Key**	Make sure to get your room key before you go to your room.					
12							
13	**Notes:**						
14	*	When setting up the room make sure to select round 1 to start the day.					
15	*	Make sure you always check the schedule for each round to make sure the teams you have in the room are supposed to be there for the current round.					
16	*	During the match if you get behind, ask the quizmaster to pause until your ready.					
17	*	If you run into a problem. Stop the match and come to the stats room to get assistance.					
18	*	When the coaches agree to the scores, post the match. You do NOT need to come to the stats room after each match.					
19							

Code Migration Checklist

This is the sheet that I use to track with code modules I work on. If I work on a file I mark it in the testing area, and then when I go to deploy the code to training and production I mark each file across so I know that it's been promoted.

QuizMate Development

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

	A	B	C	D	E	F	G	H	I	J	K
1	Code	Upload	Testing		Code	Upload	Training		Code	Upload	Production
2			scorekeepertesting				scorekeepertraining				scorekeeper
3			MatchSummary.php				MatchSummary.php				MatchSummary.php
4			scoreinput.php				scoreinput.php				scoreinput.php
5			scorepost.php				scorepost.php				scorepost.php
6			unlockroom.php				unlockroom.php				unlockroom.php
7			webincludetesting				webincludetraining				webinclude
8			dbadd.php				dbadd.php				dbadd.php
9			dbconnect.php				dbconnect.php				dbconnect.php
10	x	x	dbpull.php				dbpull.php				dbpull.php
11			dbscoreboard.php				dbscoreboard.php				dbscoreboard.php
12			dbscoreinput.php				dbscoreinput.php				dbscoreinput.php
13			dbteamresults.php								dbteamresults.php
14			adminkeys.php				adminkeys.php				adminkeys.php
15			adminkeysrecv.php				adminkeysrecv.php				adminkeysrecv.php
16			quizreports.php								
17			roomassist.php				roomassist.php				roomassist.php
18	x	x	config.php				config.php				config.php
19	X	X	utilities.php				utilities.php				utilities.php
20			scoreboardtesting				scoreboardtesting				scoreboardtesting
21			scoreboard.php				scoreboard.php				scoreboard.php
22			Tables				Tables				Tables
23			settings				settings				settings
24			QnumDefinitions				QnumDefinitions				QnumDefinitions
25	X	X	Teams				Teams				Teams
26			RoomKeys				RoomKeys				RoomKeys
27			CountOfTeamsPerLiveRoom								CountOfTeamsPerLiveRoom
28			Accounts				Accounts				Accounts
29	X	X	Schedules								Schedules
30	x	x	Rooms				Rooms				Rooms
31	x	x	ScheduleMatchupsTemp								ScheduleMatchupsTemp
32											
33			admintesting								admin
34			add.php								add.php
35			addrecv.php								addrecv.php
36	X	X	admin.html								admin.html
37			config.php								config.php
38			dashboard.php								dashboard.php
39			dbadd.php								dbadd.php
40	X	X	generateschedules.php								generateschedules.php
41			settings.php								settings.php
42			settingsrecv.php								settingsrecv.php
43			moves.php								moves.php
44			movesrecv.php								movesrecv.php
45			randommatchinsert.php								

Code Samples

Scoreinput.php ajax post and response

```
1342                $.ajax({
1343                    type: "POST",
1344                    url: "scorepost.php",
1345                    dataType: 'text', //This is what type of data you are expecting from the server.
1346                    data: {
1347                        postFunction: 'addrecord',
1348                        postQNum: getCookie("buzz_in_quizzer_selected"),
1349                        postPTS: getCookie("points"),
1350                        postQO: 0,
1351                        postFouls: getCookie("fouls"),
1352                        postRD: getCookie("round_num"),
1353                        postRoom: getCookie("room_name"),
1354                        postTeamNum: getCookie("buzz_in_team"),
1355                        postQuestionNum: getCookie("q_num")
1356                    },
1357                    success: function(data) {
1358                        //debug line.
1359                       //alert(data);
1360
1361                         data = data.trim();
1362
1363
1364                       if (data=== "1"){
1365                             alert ("Post failed due to ineligible quizzer. Things to check QO\'s or fouls for this quizzer or possibly the quizzer name.");
1366
1367                         }else{
1368
1369                             if (type === "Correct" || type === "NoResponse"||type=== "Wrong"){
1370                                     var cookie = parseInt(getCookie('q_num'));
1371                                     setCookie('q_num', cookie+1, .1);// Key, Value, Time
1372
1373
1374                             }
1375
1376
1377                         }
1378
1379
1380
1381                           window.location.href = "scoreinput.php";
1382                           //document.location.reload(true); //Older code that's slower.
1383                    },
1384                        error: function(jqXHR, textStatus, errorThrown) {
1385                            // Some errors that get returned
1386                            alert ("Error occured. Please try again.");
1387                            document.getElementById("confirm_btn").disabled = false;
1388                            document.getElementById("reread_btn").disabled = false;
1389                        }
1390                    });
1391                    return false; // I return false within the function so when I call it, it doesn't reload the page.
1392
1393            }
```

