Nicolas Meneses

Description of software project under test

- The web application under test is a simple Typing Speed Test application with two main modes: Mode A and Mode B. In Mode A, users are asked to type the largest amount of words they can under 60 seconds then, after the 60 seconds, the following results are displayed: ‘accerts’ (number of words that were written without typos), ‘errors’ (number of words that were written incorrectly), and “words per minute” (words written under 60 seconds, without considering typos). On the other hand, in Mode B, users are asked to type the alphabet as fast as they can. While the user is typing the alphabet, the number of incorrect characters, the time, and the accuracy percentage are displayed. After the user finishes typing the alphabet without any mistakes, the following results are displayed: “words per minute”, “characters per minute”, and “time taken to write the alphabet”.
- In total, the web application has 3 URLs: One landing page (see Figure 1: Landing Page), Mode A page (see Figure 2: Mode A), and Mode B page (see Figure 3: Mode B).

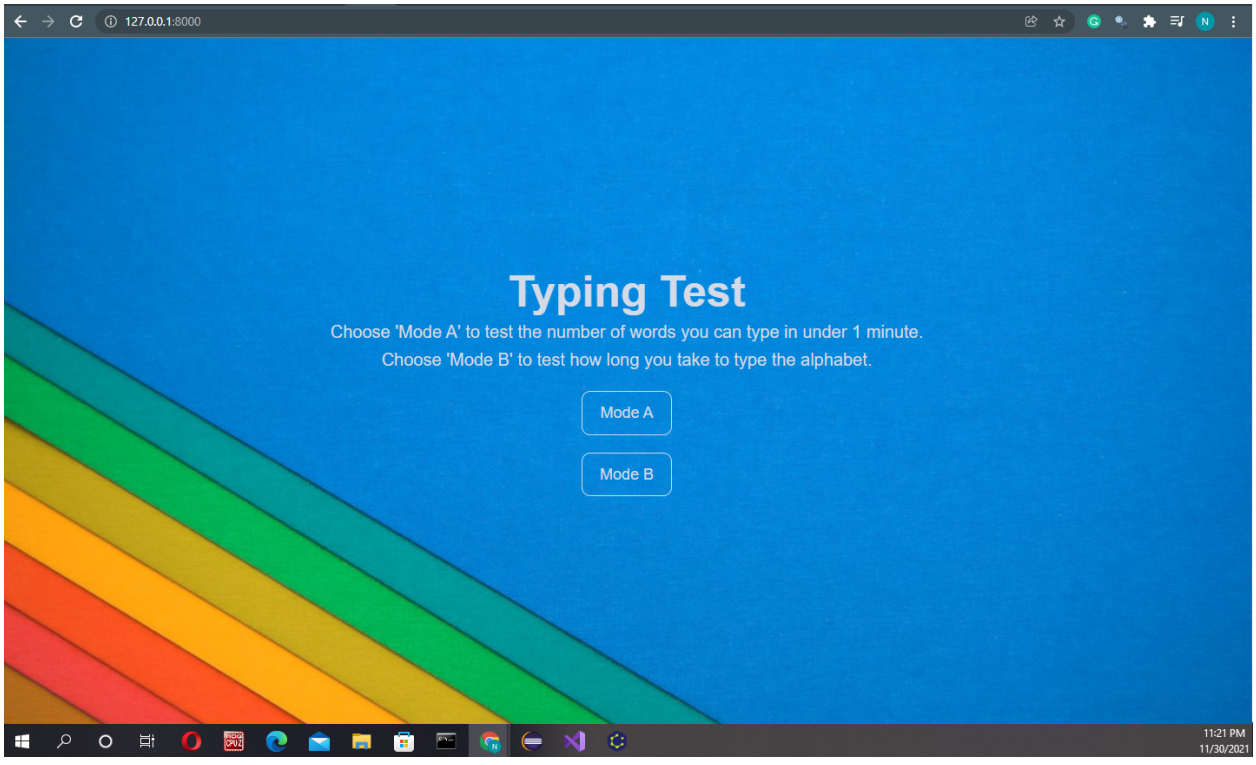


Figure 1: Landing Page

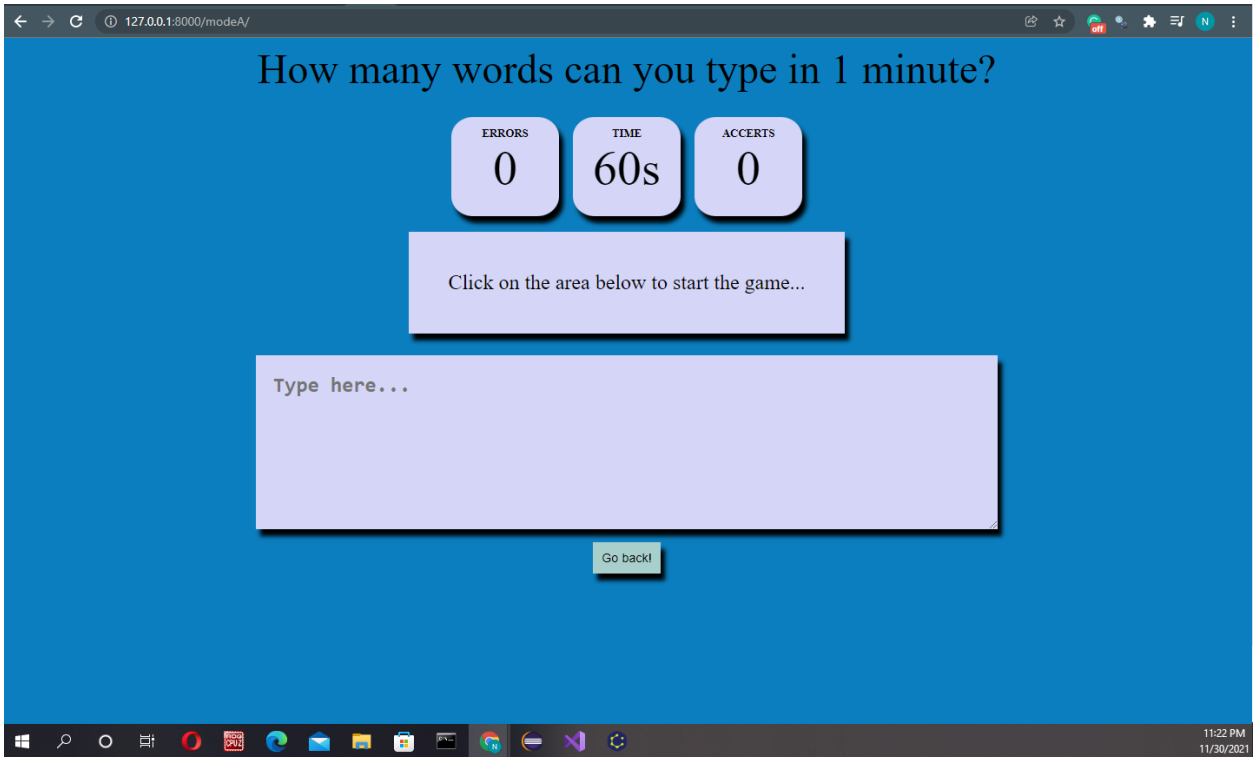


Figure 2: Mode A

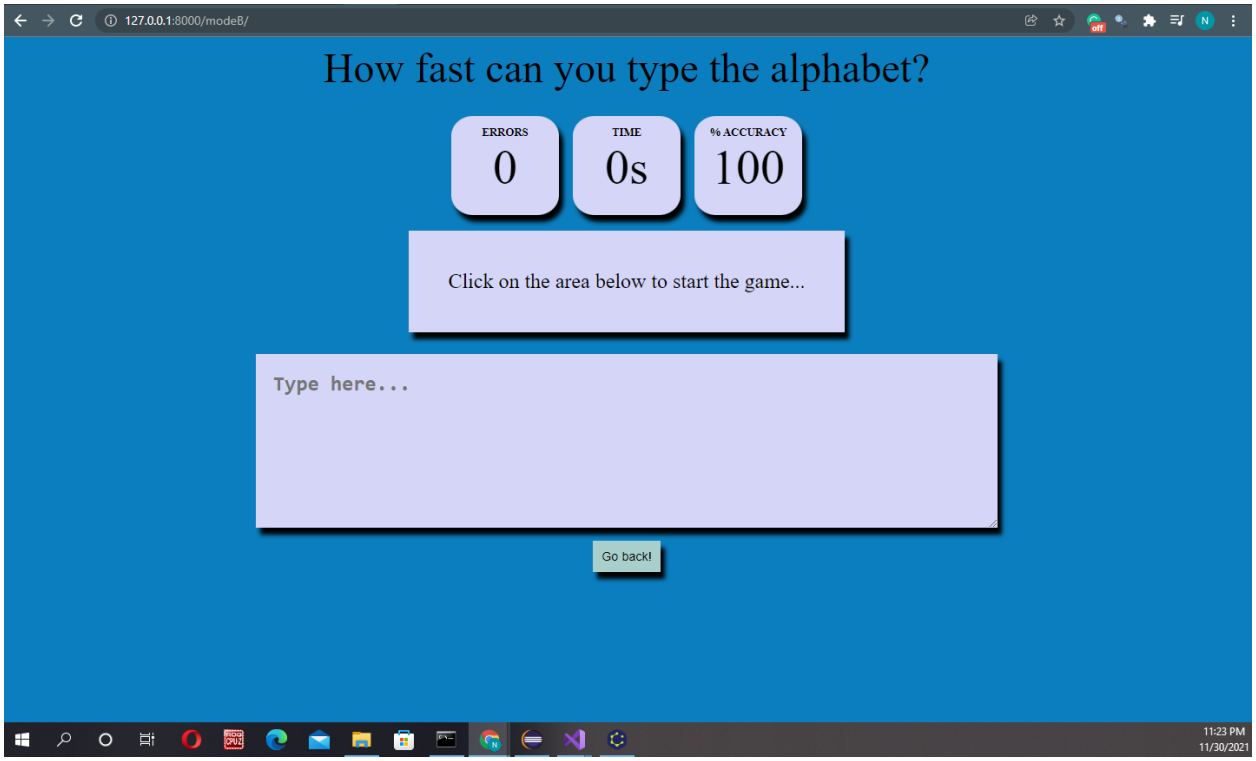


Figure 3: Mode B

Nicolas Meneses

Authorship:

- To build this web application, I followed the Django youtube tutorial: https://www.youtube.com/watch?v=_mrj6OW9DA8 by Sachin Pawar. I made several modifications and adaptations to build Mode A and Mode B for my web application. Additionally, I used the following HTML+CSS template by Traversy Media (https://www.youtube.com/watch?v=hVdTQWASliE) for my landing page: https://codepen.io/bradtraversy/pen/XerXYV.

Number of Components:

- Default files from Django Framework:
  - SQLITE3 Files: 3
  - Python Files: 14
  - .pyc Files: 20
- Files I created:
  - Javascript Files: 2
  - HTML Files: 3

Lines of Code (blank lines and comments included)

- gameModeA.js: 218 lines of code
- gameModeB.js: 160 lines of code
- home.html: 23 lines of code
