A coat of arms with white birds and a book

Description automatically generated

**University Of Bradford**

**Department of Computer Science**

**Testing Document**

*Workflow management system for non-crime*

*related activity for Yorkshire and Humber Regional Organised Crime Unit (YHROCU)*



**Team 1**

**Team 1**

**Project Choice:** P1. Workflow management system for non-crime related activity

17003493 - Sania Bibi ([s.bibi60@bradford.ac.uk](mailto:s.bibi60@bradford.ac.uk))

22031400 - Javairia Shahid ([j.shahid2@bradford.ac.uk](mailto:j.shahid2@bradford.ac.uk))

20010596 - Amal Abeso Ela ([a.m.abesoela@bradford.ac.uk](mailto:a.m.abesoela@bradford.ac.uk))

22006776 - Imara Ali ([i.ali72@bradford.ac.uk](mailto:i.ali72@bradford.ac.uk))

21045121 - Oluwadamilare Falade ([o.s.falade@bradford.ac.uk](mailto:o.s.falade@bradford.ac.uk))

22036203 - Shafeeq Shuaib ([s.shuaib@bradford.ac.uk](mailto:s.shuaib@bradford.ac.uk))

**Contents**

1. **Acceptance testing.........................................................................................4**
2. **Code inspection.............................................................................................8**
3. **PHP Unit testing............................................................................................12**
4. **Relevant links to Github................................................................................12**
5. **Peer review..................................................................................................12**

**Acceptance testing**

|  |  |  |  |
| --- | --- | --- | --- |
| Input sequence | Expected result | Current output | Comment |
| Requirement 1: The user should be able to log in with a 6-digit staff number and password | | | |
| Correct login for normal user:  Staff no. = 1234  Password = 1234 | User logs in successfully | A screenshot of a computer  Description automatically generated | As the user has logged in successfully, they have been directed to the dashboard. |
| Incorrect login  Staff no. = 5643  Password = wrong | User does not log in successfully |  | As the details are incorrect the page does not re-direct anywhere and the search bar shows “invalid\_credentials”. |
| Requirement 2: The user should be able to create a task | | | |
| Creating a task successfully  Filling in unique details to create a task | The task should appear in the database | A screenshot of a task  Description automatically generated  A screenshot of a black box with white text  Description automatically generated | The task has been created successfully. The three screenshots show the various stages. First filling in the form, receiving confirmation and the last screenshot shows the task in the database. |
| Creating a task unsuccessfully  Attempting to create a task which already exists within the database | The task should not appear in the database | A black screen with white text  Description automatically generated | The task has not been created and the user has received an appropriate message so they can amend task details to retry. |
| Requirement 3: The admin should be able to delete a task | | | |
| Deleting a task successfully  The admin password is 1234 to delete tasks. | The task should be deleted and not exist in the database. |  | The user will input the admin pin to remove the task then the task will be deleted and will not appear in the list of tasks or in the database. |
| Deleting a task unsuccessfully  The admin password is inputted incorrectly. Wrong password = 1278 | This task will not be deleted as it is not an admin who is deleting it. |  | The task will not be deleted as the incorrect pin was entered. This will stop any unauthorised personnel from deleting tasks. |
| Requirement 4: When signing up the users: email address, staff number, team, first name, surname should be recorded. | | | |
| Correct Signup  Input staff details correctly – Filling in uniquely. | A new staff member should be created and be able to log in to the system. |  | A user account created message will appear to indicate to the user that it was successful |
| Incorrect Signup  Signing up using duplicate details. | The user will not be created as there is already an account with the same details. |  | A user has not been created as this will be create a duplicate user due to the details being the same therefore the system will stop this. |
| Requirement 5: An option to show who the tasks can be viewed by | | | |
| Task assigning Shows who this task is assigned to and therefore who can view it.  Staff ID = “1234” | The Staff ID number for the task is assigned to will be displayed or if it is assigned to all staff then everyone will be displayed. |  | Whoever a task is assigned to will be the only person who will be able to view it and the admin too. |
| Requirement 7: Enable the user to input progress updates to a rolling log. | | | |
| Adding updates correctly.  Adding an update to a task = “Part 1 of task is complete” | The updates are added to the rolling log of the task and can be viewed |  | User can put updates into a rolling log so that task progress can be checked. |
| Requirement 8: User should be able to see their assigned tasks. | | | |
| Personalised table for each user  The user can only see tasks which are assigned to them. Logged in as staff ID = “1234” | The user will be able to see a table with all the tasks they have. |  | The user will also be able to see tasks which have been assigned to everyone as this will not be specific to one person but will be for all staff members to complete. |
| Requirement 9: Supervisor should be able to update the due date for tasks | | | |
| Changing due date  Task due date can be inputted if the assignee needs more time.  Change due date to = “01/05/2024” | The due date of the task will change to 01/05/2024 and will be displayed as this on the table. |  | The due date can only be changed by a supervisor as they will need to enter their supervisor number. |
| Requirement 10: Should be able to export PDFs of tasks. | | | |
| To assess task progress or completion a PDF can be exported for records. | A PDF report of the tasks should be generated. | A screenshot of a document  Description automatically generated  A screenshot of a computer  Description automatically generated | When the user clicks the button “Export to PDF” a PDF file of the tasks gets generated. |
| Requirement 11: Only a supervisor can say a task is complete. | | | |
| Complete task  Change task completion to successfully. | The task will be changed to Y on the dashboard and will therefore be displayed as complete | A screenshot of a task  Description automatically generated  A screenshot of a computer  Description automatically generated  A white square with black text  Description automatically generated | Only a supervisor can mark the task as complete. The image shows correct insertion of the supervisor pin leading to successful completion of the task. |
| Complete task  Change task completion unsuccessfully. | The task will not be marked as completed as the supervisor pin was incorrect. | A screenshot of a computer  Description automatically generated | The task was not marked as completed as the supervisor pin was incorrect. |
| Requirement 11: Able to search for tasks | | | |
| Search correctly  Search for task that exists: “call john” | The task that was searched for should appear. |  | Anybody can search for a task |
| Search incorrectly  Search for task that does not exist: “hello” | No task should appear. |  | Anybody can search for a task |

