Final Test Report (Music Albums Archive)

Abdullah Abdullah, Naufil Ansari, Jinthushan Sutharman, Izaan Syed, Thomas Petkovic

Introduction

The purpose of this report is to outline the extensive testing that the program has been put through. The report will include a full breakdown of the tests performed as outlined in the Test Plan for Album Atlas, as well as their results.

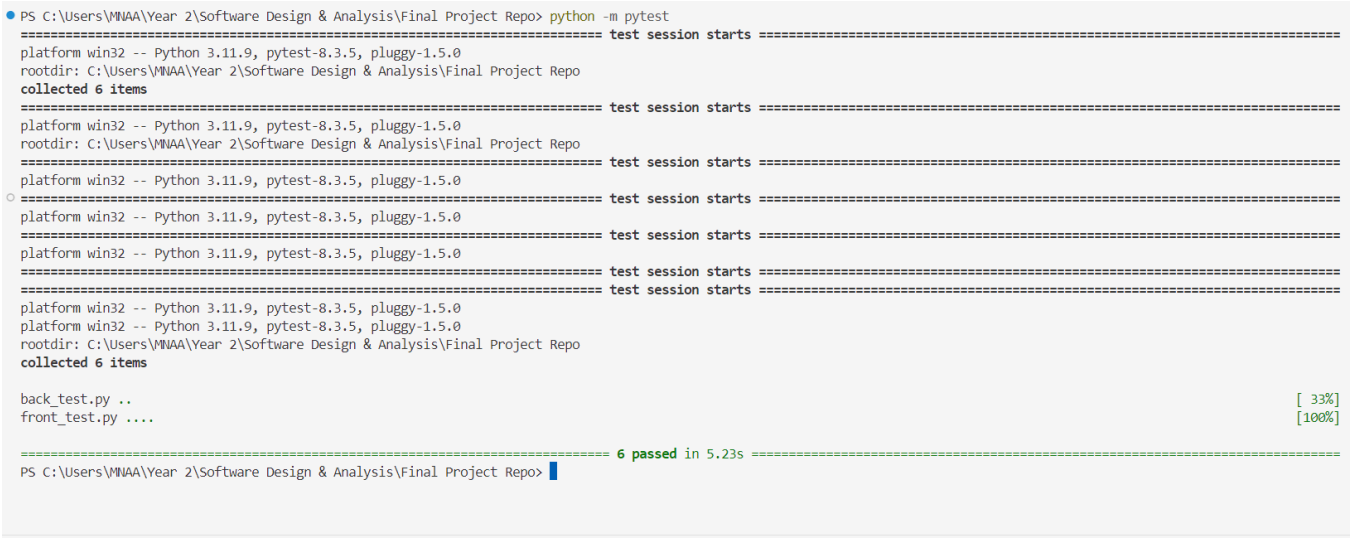
Unit Tests

Overview:

Test ID	Components Involved	Preconditions	Steps	Expected Result	Testing Approach	Assigned Team Member
UT-01-CB	Registration System	None	Call validate_register with valid and invalid inputs	Valid inputs return True, invalid inputs return False	Pytest	Thomas
UT-02-CB	Login System	None	Call validate_login with valid and invalid inputs	Valid inputs return True, invalid inputs return False	Pytest	Thomas
UT-03-CB	Song Input	None	Call validate_inputs with various song details	Valid inputs return True, invalid return False	Pytest	Naufil
UT-04-CB	Dupe Username	Duplicate usernames must be present	Call is_username_taken with different usernames	Existing usernames return True, new ones return False	Pytest	Izaan

UT-05-CB	Song Search	Song data available	Call search_item with search queries	Matching song found or error message shown	Pytest	Naufil
UT-06-CB	Lyrical Preview	Lyrics available or missing	Call lyrical_preview with song names	Returns correct lyrics or "No lyrics available"	Pytest	Jin

Results: As the unit tests were fairly straightforward, all of them were able to be automated and tested using Pytest, a Python framework designed for unit testing. As seen by the terminal image below, all tests implemented passed.