- modeA.html: 40 lines of code
- modeB.html: 44 lines of code

Testing Environment Documentation:

- The testing environment was set up locally using Django's local web server "The development server" (https://vegibit.com/how-to-run-django-development-server). Hence, for testing purposes the application was locally hosted at port 8000 (i.e. "http://127.0.0.1:8000/"). Furthermore, the framework Selenium version 3.141.59 and junit 4.12.0.v201504281640 were used to automate tests. Additionally, the operating system where the tests were executed was Windows 10 version 10.0.19042, the version of python used was 3.9.1, and the version of Django was 3.1.7.

Coverage Criteria and Design Tests:

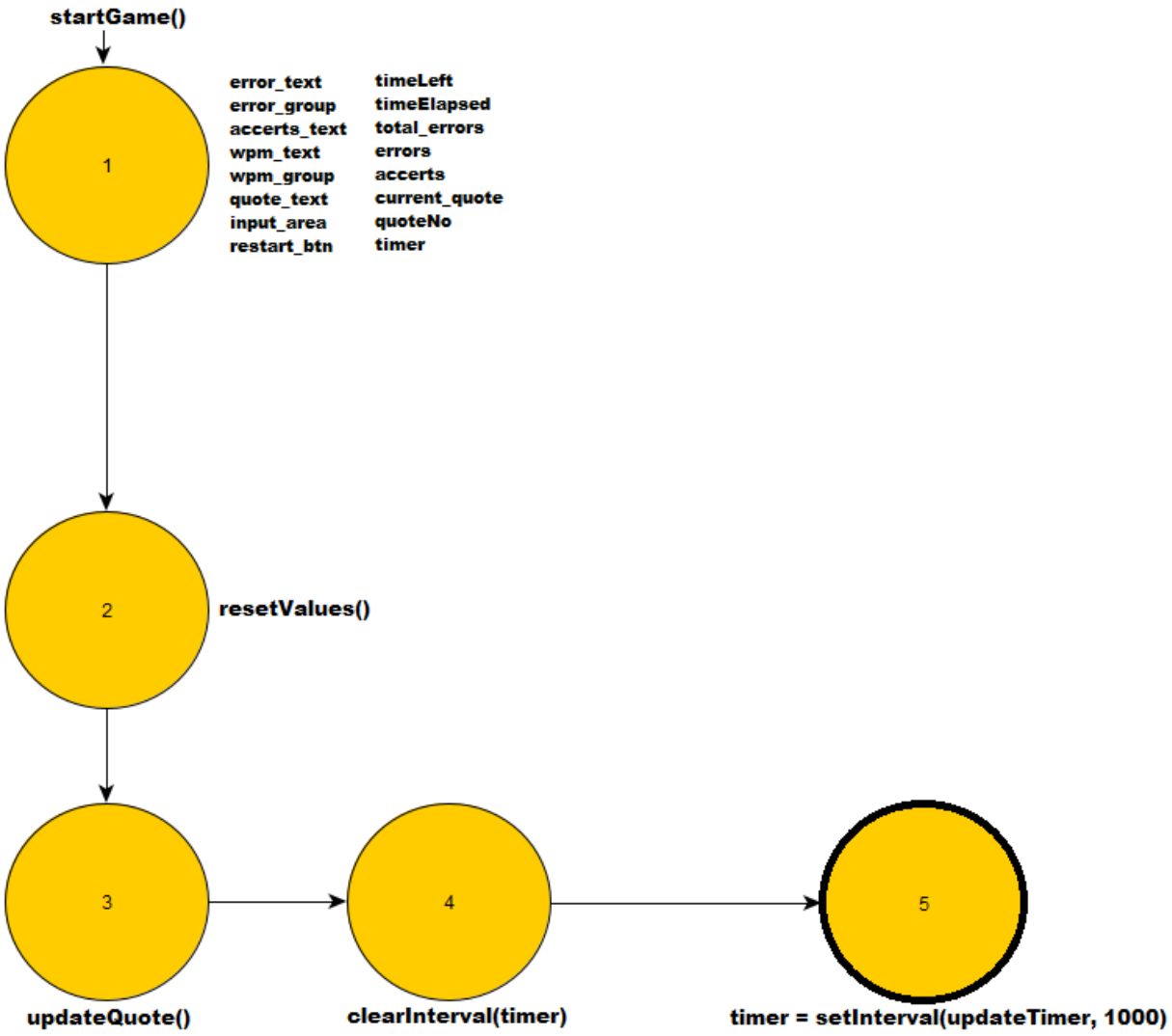
- The coverage criteria used for test designs is Edge-Coverage criteria. Furthemore, I decided to do Unit Level Testing for each method. Since the application is divided into two main modes, I created tests for each Mode in different java files. I decided to do this because, even though both Modes use different javascript files, all of the functions in

both javascript files are named equally and use almost the same variables, however most functions are not the same despite having the same name.

- The following section contains (1) Graphs for all methods for both Mode A, and Mode B. (2) Test Requirements for each graph (for unit level testing). (3) Test paths for each graph. (4) Input for tests, expected output, and assumptions or pre-states. (5) A reference from the path under test to the actual test either in AModeTest.java or BModeTest.java.

Nicolas Meneses

Test Designs for “Mode A” of application



Note:

The setInterval() method calls a function at specified intervals (in milliseconds). In this case, the function ‘updateTimer’ is being called every second.

The setInterval() method will continue calling the function until clearInterval() is called, or the window is closed.

The ID value returned by setInterval() is used as the parameter for the clearInterval() method. In this case, it is stored in the variable “timer”.

