Crimson

Testing Document

Group 4: Ed Manolache, Parth Mody, Suraj Shetty, Vishal Doshi

Black Box Tests:

In black box testing, we check the functioning of each function module to check for correctness. The sequences of events that occur during the execution of the function are displayed in these cases.

Test Case 1:

Case name	Check-In
Test Procedure	The user selects the location to check in by tapping on the list item
Feature pass/fail criteria	The test passes if the user gets a toast saying he has successfully checked in and the test fails if he doesn't receive a toast
Event	GPS sensor tries to get current location from the Satellite
Means of control	The check-in takes place only after the user has logged in and is hence user specific.
Data	Location coordinates.
Result	Success

Test Case 2:

Case name	Check Artifacts
Test Procedure	The user selects to check artifacts at the placed he just checked-in by tapping on the ‘Check Artifacts’ button.
Feature pass/fail criteria	The test passes if the user gets a list of artifacts available at the place he/she checked in as well as the list of artifacts he/she owns. And the test fails if user is not displayed with either of the list
Event	The user clicks on the Check Artifacts button
Means of control	The check for artifacts takes place only after the user has checked-in to a location and is hence user and location specific.
Data	Location Artifacts, User Artifacts
Result	Success

Test Case 3:

Case name	Forage
Test Procedure	After selecting a level, the user clicks on the Forage button
Feature pass/fail criteria	The test passes if the user gets a toast saying “Foraging started” is displayed
Event	System updates the user resources and the location resources
Means of control	If the location resources have been exhausted a toast saying “No more resources to forage” is displayed
Data	stone, gold, lumber
Result	Success

Test Case 4:

Case name	Edit Profile
Test Procedure	Change the user account’s e-mail address, physical address, clan, and profile picture and confirm that the changes have been successfully saved.
Feature pass/fail criteria	If the user makes changes to their profile and they have been successfully saved then the feature passes. If the user is unable to make changes to their profile or when they make changes to their profile and the changes do not get saved the feature fails.
Event	The user clicks on the Edit Profile button, makes changes to his/her profile and then submits these changes.
Means of control	Edit one field at a time to ensure that each attribute has been successfully changed.
Data	User data: e-mail address, physical address, clan, and user profile picture.
Result	Success

Test Case 5:

Case name	Delete Account
Test Procedure	Create a new account and after logging in successfully, press the delete account button and confirm the deletion. After receiving a toast that the account has been successfully deleted attempt to log back in to the same account that was previously created.
Feature pass/fail Criteria	If the user is able to log in to an account which has been deleted the test fails, but if the user is unable to log into the account they just deleted the test passes.
Event	The user creates a new profile and logs into his/her profile and goes to the third tab and clicks edit profile then clicks delete account and confirms the deletion, then tries to log into his/her account again.
Means of control	The user will always create a new account and log into the account right after creation to make sure that the account exists before it is deleted.
Data	User database
Result	Success

White Box Tests:

The white box test cases are created to verify the outputs given by the code during execution. The actual values of particular variables in a normal case is considered here by comparing the expected variable values with the normal values

Test Case 1:

<table>
<tr><td>Case name</td><td>Fetch Current Location</td></tr>
<tr><td>Location</td><td>/Crimson/src/com/example/crimson/FragmentTab1.java</td></tr>
<tr><td>Method</td><td>public View onCreateView()</td></tr>
<tr><td>Event</td><td>GPS sensor tries to get current location from the Satellite</td></tr>
<tr><td>Variables</td><td>
<table>
<tr><td>Test No.</td><td>Name</td><td>Expected Value</td><td>Actual Value</td></tr>
<tr><td>1</td><td>double pLong</td><td>-87.6472</td><td>-87.6472</td></tr>
<tr><td></td><td>double pLat</td><td>41.8658</td><td>41.8658</td></tr>
<tr><td>2</td><td>double pLong</td><td>-86.3513</td><td>-86.3513</td></tr>
<tr><td></td><td>double pLat</td><td>39.5865</td><td>39.5865</td></tr>
</table>
</td></tr>
<tr><td>Result</td><td>Success</td></tr>
</table>

Test Case 2:

<table>
<tr><td>Case name</td><td>Display Artifact at a Place user is checked-in to.</td></tr>
<tr><td>Location</td><td>/Crimson/src/com/example/crimson/CheckArtifacts.java</td></tr>
<tr><td>Method</td><td>public void loadPlaceArtifacts()</td></tr>
<tr><td>Event</td><td>Program tries to fetch list of artifacts (by artifactID) at current location</td></tr>
<tr><td>Variables</td><td>
<table>
<tr><td>Test No.</td><td>Name</td><td>Expected Value</td><td>Actual Value</td></tr>
<tr><td>1</td><td>List<Integer> aa</td><td>[3,6,7,10]</td><td>[3,6,7,10]</td></tr>
<tr><td>2</td><td>List<Integer> aa</td><td>[4,5,8,9]</td><td>[4,5,8,9]</td></tr>
</table>
</td></tr>
<tr><td>Result</td><td>Success</td></tr>
</table>

Test Case 3:

Case name	Display artifacts owned by user
Location	/Crimson/src/com/example/crimson/CheckArtifacts.java
Method	loadUserArtifacts()
Event	Program tries to fetch list of artifacts (by artifactID) owned by current user
Variables	

Test No.	Name	Expected Value	Actual Value
1	List<Integer> al	[2,4,5]	[2,4,5]
2	List<Integer> al	[1,3,6,9]	[1,3,6,9]

Result	Success

Test Case 4:

Case name	MissingToast
Location	/Crimson/src/com/example/crimson/BattleRecieverService.java
Method	toastResult()
Event	After a user-user or user-AI battle.
Variables	

Test No.	Name	Expected Value	Actual Value
1	result	win	win
2	result	lose	null
3	result	tie	tie
4	result	lose	lose

Result	Test fails occasionally due to failure of back end update

Test Case 5:

Case name	AI Attack
Location	/Crimson/src/com/example/crimson/AIAttackService.java /Crimson/src/com/example/crimson/AIDialogActivity.java
Method	AIDialogActivity OnClick()
Event	The result of a battle with an AI agent with the user having default attributes of 500 attack rating and 500 defense rating.
Variables	

Test No.	AI Attributes	Expected Value	Actual Value
1	Anaconda (A: 300 D: 400)	Win	Win
2	Black Widow (A: 100 D: 500)	Win	Win
3	Tiger (A: 600 D: 300)	Win	Win
4	Bear (A: 500 D: 700)	Loss	Loss
5	Vulture (A: 200 D: 500)	Win	Win
6	Lion (A: 600 D: 400)	Tie	Tie

Result	Success

Test Case 6:

Case name	MultipleBattles
Location	/Crimson/src/com/example/crimson/BattleRecieverService.java
Method	Multiple methods
Event	When multiple people attack the same user simultaneously.
Variables	

Test No.	Name	Expected Value	Actual Value
1	result	win	Wrong/Invalid Value
2	result	lose	Wrong/Invalid Value
3	result	tie	Wrong/Invalid Value

Result	Test fails.

Test Case 7:

<table>
<tr><td>Case name</td><td>ForageCompletion</td></tr>
<tr><td>Location</td><td>/Crimson/src/com/example/crimson/Forage.java</td></tr>
<tr><td>Method</td><td>Multiple methods</td></tr>
<tr><td>Event</td><td>When location resources are exhausted</td></tr>
<tr><td>Variables</td><td>
<table>
<tr><td>Test no.</td><td>Name</td><td>Expected Value</td><td>Actual Value</td></tr>
<tr><td>1</td><td>isforagingcompleted</td><td>true</td><td>true</td></tr>
<tr><td>2</td><td>isforagingcompleted</td><td>false</td><td>false</td></tr>
</table>
</td></tr>
<tr><td>Result</td><td>Test fails occasionally due to lack of connection with back-end server</td></tr>
</table>

Code Inspection

Every module of the code was tested by the member who created it and inspected by the other three members. This was done to create an unbiased and impartial perspective on every test case and its result

Inspection 1:

Component:	Battle System
Location:	Crimson/src/com/example/crimson/BattleRecieverService.java Crimson/src/com/example/crimson/BattleDialogActivity.java Crimson/src/com/example/crimson/BattleChallengerService.java Crimson/src/com/example/crimson/FragmentTab1.java
Inspector Comments:	Inspector 1: Too many frames skipped...performance lag,use background threads for performance enhancements. Inspector 2: Battle System does not support concurrent battles. Inspector 3: Code needs more organization.

Inspection 2:

Component:	AI Attacks
Location:	/Crimson/src/com/example/crimson/AIAttackService.java /Crimson/src/com/example/crimson/AIDialogActivity.java
Inspector Comments:	Inspector 1: Random number generator code should be placed into a method to easily be re-used rather than implementing it twice. Inspector 2: The AI attack dialog box should display both the AI's attack and defense attributes as well as the players so the player would get a better idea of when they should attack or evade. Inspector 3: Try to avoid using public variables.

Inspection 3:

Component:	Edit Profile
Location:	/Crimson/src/com/example/crimson/EditProfile.java
Inspector Comments:	Inspector 1: In the switch case the default case should not be left blank. Inspector 2: Public variables can be condensed down or avoided altogether. Inspector 3: Use a switch case with more than one case in order to catch all the various attributes when selecting a new profile picture.

Inspection 4:

Component:	Artifacts
Location:	/Crimson/src/com/example/crimson/CheckArtifacts.java /Crimson/src/com/example/crimson/FragmentTab2.java
Inspector Comments:	Inspector 1: Save the artifacts in an order sorted by the artifactIDs. Inspector 2: Many frames skipped causing lag,use background threads for performance enhancements. Inspector 3: Store “artifact-icons” locally, would increase the fetch speed.

Inspection 5:

Component:	Forage
Location:	/Crimson/src/com/example/crimson/Forage.java
Inspector Comments:	Inspector 1: Display resources changing dynamically on screen so that user would know the amount left Inspector 2: Increase the timer duration so that resources would not be depleted quickly and other players can forage together for longer Inspector 3: Create a separate function for checking the resource exhaustion that can be called from the timer class instance.

------X------