# Assignment 3

The overall testing strategy is focused on the inner workings of each element and characteristic of the web app. The report explains some of the system tests (user story testing) and the software tests (focused on Java/JS).

The approach for testing:

* add a test case for all methods to ensure correct input usage and that the output is what was expected
* use Postman for API testing
* simulate events and check if they were successfully added to the database and the web app to test the integration of different components and how well they perform together
* add system testing for the functional requirements

### System tests:

| Test Report for S1 |
| --- |
| S1 : Verification of correct login credentials  User story: As a Crew Member/Admin I want to be able to login to my dashboard. |
| Expected behavior: The web app should redirect the user to their corresponding dashboard if the username and password are correct, otherwise an alert should be displayed on the screen stating that wrong credentials have been used. |
| Testing result:  Step 1: Go to http://shotmaniacs\_war/login.html  Step 2: Input the wrong credentials for the username or password  Step 3: Press ‘OK’ on the pop-up that states wrong credentials have need used    Step 4: Input the right password and username for the admin dashboard  Step 5: Check if the redirected page is the admin dashboard |
| Unexpected behavior :   * after step 2 the user is redirected to a different page * after step 4 the user is still getting the error for the wrong credentials * after step 4 the user is redirected to the crew dashboard |

| Test Report for S2 |
| --- |
| S2 : Uploading the xlsx file content to the database  User story: As a client I want to be able to see a template describing the format of the csv file to fill out instead of submitting each event from the event form. |
| Expected behavior: The web app should be able to accept only a xlsx file and let the user submit the file. The xlsx file should be read line by line and the information from the cells added to the database and from the database to the admin dashboard as a new event (not yet accepted by the admin). |
| Testing result:  Step 1: Go to http://shotmaniacs\_war/  Step 2: Scroll down to ‘Book now’  Step 3: Download the ‘template.xlsx’ file and fill it in with a number of events  Step 4: Press the ‘choose file’ button and add the ‘template.xlsx’ file and press ‘submit’    Step 5: Login as an administrator and go to the dashboard  Step 6: The new event should appear in the ‘Bookings’ section |
| Unexpected behavior :   * After step 4 the file does not submit and the user is redirected to the page shotmaniacs\_war/api/event/upload where you get a 500 error      * Step 6 is false |

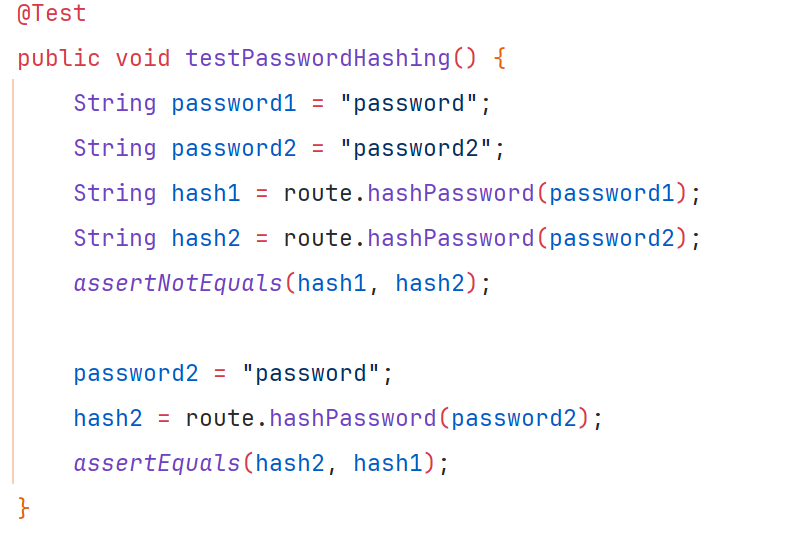
| Test Report for S3 |
| --- |
| S3 : Modifying the user’s permission  User story: As an administrator I want to be able to modify the permissions for each user taking into consideration the restrictions of the system. |
| Expected behavior: The web app should allow only the admin to change the role of each user without disturbing the pre-established regulations imposed on the admin and crew member position. |
| Testing result:  Step 1: Go to http://shotmaniacs\_war/admin-dashboard.html  Step 2: Find the ‘Services’ button on the top of the page  Step 3: Click on ’Services’ and press the ‘Permissions’ button    Step 4: A new pop-up should open in the middle of the page stating the name of the user on the left, the current role and the updated role    Step 5: Search for a user who is a not production manager  Step 6: Change the job of the user to a different job that is not production manager  Step 7: Enter the ‘Permissions’ pop-up again and the user’s current job should be changed now (ex: from other to photographer)  Step 8: Change the role of the user again to ‘Production manager’  Step 9: The user should get an alert stating that the member is not an administrator and the role can not be changed |
| Unexpected behavior :   * The pop-up does not show the details mentioned * After step 8 the user is allowed to change the job of the member to a production manager |

### Software tests:

This first test is made to ensure that the class ’DatabaseConnection’ establishes a connection to the PostgreSQL database and is able to execute a simple query to retrieve all the content of the ‘event’ table. The query should return at least one row. We handle any SQL exception with ‘fail’ and an informative message that lets us know the database connection failed.

Positive implication: If the test passes, it indicates that the ’DatabaseConnection’ class has a correct configuration and can retrieve information from the database.

Negative implication: If the test fails, it could indicate that the credentials for the database are wrong, the database server is unresponsive/unreachable, or the ‘event’ table does not exist or is empty in the database.



This method is part of the AuthTest class which has the purpose of ensuring the correct functionality of the ‘AuthRoute’ class. The ‘AuthRoute’ class is responsible for the user’ authentication and registration in the web app.

The first test covers the method ‘hashPassword’ and checks that two different passwords do not result in the same hash and that the same passwords generate the same hash.



The second test is focused on the format of the password used. The password has to respect the following restrictions: at least 1 lowercase and 1 uppercase character, no white space, at least 1 special character and at least 1 digit. The approach is to go through different password words to check if they respect the rules or not.



The third test ensures that the JWT is generated. We have to generate a token for each given username and role. We then parse the token and check if the claims match the input values that we set in the beginning. We also check that the claims have a value for when they are issued and when they expire.

Positive implication: If the test passes we ensure that the ‘AuthRoute’ class correctly handles the user authentication, hashing passwords (uniquely) for security reasons, and verifies the password format and that the JWT are generated with correct timestamps and claims.

Negative implication: If the test fails it could indicate that the hashed passwords are inconsistent/ repetitive, weak passwords are accepted regardless of the rules and the JWT are not secure tokens.