Nicolas Meneses

TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - startGame()
    - TR: {(1,2), (2,3), (3,4), (4,5)}

Test Paths

- Unit Level Testing - startGame()
    - P1:[1,2,3,4,5]

Test Cases Design

- Unit Level Testing - startGame() - Path 1 - **Test #1 in AModeTest.java**
    - Test values/inputs: The method startGame() is called after the user clicks on the input text area. Hence, the input for this test would be clicking on the text area.
    - Expected Output: The text “Click on the area below to start the game…” should disappear and the first word from the word bank should appear.

Nicolas Meneses

TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - resetValues()
  - TR: {(1)}

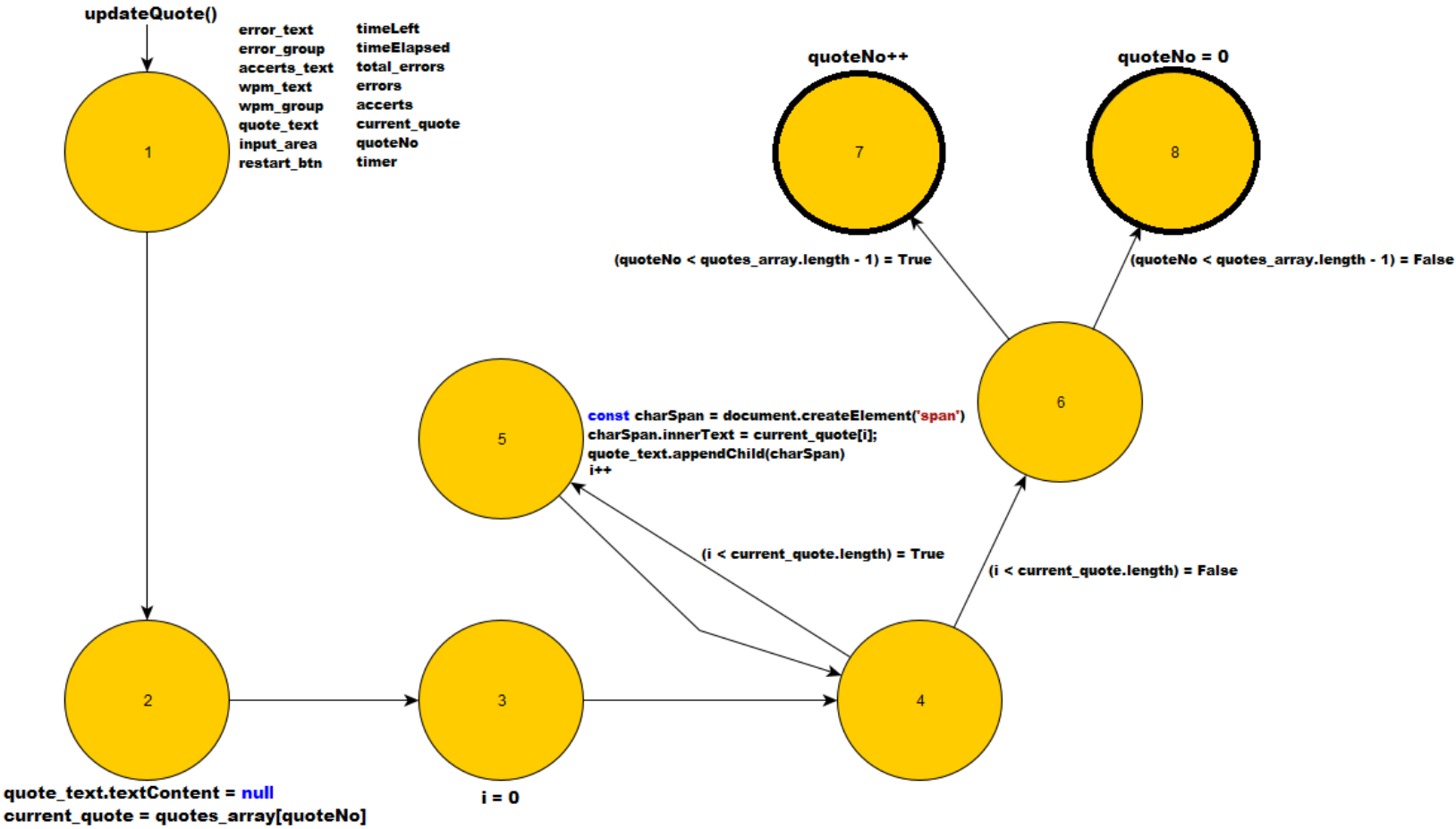
Test Paths

- Unit Level Testing - resetValues()
  - P1:[1]

Test Cases Design

- Unit Level Testing - resetValues() - Path 1 - **Test #2 in AModeTest.java**
  - Test values/inputs: The method resetValues() is called by the startGame() method after the user clicks on the input text area. The input for this test would be clicking on the text area, waiting a second, clicking outside the text area and then clicking in the text area again.
  - Expected Output: The first click should call startGame() and the timer should start, clicking outside the textarea and then clicking in the text area again should restart the game; the method resetValues() should reset the timer to 60 seconds.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - updateQuote()
  - TR: {(1,2), (2,3), (3,4), (4,5), (5,4), (4,6), (6,7), (6,8)}

Test Paths

- Unit Level Testing - updateQuote()
  - P1:[1,2,3,4,5,4,6,7]
  - P2:[1,2,3,4,5,4,6,8]

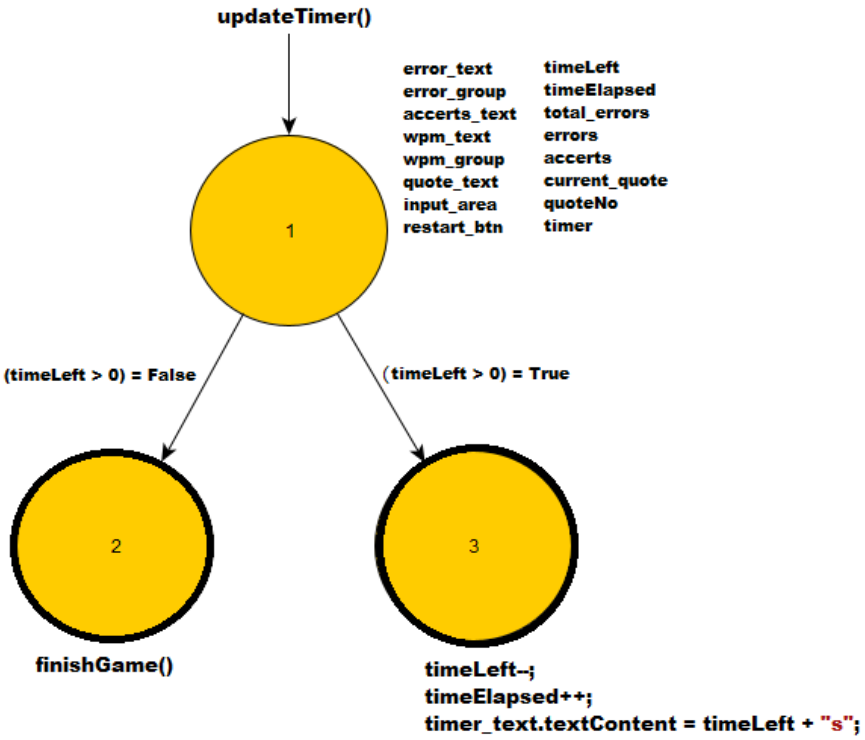
Test Cases Design

- Unit Level Testing - updateQuote() - Path 1 - **Test #3 in AModeTest.java**
  - Test values/inputs: The method updateQuote() is called by the processCurrentText() method after the user has input enough characters to match the characters of the current word. The input for this test would be clicking on the text area and typing “zebra” (the first word on the word-bank), so that the updateQuote() method is called by the processCurrentText() method.
  - Expected Output: The symbol “$” should appear on screen as the second word in the word-bank is “!@#$”.

