Team Maman

Test Documentations

Validation, Integration, and Unit Tests

Aaron Andrews, Andrew Torgeson, Micheal Burton, Miles English, Nathan Sargeant, and Kami WIlson

4/19/2012

Test Case #	Req #	Test Steps	Expected Output	Comments	Pass/ Fail
1	3.1	1. Launch the application 2. Enter correct admin username “admin” 3. Enter correct password “password1” 4. Click “Log in”	The admin will be successfully logged in, and taken to the setup competition screen	User also needs to check the “admin” dialog box.	Pass
2	3.1	1. Launch the application 2. Enter correct admin username “admin1” 3. Enter incorrect password “wutevs@#$” 4. Click “Log in”	An error message will populate saying that the password is in-correct. Admin not logged in	An error message says Login failed	Pass
3	3.1	1. Launch the application 2. Enter incorrect username “!@#Joe” 3. Enter correct password “password1” 4. Click “log in”	An error message will populate saying the password is incorrect. Admin not logged in	An error message says “Incorrect username or pass-word.”	Pass
4	3.2	1. Get to the manage team screen 2. Enter team name in the team name box “vikings” 3. Click “create team”	Team should auto-populate in the “Edit team” section. Team values/members set to n/a	Works as described	Pass
5	3.2	1. Get to the manage team screen. 2. Enter a bad team name in the box. “team @#$@” 3. click “create team”	Error message should populate saying bad symbols are used.	Error message comes up, but it still adds the team	---
6	3.2	1. Get to the manage teams screen. 2. Choose team you want to manage “team 4” 3. Click modify 4. click “Add member” 5. Enter name “bobby” 6. Click “add member”	“bobby” should show on the team roster	Admin simply se-lects the student to add from a drop-down menu.	Pass
7	3.2	1. Repeat steps 1-3 2. Select player “bobby” 3. click “Delete Player”	Player “bobby” should not show on the team roster	Player is removed	Pass
8	3.3	1. Get to the Progress and Sta-tistics Screen	The complete list of teams should show with a list of perti-nent statistics for each player	Statistics only show as a team	Pass
9	3.4	1. Get to the Hints screen 2. Select a premade hint from the dropdown box. “1” 3. Press “send hint”	Notification will say hint has been sent.	Hints work	Pass
10	3.4	1. Get to the hints screen 2. Enter a hint in the box “Look at line 10!” 3. Press “Send Hint”	Notification will say hint has been sent.		Pass

11	3.5	Assign 6 problems for the competition	Cannot have more than 5 problems in a competition please change before starting.	Not possible to assign 6 problems, drop down only goes to 5.	Pass
12	3.5	Assign 5, 4, 3, 2, and 1 problems	Success	Any number works	Pass
13	3.5	Assign less than 1 problem	Cannot assign less than one problem	Drop down must be at least 1.	Pass
14	3.5	Assign no problems	Must select at least one problem	Can't assign 0 problems.	Pass
15	3.6	Create and view multiple competitions	Doesn't crash	Each competition needs it's own admin.	Pass
16	3.7.1	Set up a competition with invalid time such as -20 minutes	Cannot create a competition with a negative time	Message says "invalid time" competition won't start.	Pass
17	3.7.1	Create a competition with valid time such as 1 hour	Success	Works fine.	Pass
18	3.7.1	Create a competition with an invalid team name i.e one that doesn't exist	Team _____ does not exist cannot create competition	Can't create teams till competition starts.	Pass
19	3.7.1	Create a competition with a valid and invalid team	Team _____ does not exit cannot create competition		Pass
20	3.7.1	Create a competition with all valid teams	Success		Pass
21	3.7.1	Selecting problems see test cases 1-4	Success		Pass
22	3.7.1	Do not select a mode	Cannot create a competition that doesn't have a mode selected.		na
23	3.7.1	Select mode 1 create a comp Select mode 2 create a comp Select mode 3 create a comp	Success		Pass
24	3.7.1	Start a competition	Success		Pass
25	3.7.1	Stop a competition while it is running	Competition ends and students are notified		Pass
26	3.7.1	Create a competition with an invalid team, a valid time, and a valid mode	Team _____ is invalid cannot create a competition		Pass
27	3.7.1	Create a competition with a valid team, time, and mode.	Success		Pass
28	3.7.1	Create a competition with an invalid team, time, and mode.	Team _____ doesn't exist Time is invalid Invalid mode selected		Pass
29	3.7.1	There are many more combinations of valid and invalid test all combinations	If any invalid, then tell what is invalid and why else Success		Pass

