Testing

# Local Unit Tests

These tests run on the local JVM and do not have access to functional Android framework APIs.

### InputValidatorTest

|  |  |
| --- | --- |
| isEmailValid\_Correct\_ReturnsTrue | OK |
| isEmailValid\_Incorrect\_ReturnsFalse | OK |
| isUsernameValid\_Correct\_ReturnsTrue | OK |
| isUsernameValid\_Incorrect\_ReturnsFalse | OK |
| isPasswordValid\_Correct\_ReturnsTrue | OK |
| isPasswordValid\_Incorrect\_ReturnsFalse | OK |
| isTitleValid\_Correct\_ReturnsTrue | OK |
| isTitleValid\_inCorrect\_ReturnsFalse | OK |
| isBodyValid\_Correct\_ReturnsTrue | OK |
| isBodyValid\_inCorrect\_ReturnsFalse | OK |

# Instrumented Unit Tests (Espresso)

These are all tests that must run on an Android hardware device or an Android emulator.

### LoginActivityTest

|  |  |
| --- | --- |
| email\_valid\_ReturnsTrue | OK |
| email\_notExists\_ReturnsFalse | OK |
| username\_valid\_ReturnsTrue | OK |
| username\_short\_ReturnsFalse | OK |
| username\_long\_ReturnsFalse | OK |
| username\_invalid\_ReturnsFalse | OK |
| username\_notExists\_ReturnsFalse | OK |

# Usability Testing

As the developers of our application, we have spent literally hundreds of hours trawling through our own code, yet ironically we can become blinded to to our applications most obvious pitfalls. To work around this, we decided to ask one of ours friends studying Engineering in DCU to test out our app and provide us with feedback.

After giving him a brief description of what the app does (so he had some context of what was going on), we gave him a list of objectives which he had to achieve, and then closely watched him as he attempted to complete each task. In line with our original project requirements, his objectives were:

1. Create an account
2. Sign in with the newly created account
3. Search for solutions to a particular exam paper
   * Enlarge any images the solution may have
   * Report the solution
   * Vote the solution up or down
4. Save the answer for offline use
5. View the saved answer (with the internet disabled)
6. Submit a new solution to a particular exam paper
7. Reply to any exam paper (and delete the reply again)
8. Change the password
9. Sign out

|  |  |  |
| --- | --- | --- |
| Objective | Problem(s) | Resolution(s) |
| 1 | 1. Didn’t see “create account button” 2. Was not clear why his password wasn’t accepted 3. Password requirements were far too strict | 1. Changed “create account” to “Don’t have an account?”, and increased font size 2. Implemented a clear message explaining the password requirements 3. Made password requirements less strict |
| 2 | N/A | N/A |
| 3 | N/A | N/A |
| 4 | N/A | N/A |
| 5 | N/A | N/A |
| 6 | Was not clear whether selected images had actually been selected | Implemented a clear message to confirm that the images had been selected |
| 7 | N/A |  |
| 8 | N/A |  |
| 9 | N/A |  |

# Assertion Testing

Our application is performing all of the project requirements outlined above.

# Backward Compatibility Testing

Our application was designed for Android 7.0 (Nougat, API 25) with compatibility for Android 4.4 and above. We tested our application on devices running Android 4.4 (KitKat, API 19), Android 6.0 (Marshmallow, API 23) and Android 7.0 (Nougat, API 25).