IT 314

Software Engineering

RESTAURANT CONTROLLER MUTATION TESTING

Fork & Feast

Group 28

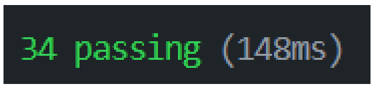
- Tool Used
  Stryker-mutator/mocha-runner @8.6.0

  Link to the source code: [restaurantController]
  Link to its test file: [restaurantController_test]

- Note: For the Mutation Testing we used Stryker which is a tool for performing mutation testing in JavaScript. In this we use the already made unit test file for the particular component and run it after mutating the code. If some mutants are not killed, then we need to add some extra test cases which are redundant for the code coverage as 100% coverage is already achieved but are necessary for mutation testing in order to kill the variant as required.

- In our case we had to add approximately 22 new test cases in order to kill the mutants which got survived because of the changes in the code.

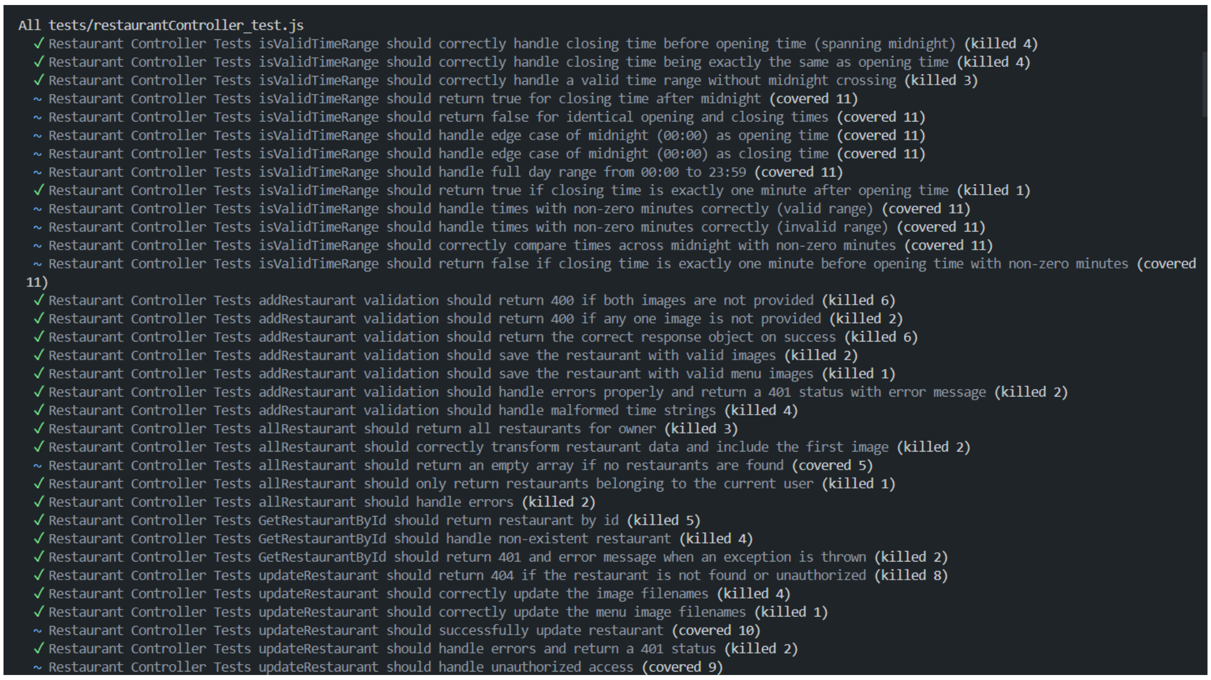
- This is also verified by the output in the terminal which shows all the 34 test cases passing (12 old + 22 new tests).



