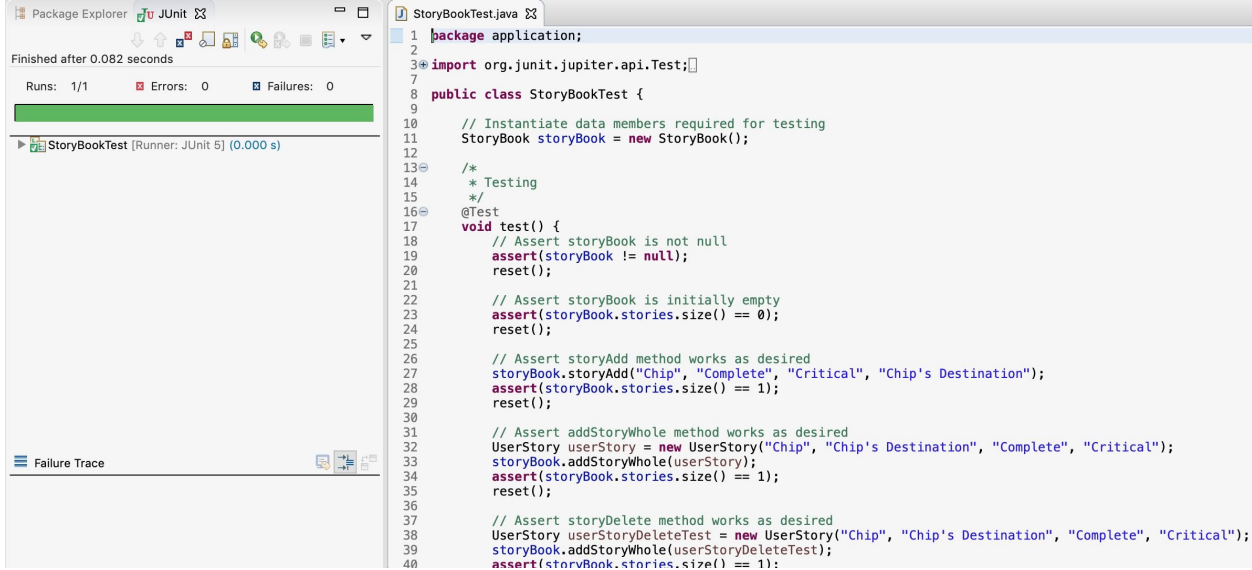


Acceptance Tests

Preconditions	Steps	Postconditions	Results
- 3 stories are created	<ul style="list-style-type: none">- Stories are initialized with random inputs (header, content, priority, client)- Each story holds unique ID- Client(s) update	- All stories are present on Server/other clients	Works as expected
- Stories Modified/Deleted	<ul style="list-style-type: none">- Stories modified by different client then who created story- Each story stays robust, only changes when one client updates at a time- Deleted stories are updated across Clients/Server	- All stories appropriately reflected on Server/other clients	Story Creation/Deletion and Moving around the Stages was able to be reflected in the server. Description/Priority Changes had trouble being pushed. Were not able to have network support automatic updates
- Burndown Chart initialized	<ul style="list-style-type: none">- Stories are initialized randomly- Server updates and holds all stories	- All stories are appropriately reflected on the BurnDown chart	Chart worked as expected, due to issues with pushing automatic updates, we were not able to fully implement the chart

Integration Tests (Unit Testing)

Screenshot of Results/Code



The screenshot displays an IDE interface with two main panels. The left panel shows the JUnit test results for 'StoryBookTest'. It indicates that the test was 'Finished after 0.082 seconds', with 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. A green progress bar is visible, and the test is listed as 'StoryBookTest [Runner: JUnit 5] (0.000 s)'. The right panel shows the source code for 'StoryBookTest.java'. The code is as follows:

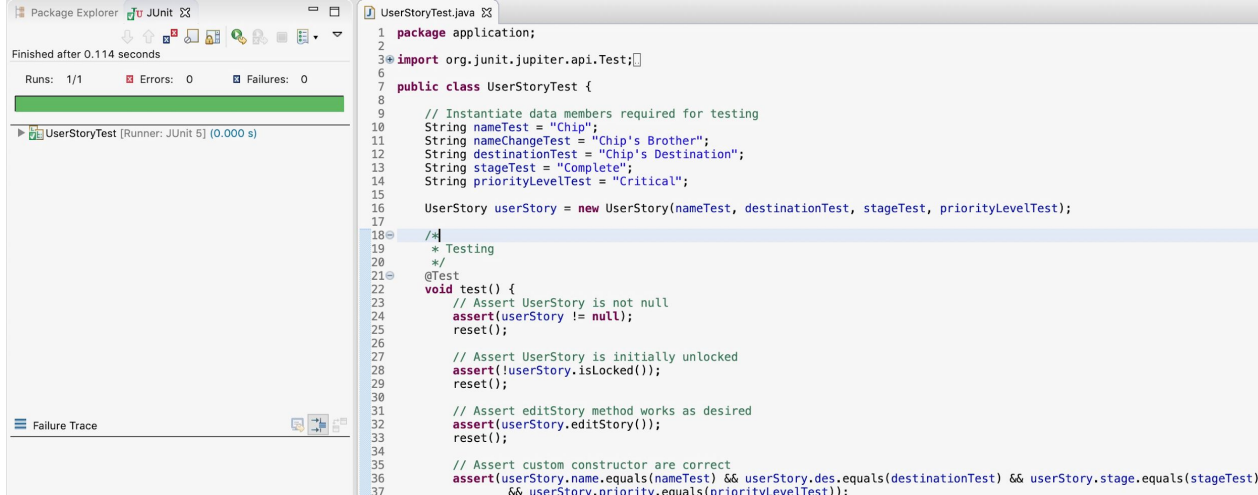
```
1 package application;
2
3 import org.junit.jupiter.api.Test;
4
5 public class StoryBookTest {
6
7     // Instantiate data members required for testing
8     StoryBook storyBook = new StoryBook();
9
10    /*
11     * Testing
12     */
13    @Test
14    void test() {
15        // Assert storyBook is not null
16        assert(storyBook != null);
17        reset();
18
19        // Assert storyBook is initially empty
20        assert(storyBook.stories.size() == 0);
21        reset();
22
23        // Assert storyAdd method works as desired
24        storyBook.storyAdd("Chip", "Complete", "Critical", "Chip's Destination");
25        assert(storyBook.stories.size() == 1);
26        reset();
27
28        // Assert addStoryWhole method works as desired
29        UserStory userStory = new UserStory("Chip", "Chip's Destination", "Complete", "Critical");
30        storyBook.addStoryWhole(userStory);
31        assert(storyBook.stories.size() == 1);
32        reset();
33
34        // Assert storyDelete method works as desired
35        UserStory userStoryDeleteTest = new UserStory("Chip", "Chip's Destination", "Complete", "Critical");
36        storyBook.addStoryWhole(userStoryDeleteTest);
37        assert(storyBook.stories.size() == 1);
38    }
39 }
```

Test # - Justification

1 - Test StoryBook and functionality

Integration Tests (Unit Testing)

Screenshot of Results/Code



The screenshot displays an IDE interface with two main panels. The left panel shows the 'Package Explorer' and 'JUnit' test results. The 'JUnit' section indicates 'Finished after 0.114 seconds' and 'Runs: 1/1', 'Errors: 0', 'Failures: 0'. Below this, a green progress bar is shown. The right panel displays the source code for 'UserStoryTest.java'. The code is as follows:

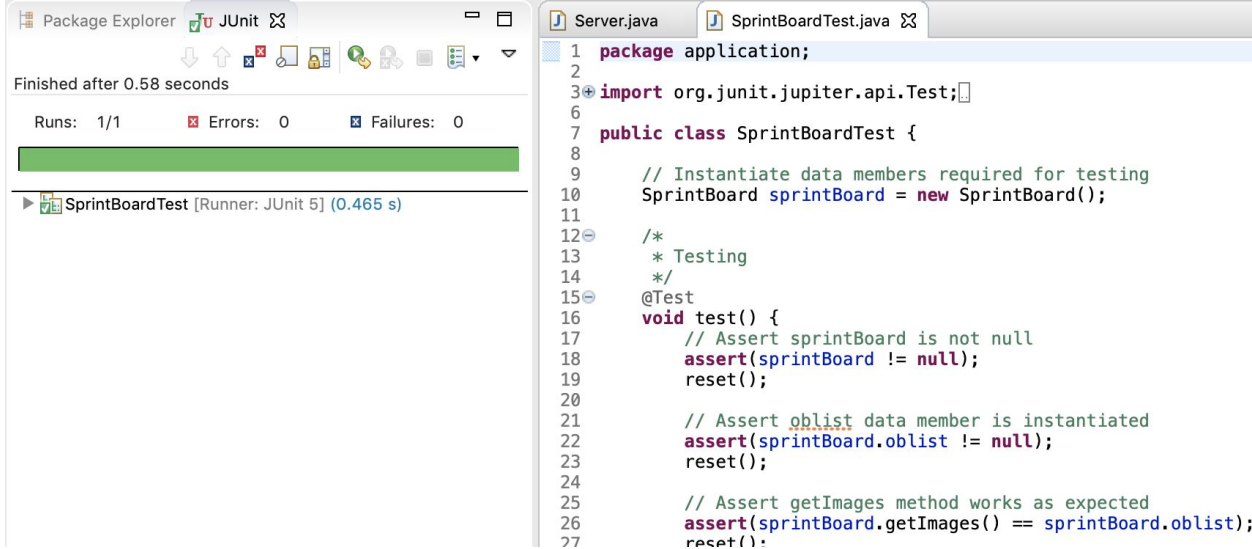
```
1 package application;
2
3 import org.junit.jupiter.api.Test;
4
5 public class UserStoryTest {
6
7     // Instantiate data members required for testing
8     String nameTest = "Chip";
9     String nameChangeTest = "Chip's Brother";
10    String destinationTest = "Chip's Destination";
11    String stageTest = "Complete";
12    String priorityLevelTest = "Critical";
13
14    UserStory userStory = new UserStory(nameTest, destinationTest, stageTest, priorityLevelTest);
15
16    /*
17     * Testing
18     */
19    @Test
20    void test() {
21        // Assert UserStory is not null
22        assert(userStory != null);
23        reset();
24
25        // Assert UserStory is initially unlocked
26        assert(!userStory.isLocked());
27        reset();
28
29        // Assert editStory method works as desired
30        assert(userStory.editStory());
31        reset();
32
33        // Assert custom constructor are correct
34        assert(userStory.name.equals(nameTest) && userStory.des.equals(destinationTest) && userStory.stage.equals(stageTest)
35            && userStory.priority.equals(priorityLevelTest));
36    }
37 }
```

Test # - Justification

2 - Test the functionality of the lock and function of UserStory

Integration Tests (Unit Testing)

Screenshot of Results/Code



The screenshot displays an IDE interface. On the left, the 'JUnit' tab shows test results: 'Finished after 0.58 seconds', 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. A green progress bar is visible. Below this, the test runner 'JUnit 5' is shown with a duration of '0.465 s'. On the right, the 'SprintBoardTest.java' file is open, showing the following code:

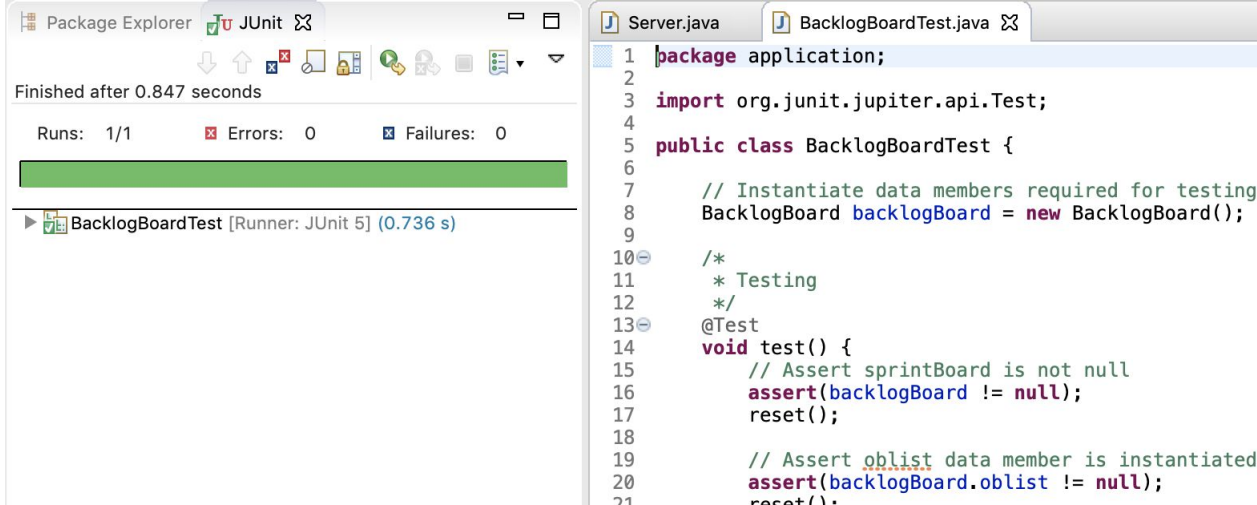
```
1 package application;
2
3 import org.junit.jupiter.api.Test;
4
5
6 public class SprintBoardTest {
7
8     // Instantiate data members required for testing
9     SprintBoard sprintBoard = new SprintBoard();
10
11     /*
12      * Testing
13      */
14     @Test
15     void test() {
16         // Assert sprintBoard is not null
17         assert(sprintBoard != null);
18         reset();
19
20         // Assert oblist data member is instantiated
21         assert(sprintBoard.oblist != null);
22         reset();
23
24         // Assert getImages method works as expected
25         assert(sprintBoard.getImages() == sprintBoard.oblist);
26         reset();
27     }
28 }
```

Test # - Justification

3 - Test SprintBoard and functionality

Integration Tests (Unit Testing)

Screenshot of Results/Code

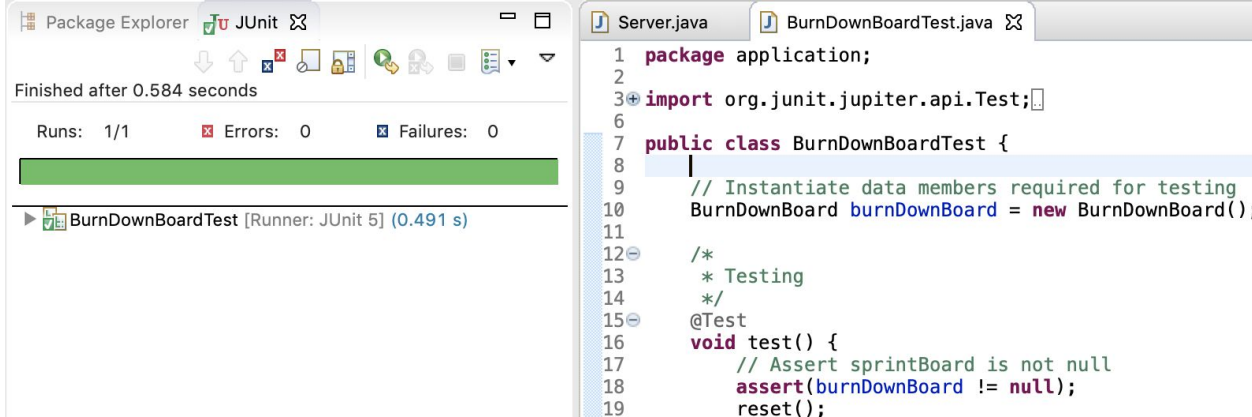


Test # - Justification

4 - Test BacklogBoard and functionality

Integration Tests (Unit Testing)

Screenshot of Results/Code



Test # - Justification

5 - Test BurnDownBoard and functionality