Nicolas Meneses

- Unit Level Testing - updateQuote() - Path 2 - **Test #4 in AModeTest.java**
    - Test values/inputs: The method updateQuote() is called by the processCurrentText() method after the user has input enough characters to match the characters of the current word. The input for this test would be clicking on the text area and typing all the 50 words in the word-bank, so that the updateQuote() method is called by the processCurrentText() method after the word-bank has reached its end (i.e. so that the path goes from node 6 to node 8).
    - Expected Output: The letter “z” should appear on screen as the first word in the word-bank is “zebra”.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - updateTimer()
  - TR: {(1,2), (1,3)}

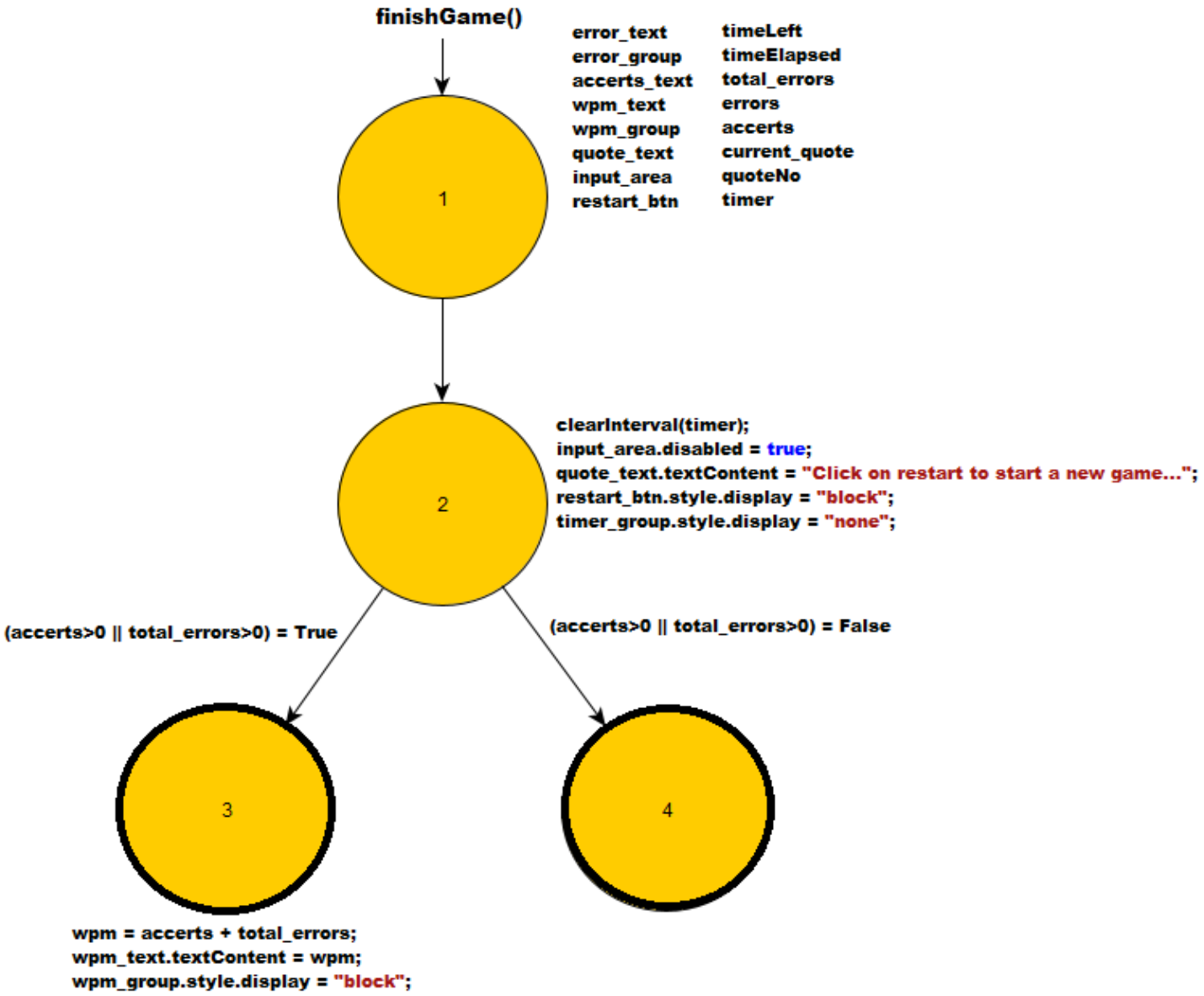
Test Paths

- Unit Level Testing - updateTimer()
  - P1:[1,3]
  - P2:[1,2]

Test Cases Design

- Unit Level Testing - updateTimer() - Path 1 - **Test #5 in AModeTest.java**
  - Test values/inputs: The method updateTimer() is called every second after the startGame() method is invoked. The input for this test would be clicking on the text area and waiting 3 seconds.
  - Expected Output: The timer should output “57s”.
- Unit Level Testing - updateTimer() - Path 2 - **Test #6 in AModeTest.java**
  - Test values/inputs: The method updateTimer() is called every second after the startGame() method is invoked. The input for this test would be clicking on the text area and waiting 60 seconds so that the method finishGame() (node 2) is called.
  - Expected Output: The “Restart” button should appear on screen.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - finishGame()
  - TR: {(1,2), (2,3), (2,4)}

Test Paths

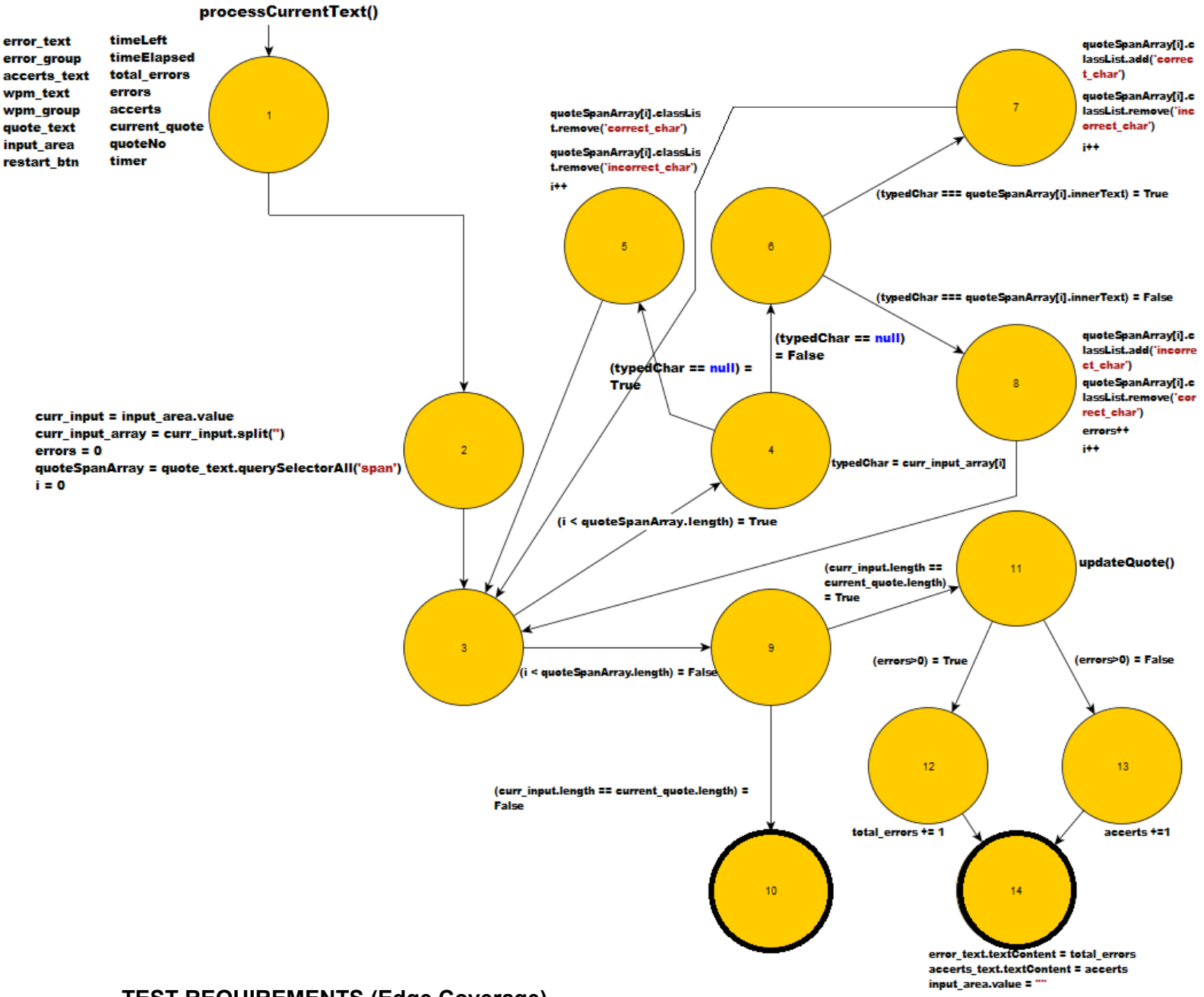
- Unit Level Testing - finishGame()
  - P1:[1,2,3]
  - P2:[1,2,4]