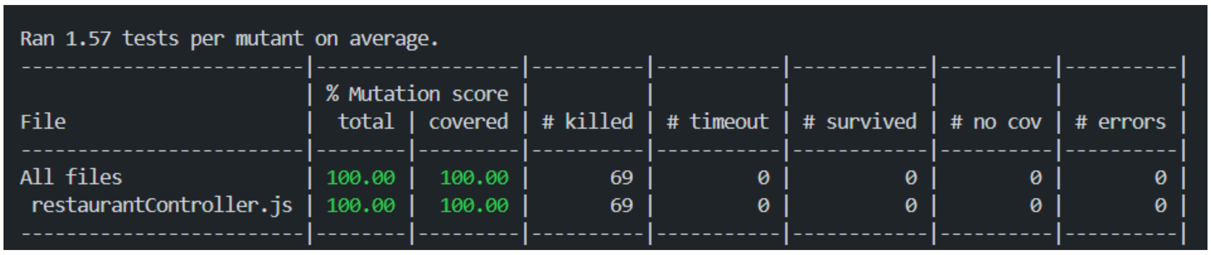
```
34 passing (148ms)
```

- Test Report

  This is a brief test report which shows which tests were present that got killed due to mutant and which didn’t affect the mutant.

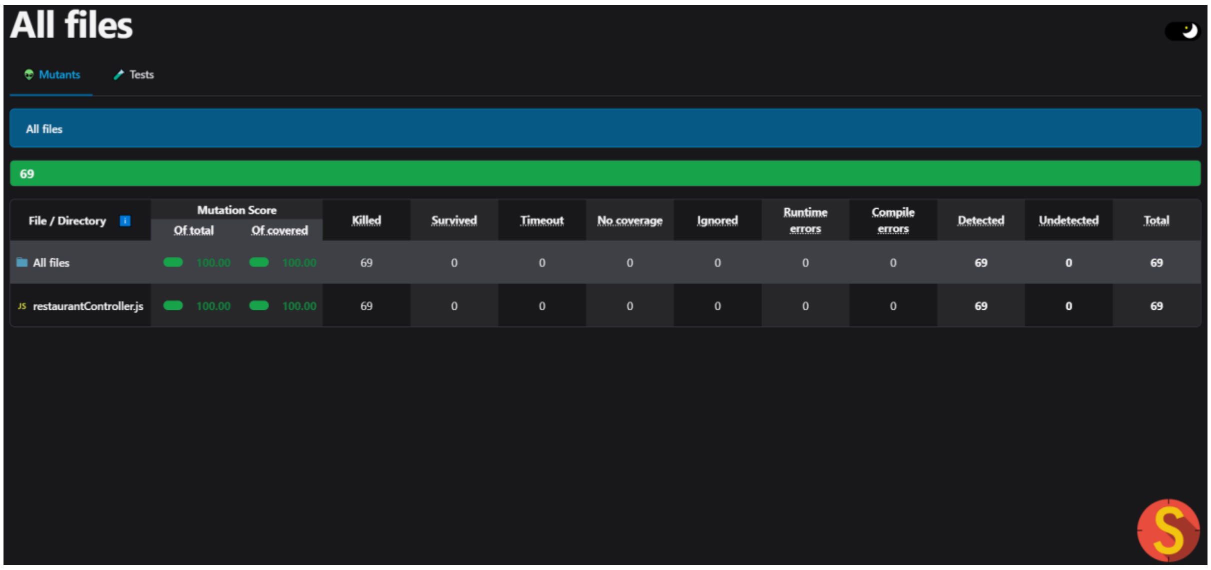


```
All tests/restaurantController_test.js
  ✓ Restaurant Controller Tests isValidTimeRange should correctly handle closing time before opening time (spanning midnight) (killed 4)
  ✓ Restaurant Controller Tests isValidTimeRange should correctly handle closing time being exactly the same as opening time (killed 4)
  ✓ Restaurant Controller Tests isValidTimeRange should correctly handle a valid time range without midnight crossing (killed 3)
  ~ Restaurant Controller Tests isValidTimeRange should return true for closing time after midnight (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should return false for identical opening and closing times (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should handle edge case of midnight (00:00) as opening time (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should handle edge case of midnight (00:00) as closing time (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should handle full day range from 00:00 to 23:59 (covered 11)
  ✓ Restaurant Controller Tests isValidTimeRange should return true if closing time is exactly one minute after opening time (killed 1)
  ~ Restaurant Controller Tests isValidTimeRange should handle times with non-zero minutes correctly (valid range) (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should handle times with non-zero minutes correctly (invalid range) (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should correctly compare times across midnight with non-zero minutes (covered 11)
  ~ Restaurant Controller Tests isValidTimeRange should return false if closing time is exactly one minute before opening time with non-zero minutes (covered
 11)
  ✓ Restaurant Controller Tests addRestaurant validation should return 400 if both images are not provided (killed 6)
  ✓ Restaurant Controller Tests addRestaurant validation should return 400 if any one image is not provided (killed 2)
  ✓ Restaurant Controller Tests addRestaurant validation should return the correct response object on success (killed 6)
  ✓ Restaurant Controller Tests addRestaurant validation should save the restaurant with valid images (killed 2)
  ✓ Restaurant Controller Tests addRestaurant validation should save the restaurant with valid menu images (killed 1)
  ✓ Restaurant Controller Tests addRestaurant validation should handle errors properly and return a 401 status with error message (killed 2)