Integration Tests

Overview:

Test ID	Components Involved	Preconditions	Steps	Expected Result	Assigned Team Member
IT-01-TB	Window and UI Rendering	Application is launched	Open the application. Observe if all	Main window loads successfully.	Abdullah

			UI components are displayed properly.	Search bar, buttons, and song list are visible and functional.	
IT-02-TB	Search Bar, Song Database	Application is launched	Type a valid name in the search bar and press Enter. Observe if the results update correctly. Clear the search bar and press Enter. Type a non-existent name and press Enter.	Matching results appear in the song list. Clearing the search restores the full list. Non-existent name shows an error message or empty results.	Naufil
IT-03-TB	"Add Item" Button, Entry Fields, Save Button,	Application is launched	Click the "Add Item" button. Enter valid data in the input fields. Click the "Save" button. Repeat with missing/invalid data.	With valid data: Song appears in the song list. With invalid data: Warning or error message appears.	Jin
IT-04-TB	"Edit Item" Button, Entry Fields, Save Button,	Application is launched, there is a pre-existing song to edit	Select an item from the song list. Click the "Edit" button.	Changes are reflected in the song list. Saving without changes	Izaan

			Modify the fields. Click "Save" and check if changes apply. Try saving without making changes.	does not modify data.	

Results: Since the integration tests were larger in scale, they needed to be manually tested to ensure functionality.

- IT-01-TB: Window launched properly, all components displayed successfully
- IT-02-TB: Search bar filters results based on input field, invalid inputs don't show anything
- IT-03-TB: Add Item validates data properly, throws error if not valid, adds song to list otherwise
- IT-04-TB: Edit function saves changes appropriately, if unchanged, does not affect song

All the integration tests passed and the program performed up-to-standard during the tests.

System Tests

Overview:

Test ID	Components Involved	Preconditions	Steps	Expected Result
ST-01-OB	UI, Login System, Sort System, Add functionality, Edit functionality, Delete	The application is installed and running. A test user	1. Login System Open the application. Enter valid	Login should work with no error message Songs should only be added to the

	functionality, Logout system,	account exists (Thomas1 / 1234).	credentials (testuser / testpass). Click the Login button. 2. Adding a New Item Click the Add Item button. Fill in the form with valid details (e.g., Item Name: "Test Song", Category: "Music"). Click Save. 3. Searching for an Item: Type "Test Song" into the Search Bar. Press Enter. 4. Editing an Item: Select "Test Song" from the UI. Click Edit. Modify the category from "Music" to "Favorite Music". Click Save. 5. Sorting Items:	current users' database. Search should only find songs in the current users' database. Editing a song should only affect the specific song in the current user's database Item sort should work regardless of database chosen Deleting a song should only affect the specific song in the current user's database Logout should reset chosen database to default Application should not crash or hang

			Click on the Category Column Header in the UI. Observe the item order. Click again to change sorting direction. 6. Deleting an Item: Select "Test Song" in the UI. Click Delete and confirm. 7. Logging Out: Click the Logout button.	
ST-02-OB	UI, Login System, Add functionality, Lyrical Preview	The application is installed and running.	1. Create new user account 2. Log into new account 3. Add new (real) song 4. Check lyrical preview	Registration should create new account, login should be performed, song should be added to DB, lyrical preview should show real lyrics
ST-03-OB	UI, Login System, Add functionality, Image addition/preview	The application is installed and running.	1. Log in to admin account 2. Add new	Login should work with no error message Song should be

		An admin account exists. A sample image exists.	(real) song 3. View details and upload image of the album 4. View details and verify image	added to all databases, including guest database Image should upload with no problem (admin privilege) Song should have cover art of the album visible to all users and guests

Results: As with the integration tests, the system tests are performed on too large of a scale to automate; thus, developers needed to recreate the exact scenario and ensure the program outputs the expected behaviour.

- ST-01-OB: Login worked as expected, all operations (search, edit, delete) existed only within the user's database, logout worked as expected, upon logout, the system database reverted to guest data.

- ST-02-OB: Registration and login were performed successfully, item was added, lyrical preview generated a lyrics list of the song (assuming it was a real song; dummy objects will not work)

- ST-03-OB: Login was successful, song was also successfully added, image was connected to the item successfully, and upon viewing the song details, the image was shown in the window as expected.

All the system tests passed and the program performed up-to-standard during the tests.

Conclusion

Judging by the success of all the tests conducted in our testing plan, we can reasonably conclude that the product has been tested thoroughly enough and has held up to standard. In other words, this product is, from a quality-assurance standpoint, ready for shipping.