**Code inspection**

|  |  |  |  |
| --- | --- | --- | --- |
| Inspecting | Expected result | Current output | Comment |
| Correctly naming elements (Example 1)  Task complete button | taskCompleted, completeTask, uponCompletion |  | Input fields and buttons have been correctly named so they can be identified easily. |
| Correctly naming elements  (Example 2)  Navigation bar | menuItems, navigationBar, navBar | A computer screen shot of text  Description automatically generated | Classes for the navigation bar have been correctly named. |
| Using comments | Systematic use of comments throughout code | A screen shot of a computer code  Description automatically generated  A computer screen with text and images  Description automatically generated | Screenshots from three different files, all showing systematic and consistent use of comments throughout. |
| Using correct coding conventions such as camel casing | Correct use of camel casing when naming elements | A computer code with text  Description automatically generated with medium confidence | Systematic and correct use of camel casing throughout code. |
| No harsh use of colours. | Colours are expected to flow well and are easy on the eyes. |  | Appropriate use of colours throughout e.g., green for submit/success and red for failure/delete. Other colours flow well and do not clash with each other or other elements such as font. |
| Appropriate use of font styles. | No elaborate use of styles such as handwriting. |  | Clear and easy to read font styles have been used. |
| Appropriate use of font sizes. | The font sizes should not be too large or too small. |  | Font sizes have been used appropriately, headings large, sub-headings slightly smaller and so on. |
| User experience and useability. | The site should be easy to navigate and self-explanatory. |  | Menu items in the navigation bar are self-explanatory. All elements/buttons have been labelled clearly. |
| Are login details being sent securely to the server. | Login details are not visible in the search bar and are not showing in inspector tools. |  | As we can see only the invalid credential shows in the navbar. When the user enters correct details, they are automatically redirected. |
| Error handling mechanism (Example 1) | Trying to sign up a user which has a staff number that already exists in the database. The user should not be created. |  | Appropriate error message appears when trying to sign up a user which already exists in the database. |
| Error handling mechanism (Example 2) | Trying to add a task which already exists in the database. The task should not be added. |  | Appropriate error message appears when trying to create a task which already exists. |
| Missing input handling | Leaving signup/login fields blank. |  | The user is prompted to enter the missing details. |
| Clearing cache after the user signs out so content is not accessible | Content should not visible after the user signs out as this would violate security. |  | Screenshots show cache is being cleared and session variables being reset and destroyed. This provides security to ensure logged in user cannot see previous users’ data. |

**PHP Unit testing**

|  |  |
| --- | --- |
| Test case 1: Log out successful | |
| Function being tested (Logout successful) |  |
| Test case of function | **A screen shot of a computer program  Description automatically generated** |
| Assertion result | **A screenshot of a computer program  Description automatically generated** |
| Explanation | The logout function unsets the session variable, destroys the session and returns true when successful otherwise it returns false. The test case confirms this by using **assertTrue** and passing the $result/logout function as a parameter. **The assertion shows that this has been executed successfully**, however there is a notice which is indicative of minor issues that do not affect the code. |

|  |  |
| --- | --- |
| Test case 2: Log out unsuccessful | |
| Function being tested (Logout unsuccessful) | **A screen shot of a computer program  Description automatically generated** |
| Test case of function | **A computer screen shot of a program code  Description automatically generated** |
| Assertion result | **A screenshot of a computer  Description automatically generated** |
| Explanation | The logout function unsets the session variable, destroys the session and returns true when successful otherwise it returns false. The test case in this scenario does not set a session variable initially meaning no user is logged in... if a user is not logged in then they cannot possibly log out. **The assertion is successful**. |