  ✓ Restaurant Controller Tests addRestaurant validation should handle malformed time strings (killed 4)
  ✓ Restaurant Controller Tests allRestaurant should return all restaurants for owner (killed 3)
  ✓ Restaurant Controller Tests allRestaurant should correctly transform restaurant data and include the first image (killed 2)
  ~ Restaurant Controller Tests allRestaurant should return an empty array if no restaurants are found (covered 5)
  ✓ Restaurant Controller Tests allRestaurant should only return restaurants belonging to the current user (killed 1)
  ✓ Restaurant Controller Tests allRestaurant should handle errors (killed 2)
  ✓ Restaurant Controller Tests GetRestaurantById should return restaurant by id (killed 5)
  ✓ Restaurant Controller Tests GetRestaurantById should handle non-existent restaurant (killed 4)
  ✓ Restaurant Controller Tests GetRestaurantById should return 401 and error message when an exception is thrown (killed 2)
  ✓ Restaurant Controller Tests updateRestaurant should return 404 if the restaurant is not found or unauthorized (killed 8)
  ✓ Restaurant Controller Tests updateRestaurant should correctly update the image filenames (killed 4)
  ✓ Restaurant Controller Tests updateRestaurant should correctly update the menu image filenames (killed 1)
  ~ Restaurant Controller Tests updateRestaurant should successfully update restaurant (covered 10)
  ✓ Restaurant Controller Tests updateRestaurant should handle errors and return a 401 status (killed 2)
  ~ Restaurant Controller Tests updateRestaurant should handle unauthorized access (covered 9)
```



```
Ran 1.57 tests per mutant on average.
------------------------|-----------------|---------|----------|-----------|----------|----------|
                        | % Mutation score |         |          |           |          |          |
File                    |  total | covered | # killed | # timeout | # survived | # no cov | # errors |
------------------------|--------|---------|----------|-----------|------------|----------|----------|
All files               | 100.00 |  100.00 |       69 |         0 |          0 |        0 |        0 |
 restaurantController.js | 100.00 |  100.00 |       69 |         0 |          0 |        0 |        0 |
------------------------|--------|---------|----------|-----------|------------|----------|----------|
```

  This shows the category wise report based on number of cases killed ,survived ,etc. It shows that there were 69 cases that got killed and there were 0 that survived, thereby achieving **“mutation score”** as **1**.