Nicolas Meneses

Test Cases Design

- Unit Level Testing - finishGame() - Path 1 - **Test #7 in AModeTest.java**
    - Test values/inputs: The method finishGame() is called by the updateTimer() method when the time runs out (60 seconds). The input for this test would be clicking on the text area, typing “aaaaaa”, and waiting 61 seconds.
    - Expected Output: “WPM” should appear on screen.
- Unit Level Testing - finishGame() - Path 2 - **Test #8 in AModeTest.java**
    - Test values/inputs: The method finishGame() is called by the updateTimer() method when the time runs out (60 seconds). The input for this test would be clicking on the text area and waiting 61 seconds.
    - Expected Output: “Click on restart to start a new game...” should appear on screen.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - processCurrentText()
  - TR: {(1,2), (2,3), (3,4), (4,5), (5,3), (4,6), (6,7), (7,3), (6,8), (8,3), (3,9), (9,10), (9,11), (11,12), (11,13), (12,14), (13,14)}

Test Paths

- Unit Level Testing - processCurrentText()
  - P1:[1,2,3,4,5,3,4,6,7,3,4,6,8,3,9,10]
  - P2:[1,2,3,9,11,12,14]
  - P3:[1,2,3,9,11,13,14]

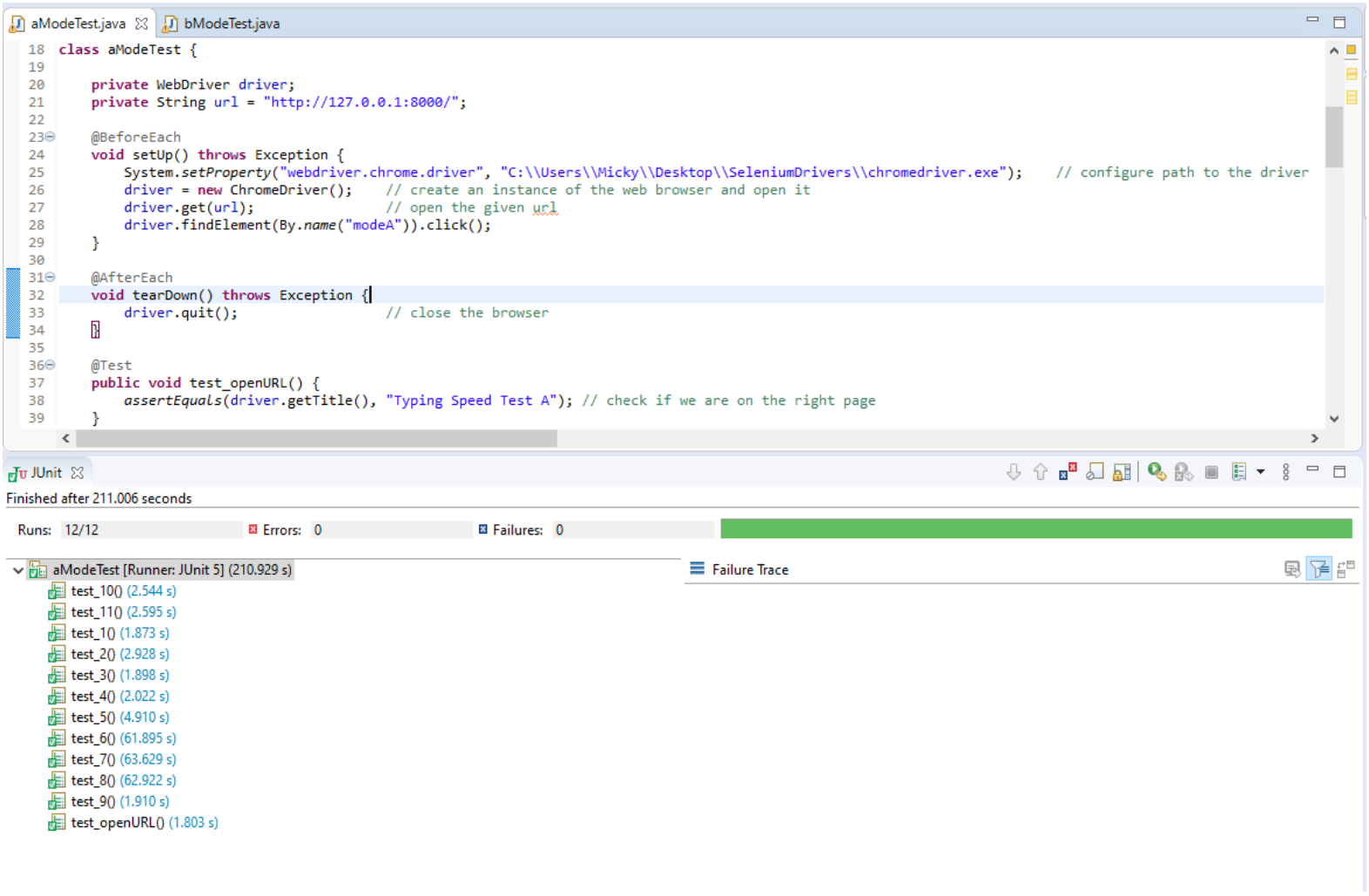
Nicolas Meneses

Test Cases Design

- Unit Level Testing - processCurrentText() - Path 1 - **Test #9 in AModeTest.java**
    - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be clicking on the text area and typing “za”.
    - Expected Output: The letter “z” should be green (.correct_char attribute), the letter “e” should be red (.incorrect_char attribute).
- Unit Level Testing - processCurrentText() - Path 2 - **Test #10 in AModeTest.java**
    - Previous Set-up: Click on the text area and type “zebra”, “!@#$”, “dress”, and “mal”
    - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be typing “e” in the text area.
    - Expected Output: The number of accerts should be 4.
- Unit Level Testing - processCurrentText() - Path 3 - **Test #11 in AModeTest.java**
    - Previous Set-up: Click on the text area and type “zebra”, “!@#$”, “dress”, and “mal”
    - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be typing “+” in the text area.
    - Expected Output: The number of accerts should be 3.

