**Test Report**

**for**

**Gizmos: Gizzy Gallery**

**Authors:**

Zyad Ahmed,

Rostam Boroumand Rad,

Alina Momin,

Wes Robinson,

Thomas Wang,

Huzaifa Zia

UMBC

12/04/2022

Revisions

| Name | Revision |
| --- | --- |
| Alina Momin | 1.0 Document setup |
| Alina Momin | 1.1 Creating test scenarios |
| Alina Momin | 1.2 Descriptions written of how each scenario should be tested based on our SRS |
| Alina Momin | 1.3 Conducting testing against software, noting bugs |
| Alina Momin | 1.4 Retesting with updated software |

Table 1. Unit test of the login interface

| **TEST.ID** | **TEST SCENARIOS** | **DESCRIPTION** | **TEST STEP** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Valid login credentials | Test the login functionality of the web application to ensure that a registered user is allowed to login with username and password | 1.Launch the application  2.Navigate the login page  3.Enter valid username  4.Enter valid password  5.Click on login button | Login should be success | as expected | Pass |
| 2 | Create user account | Test the creation of a new account. | 1.Allows user to create a new user  2.confirmation is sent to their gmail | User should be able to create a new account | As expected | Pass |
| 3 | Sign out functionality | Check sign out functionality | 1. select sign out button | The user should be able to sign out. | As expected | Pass |
| 4 | Forgot Password | Test that user is sent information via email to recover their passwords | 1. Test that user receives recovery email  2. Test that user is able to re-enter a password  3. The user is able to access their account | User is able to recover their account | As expected | Pass |

Table 2. Unit Test of Home Page

| **TEST.ID** | **TEST SCENARIOS** | **DESCRIPTION** | **TEST STEP** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Gallery renders images onto the page. | Home gallery page will show the user’s Photos | 1. User is successfully logged in 2. The gallery on the home page renders the images. | Images are displayed in the gallery. | As expected | Pass |
| 2 | Sort By | Sort by has options such as date, color, and tag. The sort by options will accurately sort by the prompt selected. | 1. User selects to sort by an option such as date. 2. The files are rendered in sorted order. | Images are rerendered and organized based on the sort method selected. | As expected | Pass |
|  |  |  |  |  |  |  |
| 3 | Allows for images to be clicked | Images have pop up box with image information | 1. User selects the thumbnail of an image 2. User selected image maximizes 3. User metadata shows up in the left and right columns | Image is clicked, user can access image metadata and have image editing options. | As expected | Pass |

Table 3. Integration Testing for Firebase and React

| **TEST.ID** | **TEST SCENARIOS** | **DESCRIPTION** | **TEST STEP** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | User account credentials are stored in Firebase | Users create an account and their account information is stored in the Firebase database. | 1. Create a new account 2. User will be sent an email to verify their account | User is sent a verification email and they are able to successfully login with a verified account. | As expected | Pass |
| 2 | Metadata updates | User uploads or updates metadata. User is able to see the associated date and tags. | 1. User uploads a file 2. File contains metadata that is stored in Firebase 3. User is able to view the metadata and tags when accessing the images. | Tags are updated and stored in Firebase.  Tags are able to be retrieved with the images. | Metadata gets updated visually on the UI and in Firebase. Old data is still retrieved through the images. | Pass |

Table 4. Testing for security

| **TEST.ID** | **TEST SCENARIOS** | **DESCRIPTION** | **TEST STEP** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | User resets Password | When the user forgets the password they can reset it and the reset link is sent to their authenticated email. | 1. User clicks ‘Forgot Password’ 2. User gets sent reset password link 3. The user is able to set a new password. 4. Password is hashed | Users are sent an email to reset their password through their | As expected | Pass |
| 2 | Passwords are stored as hashes | When the user sets a password, the plain text will be hashed before storing in Firebase. | 1. User sets/resets a password. 2. Passwords are hashed and then stored in the database. | Passwords in the database are stored as hashes. | As expected | Pass |
| 3 | 2 Factor Authentication (2FA) | When user creates an account 2FA is used to verify their account | 1. User creates an account 2. Verification email sent to registrant email 3. Users will verify their account from the email prompt. | Once verified, the user has access to Gizzy Gallery account. | As expected | Pass |
| 4 | User sign out | Users are successfully able to log out of their account. | 1. User signs out 2. Access to the account and its data is terminated. | User is successfully logged out | As expected | Pass |
| 5 | User is unable to access their account after account is deleted | Users are able to delete account and when they try to sign in, they are sent an error toast. | 1. User tries to login using non existing account 2. Error message is shown at login 3. User is unable to access account | User does not have access to deleted account | As expected | Pass |
| 6 | Password has to be 6 characters | Users are prompted to add a stronger password if there the password is less than 6 characters. | 1. User attempts to create/reset a password 2. Password strength has to be 6 characters or more | User is prompted to enter a valid password | As expected | Pass |
| 7 | Uploads can only be .img or .png | Users are able to add the accepted file types: img and png. | 1. User will upload a file that is not acceptable file type 2. A error toast will appear “please upload img or png” | When user uploads a proper filetype, the upload shows up in the user gallery. | As expected | Pass |