All files

Mutants Tests

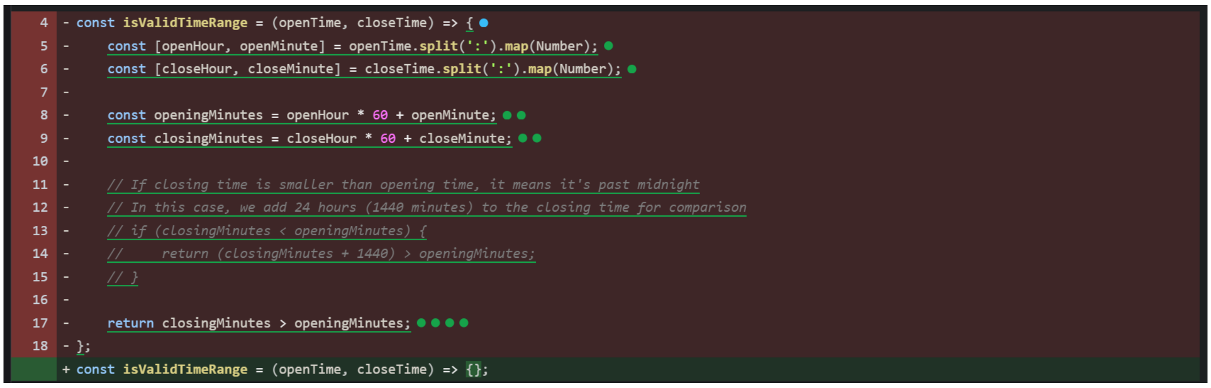
All files

69

File / Directory	Mutation Score		Killed	Survived	Timeout	No coverage	Ignored	Runtime errors	Compile errors	Detected	Undetected	Total
	Of total	Of covered										
All files	100.00	100.00	69	0	0	0	0	0	0	69	0	69
restaurantController.js	100.00	100.00	69	0	0	0	0	0	0	69	0	69

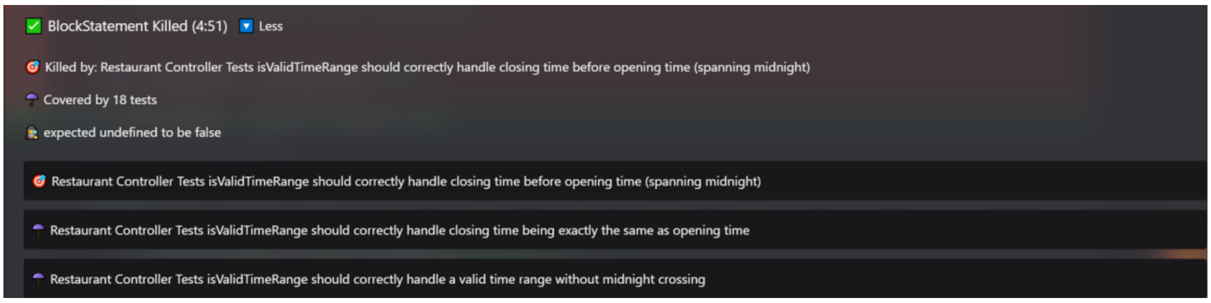
This is the detailed HTML report which is created to show as what all mutants were generated and all.

Below is the sample as to how Stryker helps in creating mutants.



```
- const isValidTimeRange = (openTime, closeTime) => {
-     const [openHour, openMinute] = openTime.split(':').map(Number);
-     const [closeHour, closeMinute] = closeTime.split(':').map(Number);
-
-     const openingMinutes = openHour * 60 + openMinute;
-     const closingMinutes = closeHour * 60 + closeMinute;
-
-     // If closing time is smaller than opening time, it means it's past midnight
-     // In this case, we add 24 hours (1440 minutes) to the closing time for comparison
-     // if (closingMinutes < openingMinutes) {
-     //     return (closingMinutes + 1440) > openingMinutes;
-     // }
-
-     return closingMinutes > openingMinutes;
- };
+ const isValidTimeRange = (openTime, closeTime) => {};
```

And it also gives detailed information about the mutated part as shown below. It shows which test cases got the mutant killed and how many times that was covered and all.