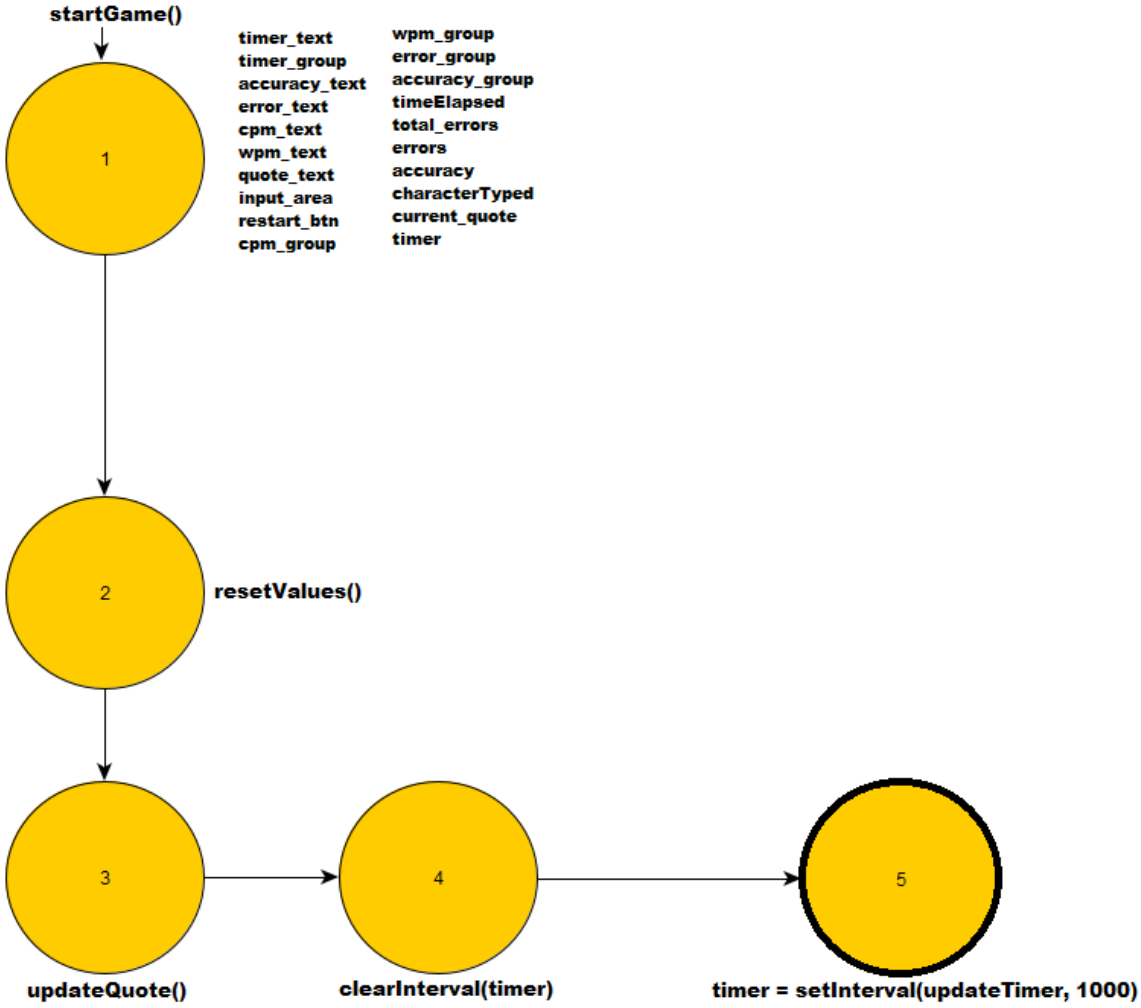
Nicolas Meneses

Results:



Nicolas Meneses

Test Designs for “Mode B” of application.



Note:

The setInterval() method calls a function at specified intervals (in milliseconds). In this case, the function ‘updateTimer’ is being called every second.

The setInterval() method will continue calling the function until clearInterval() is called, or the window is closed.

The ID value returned by setInterval() is used as the parameter for the clearInterval() method. In this case, it is stored in the variable “timer”.

Nicolas Meneses

TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - startGame()
    - TR: {(1,2), (2,3), (3,4), (4,5)}

Test Paths

- Unit Level Testing - startGame()
    - P1:[1,2,3,4,5]

Test Cases Design

- Unit Level Testing - startGame() - Path 1 - **Test #1 in bModeTest.java**
    - Test values/inputs: The method startGame() is called after the user clicks on the input text area. Hence, the input for this test would be clicking on the text area.
    - Expected Output: The text “Click on the area below to start the game…” should disappear and “abcdefghijklmnopqrstuvwxyz” should appear.

Nicolas Meneses

TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - resetValues()
  - TR: {(1)}

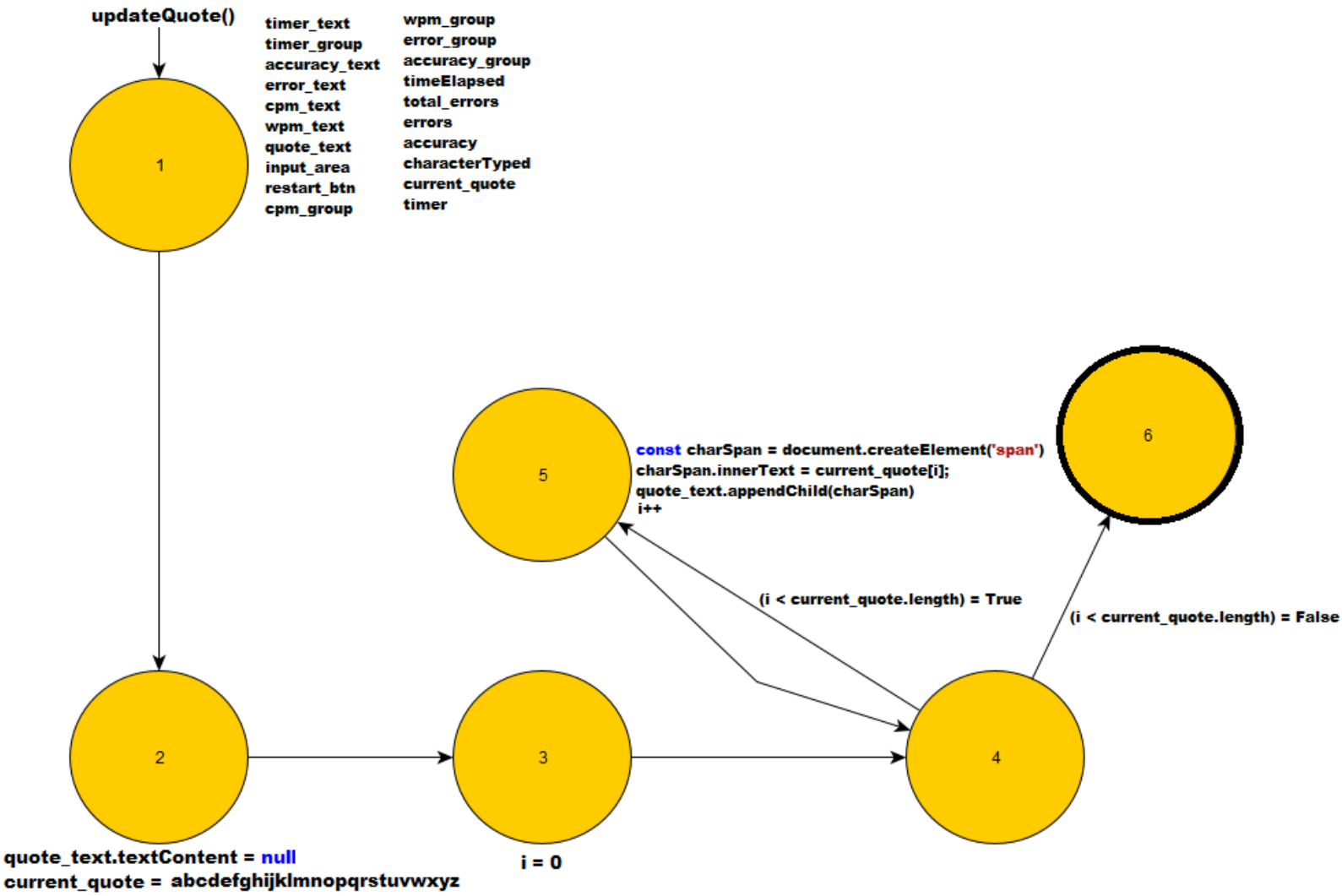
Test Paths

- Unit Level Testing - resetValues()
  - P1:[1]

Test Cases Design

- Unit Level Testing - resetValues() - **Test #2 in bModeTest.java**
  - Test values/inputs: The method resetValues() is called after clicking on the restart button. The input for this test would be clicking on the restart button.
  - Expected Output: The timer should reset to 0 seconds.
  - Assumptions: The user has finished a game, as the “restart button” only appears after a game has been finalized.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - updateQuote()
  - TR: {(1,2), (2,3), (3,4), (4,5), (5,4), (4,6)}

