




# Systems Testing Document

Team 7A(2)

## Authors

<u>Name</u>	<u>Main Section</u>	<u>Signature</u>
Grzegorz Lindert	Unit database tests, Component Testing, Logic, validation, database	
Tomasz Szelachowski	Final Document Creation, Usability	
Pawel Szymczyk	Unit test for build and userAdminAcoount, Stress testing, System testing, Logic, validation, GUI	

<u>Version</u>	<u>Authors/Editors</u>	<u>Description</u>
1.0	Tomasz Szelachowski	Document Created
2.0	Pawel Szymczyk Grzegorz Lindert	Added Unit testing results to the document
3.0	Tomasz Szelachowski	Added Usability testing results to the document
4.0	Tomasz Szelachowski	Edited Unit testing, Usability testing and started write up
5.0	Tomasz Szelachowski	Added to Write up
6.0	Pawel Szymczyk	Added Stress Testing and System testing
7.0	Grzegorz Lindert	Added component testing and tidied the document up

## **Test Process Followed**

### **Unit Testing**

By using the NetBeans unit test, the ID was set up to 1 as only the ID number increases in value, the names are simple strings set to appropriate testing parameters. We used inputs which were normal data and erroneous data and extreme data when appropriate. We then chose the Test name and wrote down the expected result for each test, we then run the specific test which came out with a result, if the result was the same as the expected it would be a pass otherwise a fail. If it fails, an explanation is created to then create a future plan how to fix the problem later on.

In the case of a fail, the problem was either solved or a plan was created to fix it in the future.

### **Functionality/Usability**

By running the program, we run tests to determine how easy the program is to use, how straightforward it was and how difficult the program was to navigate. We did this by having 5 people run the software and then found out how difficult and how complicated the software was.

We also run tests to prove the functionality of the program, by running the program and testing the different aspects of the software to prove that they work. We did this by setting out what the test was and then running it, writing down the result and provide a screenshot of the evidence. Which then showed us what was working and if there were any errors we hadn't found in earlier testing.

Any of the tests which failed were either fixed or plans were created to fix them in the future.

### **Component testing**

We carried out several component testing for each part of the system, these tests were carried out by writing test case for each part and then performing those task and recording the results

### **System testing**

For system testing, we looked backed at our requirements and asked several users to run our program and try to perform each one of functional and non-functional requirements and see if they can or can't do those task.

### **Stress Testing**

We performed Stress Testing for server and database, by running multiple one after the other connection to server and database in a short amount of time and recording the results of how well the program hands it and what are its limits.

.

## **Results**

### **Unit Testing**

The unit testing which we run on the system showed us that most of the tests we ran passed, meaning that everything is going to plan with the software development part. The tests did have a fail, however, the issue has now been flagged and the plan to deal with it is up and so the issue will be fixed.

The results of the testing are shown below in the Unit Testing section of the document in the testing tables.

### **Usability**

From the testing of the program we found that the program, was easy to use as both experienced and novice users had ease at going through the program, which meant that they should easily use the program without any kind of instructions from the programmers of the software. When asked the how straightforward and how difficult the program was to go through the tested user's response was that the functionality and the design of the navigation allow easy use of the program and that everything is straight forward.

They did, however, notice that more information should be provided on the actual types of pieces for example "what ram is" or "what a motherboard is". Which is on our list of things we need to add to the program. Any recommendations where taken into account and the possible ones have been added to the plan

Running the tests which are in the table at the back of the document, shows the tests which we ran to see which part of the program works and which does not. The test shows what the program can do. It shows that the program what the program looks like and what is the point on each page.

## Conclusion

Not everything in the program is complete and not everything works exactly to plan, thanks to the testing which was run, we have been able to assess the current state of the program which allowed us to create a plan for the future in what needs to be added, what needs to be removed and what needs to be changed for functionality reasons, usability reasons and proficiency reasons.

## Unit testing

**Class testing: ServerControl.java / Test class: ServerControlTest.java**

Methods:

- ConnectDB()
  - Connects to database returning established connection

Test name	Expected result	Actual result	Passed/fail
ConnectDB()	Should open connection with database and return false when calling connection.isClosed()	False	Pass

**Class testing: userAdminAccount.java / Test class: userAdminAccountTest.java**

Methods:

- checkPassword(username,password)
  - checks if inputted password matches password in database
- logInService (Username,password)
  - login in to program, sets username, first name, last name, email, mobile number, date of birth, account type
- usernameAvailability(username)
  - checks if inputted username when registering is taken
- reset()
  - resets users information, acts as a destructor
- getTableColName(table name)
  - returns given table columns and their data type
- getParts(tableName)
  - gets all parts from given table
- getBuilds()
  - gets all builds for logged in user or returns no builds when user has no builds
- UpdateUser()
  - updates users information that was altered by him like first name, surname etc.
- setUsername(String username)
  - Parameter "username" is set to "user"
  - Set username;
- setPassword(String password)
  - Parameter "password" is set to "password"
  - Set password
- setFname(String fname)
  - Parameter fname is set to "Kevin"
  - Set fname
- setSname(String sname)
  - Parameter sname is set to "Smith"
  - Set sname;
- setEmail(String email)
  - Parameter email is set to "smmith@yahoo.co.uk"
  - Set email

- setMobile(String mobil)
  - Parameter mobile is set to "0123423421"
  - Set phone number
- setDOB(String dob)
  - Parameter dob is set to "12091990";
  - Set date of birthday
- setType(boolean type)
  - Parameter type is set to true
  - Return type

#### Results userAdminAccountTest class

Test name	Expected result	Actual result	Passed/fail
checkPassword(user,user)	True, when inputted password matches with password in database along with username	true	Pass
loginService(user,user)	Should return User when calling getFname() after logging in as user and null if login failed	User null	Pass
usernameAvailability(user)	True if username is not taken	true	Pass
reset()	After calling reset() fname should be empty	fname is empty	Pass
getTableColName("Part")	False when calling isEmpty on results from getTableColName("Part")	false	Pass
getParts("PSU")	False, when checking is result is empty from calling getParts()	true	Fail, at time of testing PSU table is empty and results set was empty
getBuilds()	"No Builds", results should be "No Builds" as user at time of testing user didn't have any builds	"No Builds"	Pass
UpdateUser()	After changing user's first name to USER, calling updateUser() followed by reset() When user logged in again and using getFname() result should be USER also getSname after update should not change and be User	USER, update function works and updates users account details without affecting the rest of details	Pass
setUsername("user")	Set username to user	Set username to user	Passed
setPassword("password")	Set password to password	Set password to password	Passed
setFname("Kevin")	Set F name to Kevin	Set F name to Kevin	Passed
setSname("Smith")	Set s name to Smith	Set s name to Smith	Passed

setEmail("smmith@yahoo.co.uk")	Set email to smmith@yahoo.co.uk	Set email to smmith@yahoo.co.uk	Passed
setMobile("0123423421")	Set mobile to 0123423421	Set mobile to 0123423421	Passed
setDOB("12091990")	Det dob to 12091990	Det dob to 12091990	Passed
setType(true)	Set type to true	Set type to true	Passed

**Class to test: build.java / Test class: buildTest.java**

Methods:

- setBuildName(String name)
  - Parameter "name" is set to "setBuildName"
  - Set current build name;
- setMotherboard(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "Asus"
  - Set current id and motherboard name (1, "Asus")
- setCPU(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "AMD"
  - Set current id and motherboard name (1, "AMD")
- setRAM(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "KINGSTON"
  - Set current id and motherboard name (1, "KINGSTON")
- setStorage(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "WD"
  - Set current id and motherboard name (1, "WD")
- setGPU(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "MSI"
  - Set current id and motherboard name (1, "MSI")
- setPSU(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "Corsair"
  - Set current id and motherboard name (1, "Corsair")
- setPCCase(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "BeQuiet"
  - Set current id and motherboard name (1, "BeQuiet")
- setCooler(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "BeQuiet"
  - Set current id and motherboard name (1, "BeQuiet")
- setAccessories(int ID, String name)
  - Parameter: ID is set to 1
  - Parameter : name is set "mouse"
  - Set current id and motherboard name (1, "mouse")
- getBuilds()
  - retrieves given builds information and stores in build object

