International Institute of Information Technology, Bangalore

CSE 731: Software Testing

Mutation Testing on Healthcare Project

Created By:

Smit Mehta (MT2023109)

ABSTARCT

This project focuses on using a **Mental Healthcare Management System** designed to streamline interactions between patients, doctors, and moderators while ensuring optimal data handling and security. The system incorporates key functionalities like appointment scheduling, question-answer management, and moderator oversight. It features modular components, including services, controllers, and repositories, all working cohesively to deliver a seamless user experience.

To guarantee the reliability and maintainability of the system, we have implemented **extensive unit testing** for all service-layer functionalities. Each service is thoroughly tested using **JUnit 5** to validate various scenarios, including edge cases, expected behaviors, and exception handling. Mocking techniques with tools like **Mockito** are employed to isolate dependencies, ensuring precise validation of individual components.

Post unit testing, the project incorporates **Mutation Testing** using the **PIT (Pitest)** tool to assess the effectiveness of the written tests. Mutation testing works by introducing small, deliberate changes (mutations) to the source code and checking if the existing tests can detect these changes. This technique ensures that our tests are not only present but robust enough to catch subtle errors. Metrics like **mutation coverage** provide deeper insights into the quality of test cases compared to traditional code coverage.

This project demonstrates the critical role of rigorous testing practices in the lifecycle of software development, particularly in a domain like healthcare where data integrity and application reliability are paramount. By integrating **mutation testing** with **unit testing**, we establish a robust foundation for building reliable and high-quality software solutions.

Mutation Testing

Mutation testing is a fault-based testing technique used to evaluate the effectiveness of a software testing suite. The core idea is to introduce small changes (mutations) in the codebase and check whether the existing tests can detect these changes. If the tests fail to identify the mutation, it indicates a potential weakness in the test suite, suggesting that the tests are not sufficiently robust.

How it Works:

1. **Mutants Generation:** Small, syntactic changes are made to the program code, such as modifying arithmetic operators, changing variable names, or altering control flow.
2. **Test Execution:** The modified code (mutant) is executed using the existing test suite.
3. **Analysis:** If a test case detects the mutation and fails (compared to the original code), the mutant is considered "killed." If no test case catches the mutation, the mutant survives, highlighting a gap in the test coverage.

Benefits:

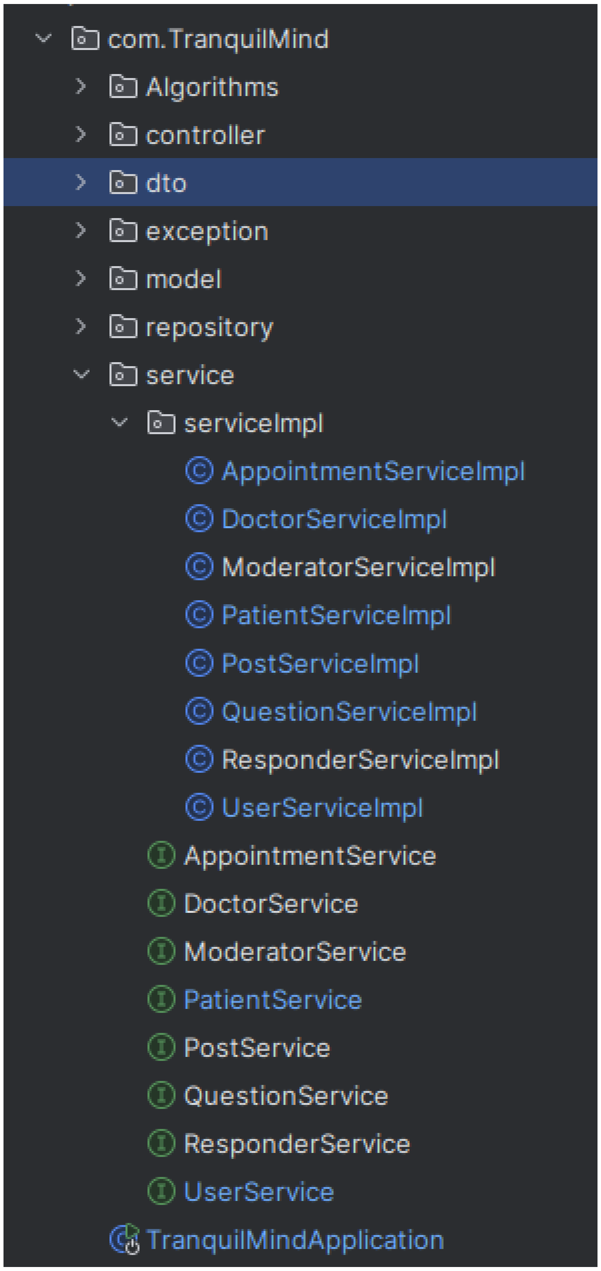
- **Test Quality Assessment:** It helps in assessing the quality of test cases by measuring their ability to detect code changes.
- **Improvement of Test Suite:** Mutation testing helps in identifying areas where tests can be improved or extended.
- **Enhanced Reliability:** It provides insights into how well the software can handle small code changes, ensuring robustness.