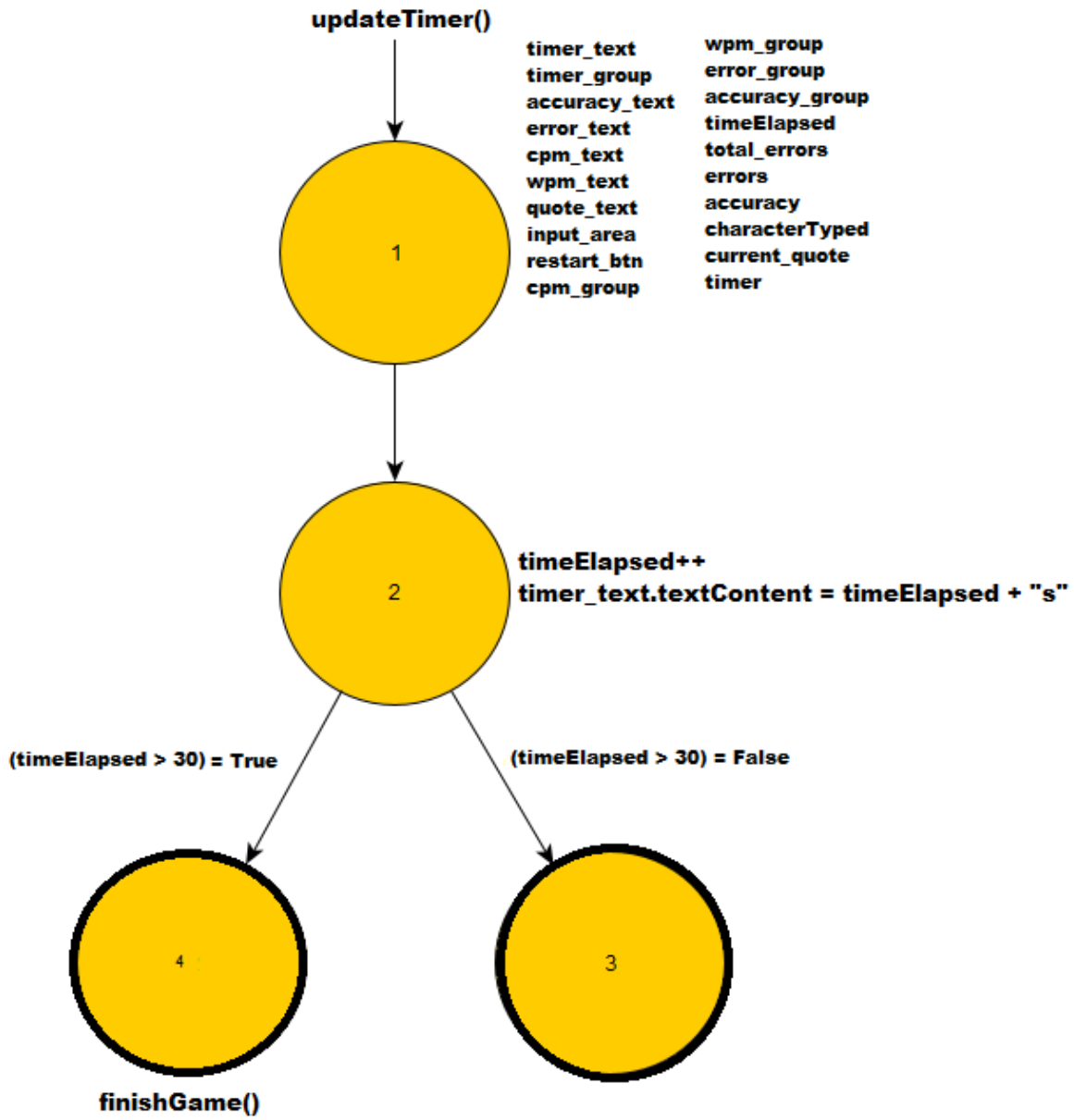
Test Paths

- Unit Level Testing - updateQuote()
  - P1:[1,2,3,4,5,4,6]

Test Cases Design

- Unit Level Testing - updateQuote() - Path 1 - **Test #3 in bModeTest.java**
  - Test values/inputs: The method updateQuote() is called by the startGame() method after the user has clock on the text area. The input for this test would be clicking on the text area.
  - Expected Output: The letter “z” should appear on screen as the unique word in the word-bank is “abcdefghijklmnopqrstuvwxyz” which contains the char “z”.

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - updateTimer()
  - TR: {(1,2), (2,3), (2,4)}

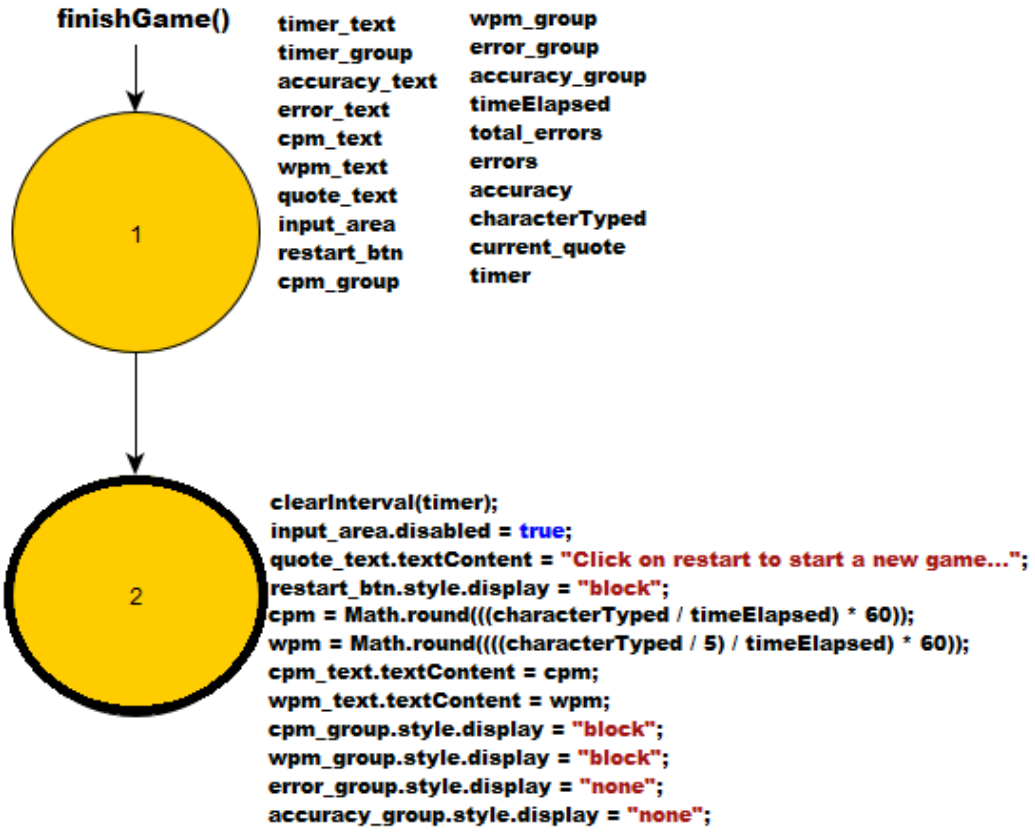
Test Paths

- Unit Level Testing - updateTimer()
  - P1:[1,2,3]
  - P2:[1,2,4]

Test Cases Design

- Unit Level Testing - updateTimer() - Path 1 - **Test #4 in bModeTest.java**
    - Test values/inputs: The method updateTimer() is called every second after the startGame() method is invoked. The input for this test would be clicking on the text area and waiting 3 seconds.
    - Expected Output: The timer should output “3s”.
- Unit Level Testing - updateTimer() - Path 2 - **Test #5 in bModeTest.java**
    - Test values/inputs: The method updateTimer() is called every second after the startGame() method is invoked. The input for this test would be clicking on the text area and waiting 31 seconds so that the method finishGame() (node 4) is called.
    - Expected Output: The timer should freeze at 30 seconds (“30s”).

Nicolas Meneses



TEST REQUIREMENTS (Edge Coverage)