### Results buildTest class

Test name	Expected result	Actual result	Passed/fail
setBuildName(setBuildName)	Set name to "setBuildName"	Set name to "setBuildName"	Passed
setBuildName(null)	Set name to "setBuildName"	Set name to "null"	Failed- method should not allow to put null strings
setMotherboard(1)	Set id to 1	Set id to 1	Passed
setMotherboard("Asus")	Set name to "Asus"	Set name to "Asus"	Passed
setCPU(1)	Set id to 1	Set id to 1	Passed
setCPU("AMD")	Set name to "AMD"	Set name to "AMD"	Passed
setRAM(1)	Set id to 1	Set id to 1	Passed
setRAM("KINGSTON")	Set name to "KINGSTON"	Set name to "KINGSTON"	Passed
setStorage(1)	Set id to 1	Set id to 1	Passed
setStorage("WD")	Set name to "WD"	Set name to "WD"	Passed
setGPU(1)	Set id to 1	Set id to 1	Passed
setGPU("MSI")	Set name to "MSI"	Set name to "MSI"	Passed
setPSU(1)	Set id to 1	Set id to 1	Passed
setPSU("Corsair")	Set name to "Corsair"	Set name to "Corsair"	Passed
setPCCase(1)	Set id to 1	Set id to 1	Passed
setPCCase("BeQuiet")	Set name to "BeQuiet"	Set name to "BeQuiet"	Passed
setCooler(1)	Set id to 1	Set id to 1	Passed
setCooler("BeQuiet")	Set name to "BeQuietD"	Set name to "BeQuiet"	Passed
setAccessories(1)	Set id to 1	Set id to 1	Passed
setAccessories("mouse")	Set name to "mouse"	Set name to "mouse"	Passed
getBuilds(user, My First Build)	Should return "My First Build" when called getBuilds() followed by getBuildName()	Null pointer exception	Failed

Test getBuilds(user, My First Build) fails and gives an unexpected result as the user has not yet created a build under the name "My First Build", and there isn't any validation on the result set after the database query is executed. We will implement validation after the query is executed so that the user is prompted about the error and what should happen next.

## Component Testing

Test case for component testing on Login window:

- Enter wrong username and password, verify if any user-friendly warning messages is shown to user - Passed, user is informed with a user-friendly message
- Enter correct username and password and click on 'Login' button to be logged in and transferred to the main window - Passed, user is transferred to the main menu with his information logged in program
- Verify if clicking 'Exit' button closes the program - Passed, program exits successfully
- Verify if clicking 'Create account' opens a new window with registration form - Passed, user is transferred to new window where he can register as new user
- Verify if clicking 'Forgot Password' opens new window with a form to reset password - Passed, user is transferred to new window where he can reset password after confirming username and email

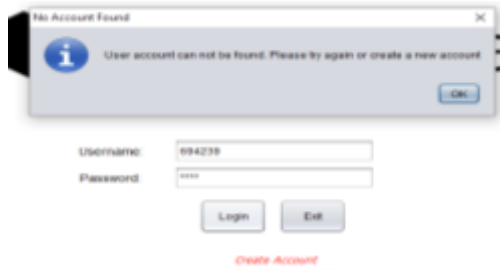
Test case for component testing on Users Menu window:

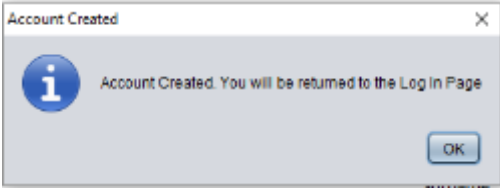
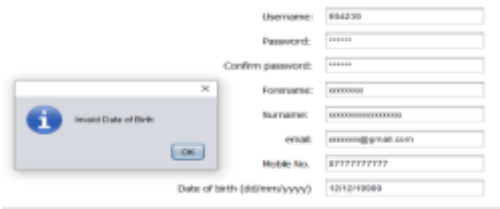
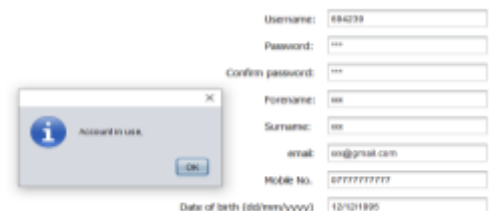


- Verify that clicking on 'Profile' button user is shown his profile information and can update these details - Passed, user is displayed with his personal information which he can edit by clicking on 'Edit' button
- Verify that users saved builds are displayed under 'Previous Builds' if there's no builds for current user 'No Builds' is shown - Passed, either build names are shown in a list or No builds is shown
- Verify that once clicked 'Log off' user is logged off and Login screen is displayed - Passed, user is logged off and application is ready for another user
- Verify that clicking on 'Add build' transfers user to build menu where he can start a new build - Passed, user is transferred to build menu where he can start a new build

Test case for component testing on Build Menu window:


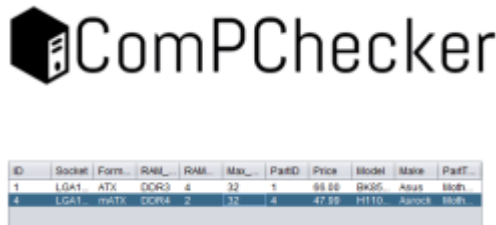

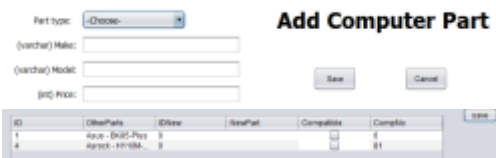
- Verify that user can't click accept until all parts chosen and light is green on all of them and build name is entered - Passed, accept button is disabled and once all buttons are green and build name is inputted, then user is able to click accept and save build.
- Verify that once clicked a button to choose a part only clicked parts are shown in table, for example, if user clicks Motherboard, the only motherboard are displayed and can be selected by user - Passed, by clicking on a button with a part a table with clicked part is displayed and user can choose one
- Verify that if a part is incompatible with the rest it turns red and if it's compatible it turns green - Passed, any part that is incompatible the background of it is red and if its compatible its background is green.
- Verify that user is able to input build name - Passed, user can input a build name
- Verify that 'Cancel' button goes back to menu - Passed, once clicked on 'Cancel' it returns user to menu

## Usability Testing

Test	Result	
<b>Login Page</b>		
If the user wants to attempt to log in, however input the wrong credentials, or a username and password that do not match.	<p>An error message is shown, showing the error that has been created and tells you what to do next.</p> <p>After pressing OK, you are returned to the window.</p>	

Create Account Page		
The user attempts creates an account.	When you succeed to create an account, a window pops up to confirm the creation which then takes you back to the login page.	
When creating an account and failing to input the correct dob, mobile no, email, name , password or username.	An error is shown specifically to show that the user has failed to input and prompts the user to input the correct information.  After pressing OK, you are returned to the window.	
When creating an account which is a duplicate of a account that already exists.	An error is shown specifically to show that the user has attempted to create a account which is already in use  After pressing OK, you are returned to the window.	
Profile Page		
The user wants to edit their profile.	The user presses the “edit profile button”, which then allows the editing of the profile and then pressing “accept” allows the edit of the profile.	
Menu Page		
The User wants to add a build	When the user presses the “add build” they are taken to the New Build Page.	
New Build Page		



When creating a new build the user adds a piece that isn't compatible.	When the user chooses a piece that isn't compatible, the button turns red.	
<b>Choose Part Page</b>		
The user wants to choose a component	The user Chooses the component that they want by double clicking it and then takes the user back to the new build page.	
<b>Admin Menu</b>		
The user is an admin and wants to login as a admin.	The admin menu is shown instead of the normal menu allowing the admin to do tasks.	
<b>Add New Part</b>		
The admin wants to add a new component.	The add computer part menu appears, where the admin types in the components details and then a new table shows up where the admin chooses compatibilities with other components	

## System testing

The test was carried out with 10 different users, they were given the program and asked to complete several tasks which tested the application's requirements, after each task they were asked a question, their results have been combined, results show which requirements are met and which not according to users.

### Results:

Functional Requirements:

- |  |          |
|--|----------|
| 1. User is going to register to program                          | - passed |
| 2. User is going to log in to the system as 'admin' or as 'user' | - passed |
| 3. User cannot input symbols in username                         | - passed |
| 4. User cannot see own password (during writing) - safety        | - passed |
| 5. User can find motherboard Asus BK85-Plus                      | - passed |
| 6. User can find list of components in motherboard type          | - passed |

- |   |   |
|---|---|
| 7. User can select component Asus BK85-Plus and add to build                    | - passed  |
| 8. User get information if cannot use components not compatible with each other | - passed  |
| 9. User easily can find out price of component Asus BK85-Plus                   | - passed  |
| 10. User can easily find out help button  | - failed - there is no available function   |
| 11. User can find all component specification                                   | - partly passed (all users found specification but 4 complained about the layout - is not too readable) |
| 12. User can save own build   | - passed  |
| 13. User can compare prices   | - failed, there is no function allow to do that   |
| 14. User can see Asus BK85-Plus picture   | - failed, there is no function  |
| 15. User can check own build  | -passed   |
- Non-functional requirements:
1. How easy system is to use - all users were happy, they marked usability for 70% - there were some issues with layout of component specifications, there are no components pictures
  2. User cannot be login as admin, and admin cannot login as user - passed

## Stress Testing

Stress testing was carried out test connection to server and database. Connection test was simulated for 10, 50, 100, 200, 300, and 500 users connection one by one. This test was carried out by implementing a for loop to iterate through and connect multiple times.

Connections tested for userAdminAccountTest:

Results:

All test from 10 up to 500 users passed, we could not conduct more than 500 tests, because of physical hardware and software limitations.

Connections tested for ServerControlTest:

Results:

156 users - passed tests

After 156 connection program failed, could not perform 157 connection

This test shows us that the limit of our program is 156 user, so in one time 156 user can use the program with no problem after that there are some issues as the server can't handle any more connection.