30	3.8	View results of competition	Results of competition		Pass
31	3.9	Save competition results	Results of competition saved to file.	Not implemented yet.	Fail
32	3.11	1. Student navigates to the web page 2. Student clicks on the register button 3. Student selects "student" option 4. Student enters "billy15" in the username slot 5. Student enters "password5" in password slot 6. Student clicks "Register" button	The student will be redirected back to the log in page with the username and password already entered into the username and password sections. Username and password will be saved to the files on the server.		Pass
33	3.12 3.14	1. Student navigates to the web page 2. Student selects the "student" option 3. Student enters "billy15" in the username slot 4. Student enters "password5" in password slot 5. Student clicks log in	Student will be successfully logged in and taken to the ready check screen. The time until the competition starts is on the screen.		Pass
34	3.12	1. Student navigates to the web page 2. Student selects the "admin" option 3. Student enters "billy15" in the username slot 4. Student enters "password5" in password slot 5. Student clicks log in	Student is not logged in. "Invalid Username/Password" is displayed on log in screen.		Pass
35	3.12	1. Student navigates to the web page 2. Student selects the "student" option 3. Student enters “invalidname" in the username slot 4. Student enters "password5" in password slot 5. Student clicks log in	Student is not logged in. "Invalid Username/Password" is displayed on log in screen.		Pass
36	3.12	1. Student nav to the web page 2. Student selects the "admin" option 3. Student enters "billy15" in the username slot 4. Student enters "notapassword" in password slot 5. Student clicks log in	Student is not logged in. "Invalid Username/Password" is displayed on log in screen.		Pass

37	3.13	1. 1-3 Students log in to the system 2. Admin puts them on a team 3. Competition is setup and begins	All students are in the competition on the same problem. They can chat via the chat box.		Pass
38	3.14	1. Student is on ready check screen after logging in 2. Student looks at the clock	The clock shows time until competition starts	Implemented differently	na
39	3.14 3.18	1. Student clicks the tab "Bug Tester" 2. Student clicks the tab labeled "1" to enter input for Problem 1 3. In the box labeled "Test Input" student types "5". 4. In the box labeled "Expected Output" student types 25 (Correct input) 5. Student clicks the "Test For Bug!" Button.	Under the Results area, Bug 1 is reported. Bugs Found is incremented by one. If student's team has now found the most bugs, Student's team name appears in the Winning Team Section.		Pass
40	3.15 3.16 3.18	1. Student clicks the tab "Bug Tester" 2. Student clicks the tab labeled "1" to enter input for Problem 1 3. In the box labeled "Test Input" student types "5". 4. In the box labeled "Expected Output" student types "10" (Incorrect input) 5. Student clicks "Test For Bug!" Button	Under the Results area, "Bug not found" is reported. Bugs Found: does not change. Winning Team Section does not change.		Pass
41	3.15 3.16 3.18	1. Student clicks the tab "Bug Tester" 2. Student clicks one of the problem # tabs. 3. In the box labeled "Test Input" student types in bad input. Example: "8j4". 4. In the box labeled "Expected Output" student types Incorrect input. Example "Shoe" 5. Student clicks "Test For Bug!" Button	Under the Results area, "Bug not found" is reported. Bugs Found: does not change. Winning Team Section does not Change.		Pass
42	3.15 3.16	1. Student clicks the tab "Requirements" 2. Student clicks a Problem # tab to view requirements for selected problem.	Requirements appear for selected problem.	Shows on the main student screen.	Pass
43	3.15 3.16 3.17	1. Student types "Hello" into the message input box. 2. Student clicks the "Send Message" Button.	Student's username and typed message "Hello" appear in chat box.	Works as expected	Pass