|  |  |
| --- | --- |
| Test case 3: Connecting to the database unsuccessfully | |
| Function being tested (Connecting to the database unsuccessfully) | A screen shot of a computer program  Description automatically generated |
| Test case of function | **A screen shot of a computer program  Description automatically generated** |
| Assertion result | **A screenshot of a computer program  Description automatically generated** |
| Explanation | **The assertion is successful**. The function here connects to the database, and if it the connection is successful it returns true, if the connection is unsuccessful, it returns false. The test case takes advantage of this and checks **assertFalse**, it tests whether the function returns false (meaning no connection). |
| Test case 4: Connecting to the database successfully | |
| Function being tested (Connecting to the database successfully) | A screen shot of a computer program  Description automatically generated |
| Test case of function | A screen shot of a computer program  Description automatically generated |
| Assertion result | **A screenshot of a computer  Description automatically generated** |
| Explanation | **The assertion is successful**. The function here connects to the database, and if it the connection is successful it returns true, if the connection is unsuccessful, it returns false. The test case takes advantage of this and checks **assertTrue**, it tests whether the function returns true (meaning a connection has been established). |

|  |  |
| --- | --- |
| Test case 5: Confirm if user is logged in successfully | |
| Function being tested (Confirm if user is logged in successfully) | **A computer screen with colorful text  Description automatically generated** |
| Test case of function | **A computer screen shot of text  Description automatically generated** |
| Assertion result | **A screen shot of a computer program  Description automatically generated** |
| Explanation | **The assertion is successful**. The function here is important as some of the content on the website is restricted to logged in users. This assertion sets a session variable to mimic a logged in user, it then runs the function and asserts if it is true. |
| Test case 6: Confirm if user is not logged in | |
| Function being tested (Confirm if a user is not logged in) | **A computer screen with colorful text  Description automatically generated** |
| Test case of function | **A screen shot of a computer program  Description automatically generated** |
| Assertion result | **A computer screen shot of a program  Description automatically generated** |
| Explanation | **The assertion is successful**. The function here is important as some of the content on the website is restricted to logged in users. This assertion sets the session variable to null/empty, that means the user is not logged in. It passes the empty session through the function we’re testing. As the function returns false because no user is logged in the assertFalse is correct. |

1. Link to: meeting minutes
2. Link to: <https://github.com/Team-1-Enterprise-Pro/YHROCU/tree/238101853232910405407367ad5cf13580219f0b/Coursework>
3. Link to unit testing files in GitHub: <https://github.com/Team-1-Enterprise-Pro/YHROCU/tree/238101853232910405407367ad5cf13580219f0b/unitTesting>

**Peer Review**

Summary: All members contributed equally and achieved tasks to a good standard. All members have a score of 10.

17003493 - Sania Bibi

* Tasks allocated: Added and developed an extra feature which had not been completed in coursework one, this was allowing users to download all of the task data as a PDF with one click of a button. Also did the Unit testing. Worked alongside Imara.
* Achievements: Developed the feature to a good standard and made sure it worked successfully. Found the Unit testing difficult but managed to make it work after many hours and implemented test scenarios with Imara. Also organised GitHub.
* Score of contribution: 10

22006776 - Imara Ali

* Tasks allocated: The client requested in the demonstration that logged in users should only see tasks specific to them, and not just a generic list of all the tasks in the database. Also worked on the Unit testing. Worked alongside Sania.
* Achievements: Logged in users now see two tabs, one for all the tasks and then one for tasks specific to them, this is just how the client requested. Unit testing was something we struggled with but worked alongside Sania and eventually managed to implement test scenarios.
* Score of contribution: 10

22031400 - Javairia Shahid

* Tasks allocated: Add supervisor and admin pins as JS functions to specific parts of the website as a security measure so not everyone can see confidential information. Also worked on the code inspection. Worked alongside Shafeeq.
* Achievements: The signup page, deleting tasks and marking a task as complete all now have appropriate restrictions and error handling e.g., if someone enters an incorrect pin. Code inspection completed with 10+ cases.
* Score of contribution: 10

22036203 - Shafeeq Shuaib

* Tasks allocated: Add supervisor and admin pins as JS functions to specific parts of the website as a security measure so not everyone can see confidential information. Also worked on the code inspection.
* Achievements: The signup page, deleting tasks and marking a task as complete all now have appropriate restrictions and error handling e.g., if someone enters an incorrect pin. Code inspection completed with 10+ cases.
* Score of contribution: 10

20010596 - Amal Abeso Ela

* Tasks allocated: Worked through the CSS and HTML and updated the pages searchTask, tasks and normalUser so they were more presentable. Also did acceptance testing. Worked alongside Dami.
* Achievements: All the pages look good now and meet the clients’ requirements in terms of presentability and also cater to neurodivergent individuals as no harsh colours or elaborate font styles have been used. Completed the acceptance testing with 15+ cases.
* Score of contribution: 10

21045121 - Oluwadamilare Falade

* Tasks allocated: Worked through the CSS and HTML and updated the pages searchTask, tasks and normalUser so they were more presentable. Also did acceptance testing. Worked alongside Amal.
* Achievements: All the pages look good now and meet the clients’ requirements in terms of presentability and also cater to neurodivergent individuals as no harsh colours or elaborate font styles have been used. Completed the acceptance testing with 15+ cases.
* Score of contribution: 10