Dashboard.php self post with refresh interval controlled through a cookie.

```
412         $(document).ready(function()
413     {
414           var usersDashboardRefreshRate = getCookie('DashboardRefreshRate');
415           if (usersDashboardRefreshRate){
416                //set the refresh rate to the cookie so that the users selection is preserved.
417                document.forms[0].interval.selectedIndex = usersDashboardRefreshRate;
418           }
419           var usersDashboardRefreshSwitch = getCookie('DashboardRefreshOn');
420           if (usersDashboardRefreshSwitch){
421                //Sticky user preference.
422                var auto = document.getElementById("auto");
423
424                auto.checked = true;
425           }
426
427           handler();
428
429
430       });
```

Query to variables filling the datatables object.

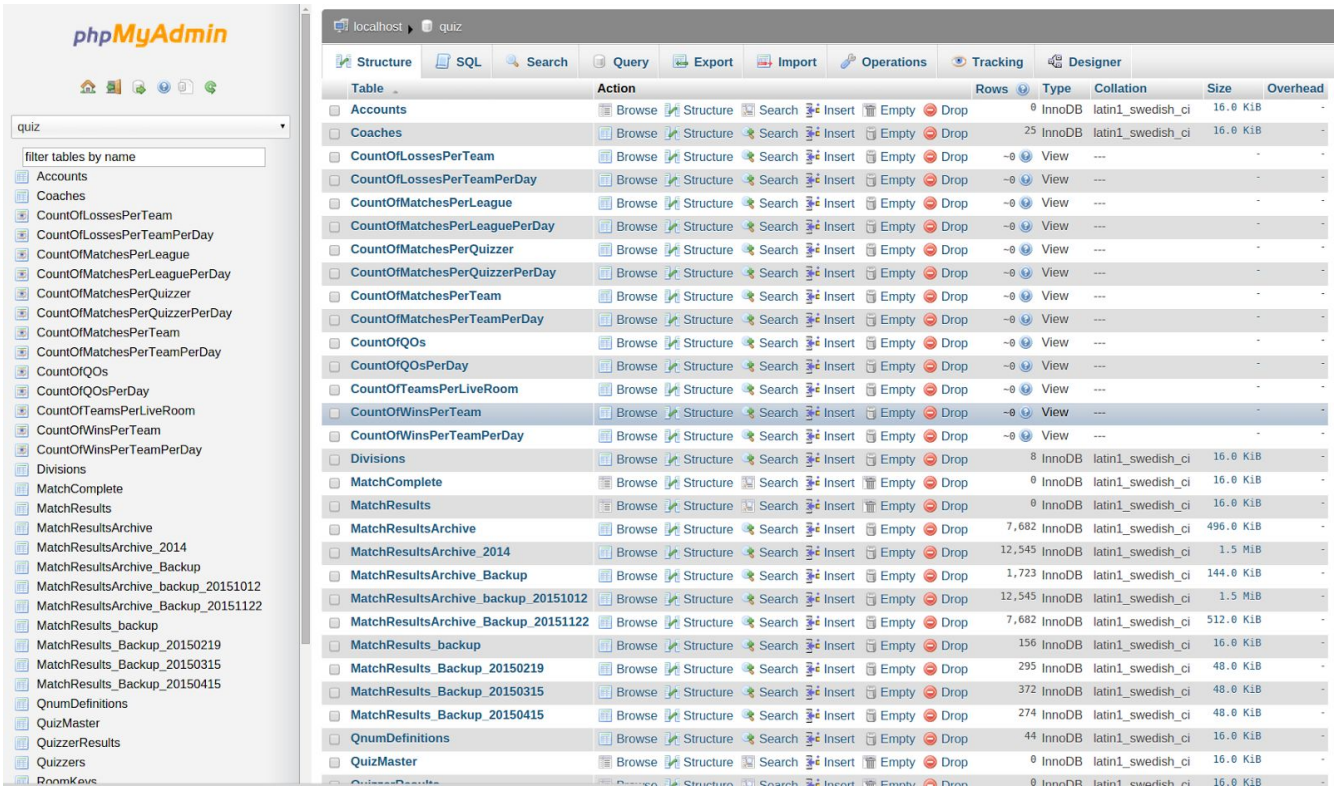
```
    foreach ($results as $ndx) {


        echo('<tr>');
        if ($results[$cnt2]['Def']=== 'Quiz Out'){      //QO detected so print edit and delete buttons.
                  echo('<td></td>');
                  echo('<td></td>');
        }else{   //No QO detected so print edit and delete buttons.
                  echo('<td><button class="btn" onclick="record_change(' . $results[$cnt2]['MRNum'] . ',' . $results[$cnt2]['QNum']
                  echo('<td><button class="btn" onclick="record_delete(' . $results[$cnt2]['MRNum'] . ',' . $results[$cnt2]['QNum']

        }

        //echo('<td style="text-align:center;" ><h4>' . $results[$cnt2]['QuestionNum'] . '</h4></td>');
        echo('<td style="text-align:left;"><h4>' . $results[$cnt2]['Def'] . '</h4></td>');
        echo('<td><h4>' . $results[$cnt2]['PTS'] . '</h4></td>');
        echo('<td style="text-align:left;"><h4>' . $results[$cnt2]["FirstName"] . '</h4></td>');
        echo('<td style="text-align:left;"><h4>' . $results[$cnt2]["LastName"] . '</h4></td>');
        echo('<td style="text-align:left;"><h4>' . $results[$cnt2]["Name"] . '</h4></td>');


        echo('<td><h4>' . $results[$cnt2]['QO'] . '</h4></td>');
        echo('<td><h4>' . $results[$cnt2]['Fouls'] . '</h4></td>');
       // echo('<td style="text-align:center;" ><h4>' . $results[$cnt2]['RD'] . '</h4></td>');
       // echo('<td style="text-align:center;" ><h4>' . $results[$cnt2]['Room'] . '</h4></td>');
        echo('<td style="text-align:left;">' . $results[$cnt2]['DateTimeStamp'] . '</td>');
        //echo('<td style="text-align:center;" ><h4>' . $results[$cnt2]['TeamNum'] . '</h4></td>');

        //echo('<td style="text-align:center;" ><h4>' . $results[$cnt2]['MRNum'] . '</h4></td>');
        echo('</tr>');

        $cnt2 = $cnt2 + 1;
    }
```