44	3.16	1. Student clicks the tab "Bug Tester" 2. Student clicks the tab labeled "1" to enter input for Problem 1 3. In the box labeled "Test Input" student types "5". 4. In the box labeled "Expected Output" student types 25 (Correct input) 5. Student clicks the "Test For Bug!" Button.	Under the Results area, Bug 1 is reported. Bugs Found is incremented by one. If student's team has now found the most bugs, Student's team name appears in the Winning Team Section. Code is highlighted where the bug was found.		Pass
45	3.15	1. Student clicks on "Code Coverage": toggle button "On".	Error message reports that code coverage is not allowed in the current competition.	Error messages are annoying.	na
46	3.16	1. Student clicks on "Code Coverage": toggle button "On".	Code coverage is enabled. Any bugs that have been found will be highlighted in the code area for each problem.		Pass
47	3.16	1. Student clicks on "Code Coverage": toggle button "Off".	Code coverage is disabled. Any bugs that have been found will no longer be highlighted in the code area for each problem.		Pass
48	3.17	1. Admin selects Yes option under scroll bar selection at Competition Setup Screen.	Student is now allowed to receive hints from the Admin.	Hints work	Pass
49	3.17	1. Admin selects No option under scroll bar selection at Competition Setup Screen.	Student is not allowed to receive hints from the Admin. Hint option on the admin screen is disabled.		Pass
50	3.19 3.20	1. Enter correct input for test case: eg. "60", "60", "60" 2. Enter the correct expected bug output: eg. "Isoceles" 3. Enter correct input for test case: eg. "60", "60", "60" 4. Enter the correct expected Oracle output: eg. "Equilateral"	In the "Results" area the student will see how many bugs they found with that test case and the oracle's output: eg. Steps 1&2: "1 bug found" & "Oracle output is Equilateral". Steps 3&4: "0 bugs found" & "Oracle output is Equilateral".	Shows bugs found with a "Bugs Found" image. On the main student screen	Pass
51	3.21	1. Begin a competition 2. Look in the top left corner where the timer is located.	The timer will display the amount of time left in the competition.		Pass
52	3.22	1. The problems selected for the competition are written in C++ on the admin side. 2. The student opens a competition and view the problems.	The problem should be displayed in C++ on the student view. Everything else should work as it did in the Java code		Pass
53	3.22	1. The problems selected for the competition are written in Java on the admin side. 2. The student opens a competition and view the problems.	The problem should be displayed in Java on the student view.		Pass

54	3.24	1. One student begins a comp on a team 2. Student clicks the tab labeled "1" to enter input for Problem 1 3. In the box labeled "Test Input" student types "5". 4. In the box labeled "Expected Output" student types 25 (Correct input) 5. Student clicks the "Test For Bug!" Button.	The stats are updated and the team the student is on displays the highest score		Pass
55	3.24	1. Two students from two different teams begin a short competition. 2. The first team finds 5 bugs quickly. 3. The second team finds 5 bugs a slower.	The stats are updated and the first team should be displayed as the team with the High Score and the winner.		Pass
56	3.25	1. A student beings a competition 2. In the box labeled "Test Input" student types in bad data "123" 3. In the box labeled "Expected Output" student types "Whatever" (Bad Input) 4. Student clicks the "Test For Bug!" Button	No animation is displayed ant the bug count is not incremented.		Pass
57	3.25	1. A students begins a competition. 2. Student clicks the tab labeled "1" to enter input for Problem 1 3. In the box labeled "Test Input" student types "5" 4. In the box labeled "Expected Output" student types 25 (Correct Input) 5. Student clicks the "Test For Bug!" Button	Animation should be displayed and the bug count is incremented by one.		Pass

Formatted Table

Formatted: Centered, Position: Vertical: 0.84", Relative to: Page

Formatted Table

Team MAMAN: Integration Test Plan

We started out by doing a high priority implementation of the project meaning that we only implemented just enough to get the web site up and running and we filled in low priority features after the fact.

Aaron- Team Manager

Andrew- Programmer and documentation

Miles- Tester

Nate- Designer

Michael- Programmer

Kami- New Employee

We split up the work based on title but after about two weeks we realized that wasn't going to work since there was more programming than anything else. The was reorganized into a much more suitable format:

Aaron-Management/Student and Admin Implementation

Andrew-Bug Testing Implementation/Documentation

Miles-Java Code/Testing

Nate-Java Code/Graphics

Michael-Overall Website Development

Kami-New Employee

This worked much better and played to everyone's strengths. Everyone contributed to the shared files of the website. This made it so we could help each other with random bugs.

Unit Tests

Team Maman

Testing and debugging the code was mostly by-product of developing the code. When writing the code, we each made an effort to make sure that the section of code we created was working, so a large part of the testing was done implicitly. After the large portion of the project was finished, we got together in a group-session and weeded out all the remaining problems. With all of us together it was fairly simple to weed out all the remaining bugs. Our development style was a little divide-and-conquer, so not everyone understood everyone else's code. This wasn't really a problem, because even if you don't know how it was written, it's still fairly simple to find any bugs. Once the bugs are found, the expert on that section of code was able to fix it.

At the beginning of the semester we wrote out a series of validation tests. This strategy of problem-finding had limited usefulness. At the beginning of the project it was difficult to predict the different ways that things can go wrong. Many of the biggest bugs in the code weren't touched on by our pre-written validation tests. However they were useful in that they helped us remember certain errors that would only come up if the user is inept or malicious, for example a student trying to log in more than once, or an admin trying to create a competition with a negative number of minutes.

We believe the debugging process went very well, because it was a group effort, and it was done a little at a time, instead of all at once at the end.