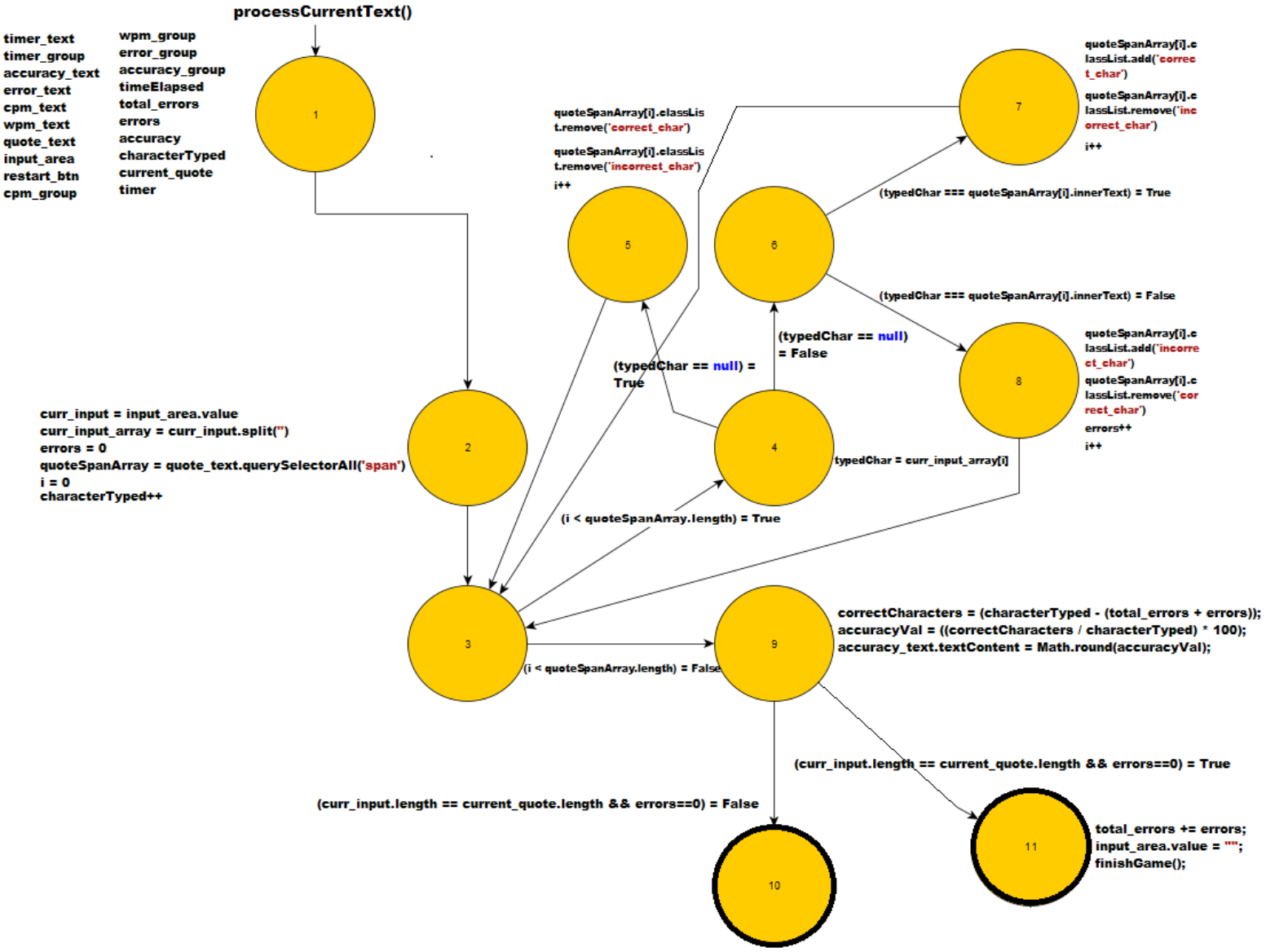
- Unit Level Testing - finishGame()
  - TR: {(1,2)}

Test Paths

- Unit Level Testing - finishGame()
  - P1:[1,2]

Test Cases Design

- Unit Level Testing - finishGame() - Path 1 - **Test #6 in bModeTest.java**
  - Test values/inputs: The method finishGame() is called by the processCurrentText() method when the user has finished inputting the alphabet. The input for this test would be clicking on the text area, typing the alphabet, and waiting 3 seconds.
  - Expected Output: Timer should be “0s”.



TEST REQUIREMENTS (Edge Coverage)

- Unit Level Testing - processCurrentText()
  - TR: {(1,2), (2,3), (3,4), (4,5), (5,3), (4,6), (6,7), (7,3), (6,8), (8,3), (3,9), (9,10), (9,11)}

Test Paths

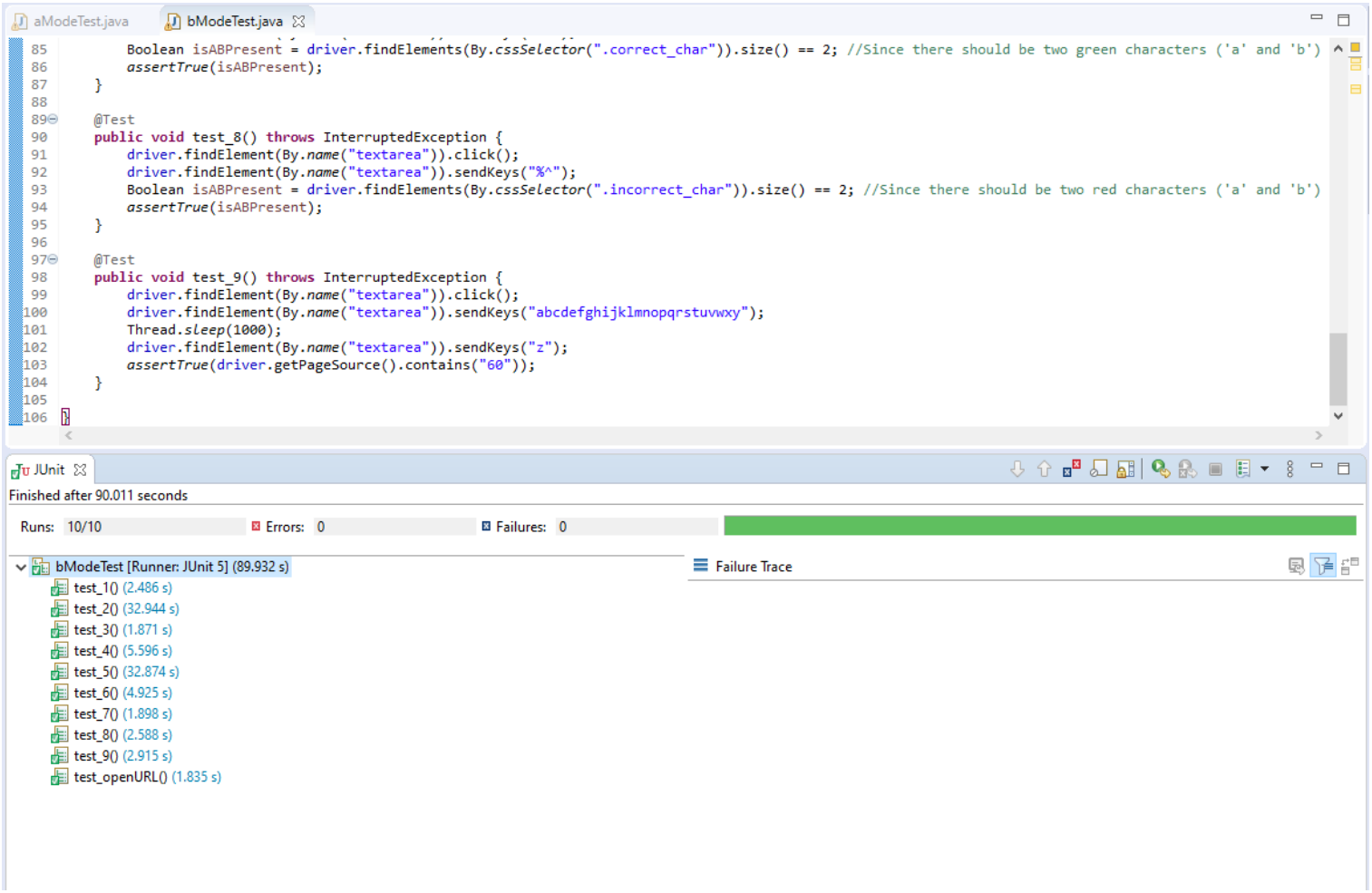
- Unit Level Testing - processCurrentText()
  - P1:[1,2,3,4,6,7,3,4,5,3,9,10]
  - P2:[1,2,3,4,6,8,3,9,10]
  - P3:[1,2,3,9,11]

Nicolas Meneses

Test Cases Design

- Unit Level Testing - processCurrentText() - Path 1 - **Test #7 in bModeTest.java**
  - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be clicking on the text area and typing “ab”.
  - Expected Output: The letters “a” and “b” should be green (.correct_char attribute).
- Unit Level Testing - processCurrentText() - Path 2 - **Test #8 in bModeTest.java**
  - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be typing “%^” in the text area.
  - Expected Output: The letters “a” and “b” should be red (.incorrect_char attribute).
- Unit Level Testing - processCurrentText() - Path 3 - **Test #9 in bModeTest.java**
  - Previous Set-up: Click on the text area and type “abcdefghijklmnopqrstuvwxy”
  - Test values/inputs: The method processCurrentText() is called when the user inputs something in the textbox. The input for this test would be waiting one second and typing “z” in the text area.
  - Expected Output: CPM should be “60”.

Results:



Nicolas Meneses

Test Results:

- We were able to achieve 100% coverage for Edge-Coverage Criteria. Our selected test paths did not cause any test infeasibilities and redundant tests were avoided by capturing the different outputs to the screen like the ‘timer’, the ‘words per minute’, the ‘characters per minute’, the ‘current word prompt’, etc. Furthermore, all of our tests (11 tests for Mode A and 9 for Mode B plus two additional tests not related to Edge-Coverage Criteria) passed. Additionally, it is important to notice the execution time of our tests is relatively high as time is part of the application’s input domain.

References:

- https://www.youtube.com/watch?v=hVdTQWASliE
- https://www.youtube.com/watch?v=_mrj6OW9DA8