Some of the functions in the pull.php module.

```
28  function pullrooms() {...}
38
39  function pullactiveroomsandrounds() {...}
58
59  function returnteamname($teamnum) {...}
66
67  function pullteams() {...}
78
79  function pullteamsbydivision($division_name) {...}
95
96  function pullquizzerssbyteam($team_name, $teamnumbol) {...}
122 function pullallquizzers() {...}
174
175 function pullquizzerssbyid($qnum, $field) {...}
222
223 function pullquizzerssbyid_print($qnum, $field) {...}
271
272 function PullAndReturnRecordCount($query) {...}
308
309 function pulldivisions() {...}
319
320
321 function numofmatchesperdivision(){...}
327
328 function numofmatchesperdivisionperday($vardate){...}
345
346
347
348
349
350
351 function returnrecordsetasarray($query){...}
388
389
390
391 //Use this function to handle two fields. One for the key value and the other for the human value.
392 function runsql_backup($query, $fieldval, $fieldname) {...}
422
```

Database Interface

I use phpMyAdmin to administer the databases for QuizMate. phpMyAdmin is a standard for administering MySQL databases on LAMP stacks.



Testimonials

“Let me begin by saying that I love Quizmate and feel like it is one of the best tools for score keeping. It hasn’t always been this way though. I began using Quizmate in the fall of 2013. To say it was frustrating for me would be an understatement. It was clunky and didn’t seem to work. Josh worked hard at it though, listened to feedback that I and others brought, and made the necessary adjustments to make it right. Now, we have experienced a season with only one hiccup (that was completely out of the control of the software) and it has made a tremendous positive impact on my kids and leaders.

…”

Scott Berkey
Children’s Pastor
Victory Worship Center
scottb@vwcaz.org