✓ BlockStatement Killed (4:51) Less

Killed by: Restaurant Controller Tests isValidTimeRange should correctly handle closing time before opening time (spanning midnight)

Covered by 18 tests

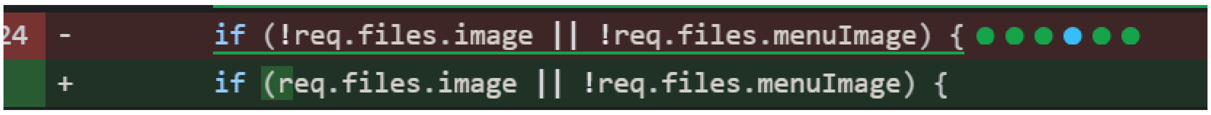
expected undefined to be false

Restaurant Controller Tests isValidTimeRange should correctly handle closing time before opening time (spanning midnight)

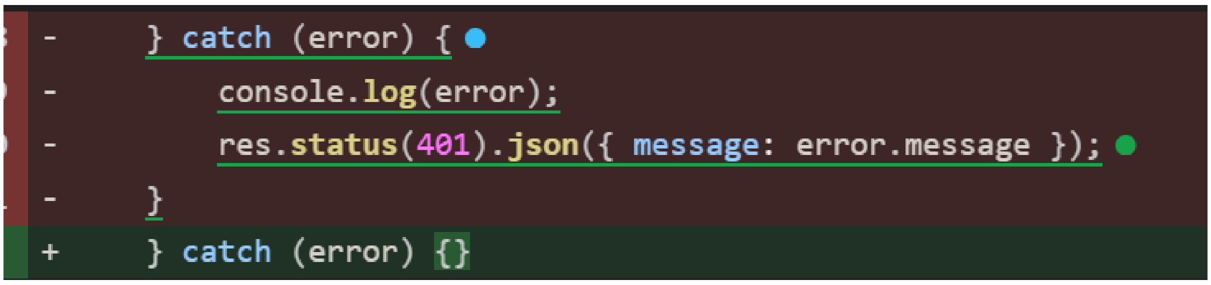
Restaurant Controller Tests isValidTimeRange should correctly handle closing time being exactly the same as opening time

Restaurant Controller Tests isValidTimeRange should correctly handle a valid time range without midnight crossing

Some more examples for the mutants are:



```
-        if (!req.files.image || !req.files.menuImage) {
+        if (req.files.image || !req.files.menuImage) {
```

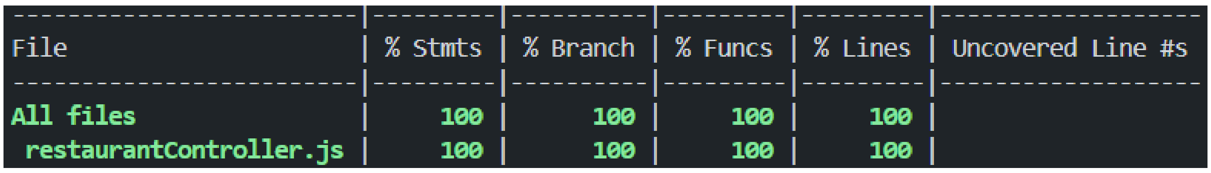


```
-     } catch (error) {
-         console.log(error);
-         res.status(401).json({ message: error.message });
-     }
+     } catch (error) {}
```

and many more….

Thus, Stryker creates different mutants (covering decision mutants, value mutants, statement mutants) by itself giving us the number of killed and survived after which we need to brainstorm in order to get the mutants killed.

- We can also verify the code coverage again that whether it was affected or not.



```
-------------------------|---------|----------|---------|---------|-------------------
File                     | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-------------------------|---------|----------|---------|---------|-------------------
All files                |     100 |      100 |     100 |     100 |
 restaurantController.js |     100 |      100 |     100 |     100 |
```

This shows that adding new test cases does not affect the code coverage.

- **The HTML report present in our “report” folder of our project can be run to see the whole analysis in detail.**