Mutation testing is an important method for improving the rigor and effectiveness of testing processes, helping to ensure that software is thoroughly validated against potential errors.

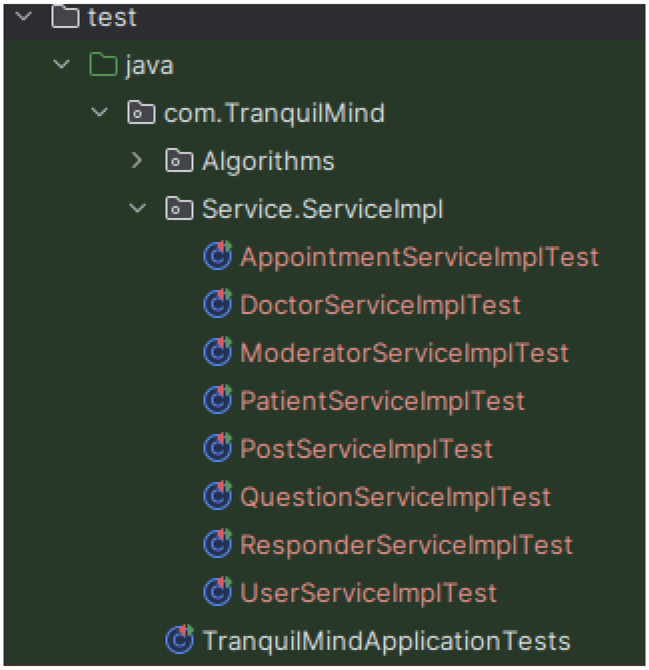
Target classed and Test classes:

I have written multiple service classes regarding the business logic of functionalities of patient, doctor, moderator etc. I have also written test classes for unit testing of these given classes.

Source code : https://github.com/SmitMehta510/Mutation-testing



```
com.TranquilMind
    Algorithms
    controller
    dto
    exception
    model
    repository
    service
        serviceImpl
            AppointmentServiceImpl
            DoctorServiceImpl
            ModeratorServiceImpl
            PatientServiceImpl
            PostServiceImpl
            QuestionServiceImpl
            ResponderServiceImpl
            UserServiceImpl
        AppointmentService
        DoctorService
        ModeratorService
        PatientService
        PostService
        QuestionService
        ResponderService
        UserService
    TranquilMindApplication
```



```
test
  java
    com.TranquilMind
      Algorithms
      Service.ServiceImpl
        AppointmentServiceImplTest
        DoctorServiceImplTest
        ModeratorServiceImplTest
        PatientServiceImplTest
        PostServiceImplTest
        QuestionServiceImplTest
        ResponderServiceImplTest
        UserServiceImplTest
      TranquilMindApplicationTests
```

Tool used: PIT Testing

PIT (or *PIT*, which stands for *PIT Mutation Testing*) is a lightweight and fast mutation testing tool for Java applications. It is designed to help developers improve the quality of their unit tests by assessing how well they can detect small changes (mutations) introduced to the code. PIT testing provides a fault-based approach to testing by simulating errors in the code and checking if the existing test suite can identify them.

How it Works:

1. **Mutant Generation:** PIT automatically generates a set of mutants by making small changes to the code. These changes could include modifying operators, changing conditions, or altering method calls.
2. **Test Execution:** The generated mutants are then tested using the existing unit tests in the project.
3. **Mutation Coverage:** The tests are evaluated based on whether they can detect and "kill" the mutants. If a test passes without detecting the mutation, it suggests that the test might not be comprehensive enough.

Benefits of PIT Testing:

- **Fast and Efficient:** PIT is optimized for speed and provides quick feedback on test effectiveness without requiring extensive setup.
- **Enhanced Test Quality:** By identifying weak test cases, PIT helps developers improve their test coverage and overall quality.
- **Integration-Friendly:** PIT integrates well with build tools like Maven and Gradle, making it easy to use within continuous integration (CI) pipelines.

Overall, PIT testing is a valuable tool for ensuring the robustness of unit tests, identifying weaknesses in the test suite, and improving the reliability of software applications.

Here are the general steps to perform mutation testing using PIT :

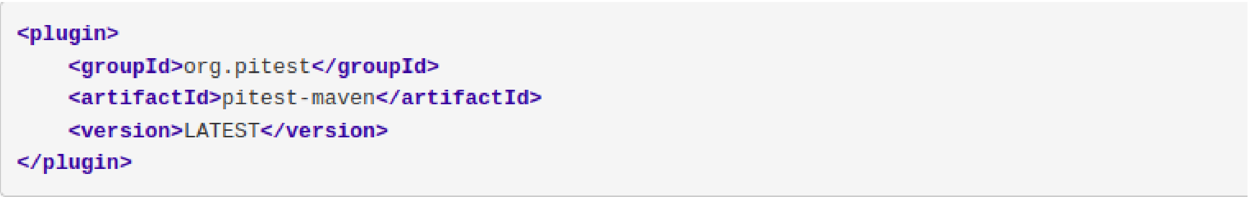
✓ Add PIT plugin to your project

✓ Configure mutators

✓ Run PIT Mutation Testing

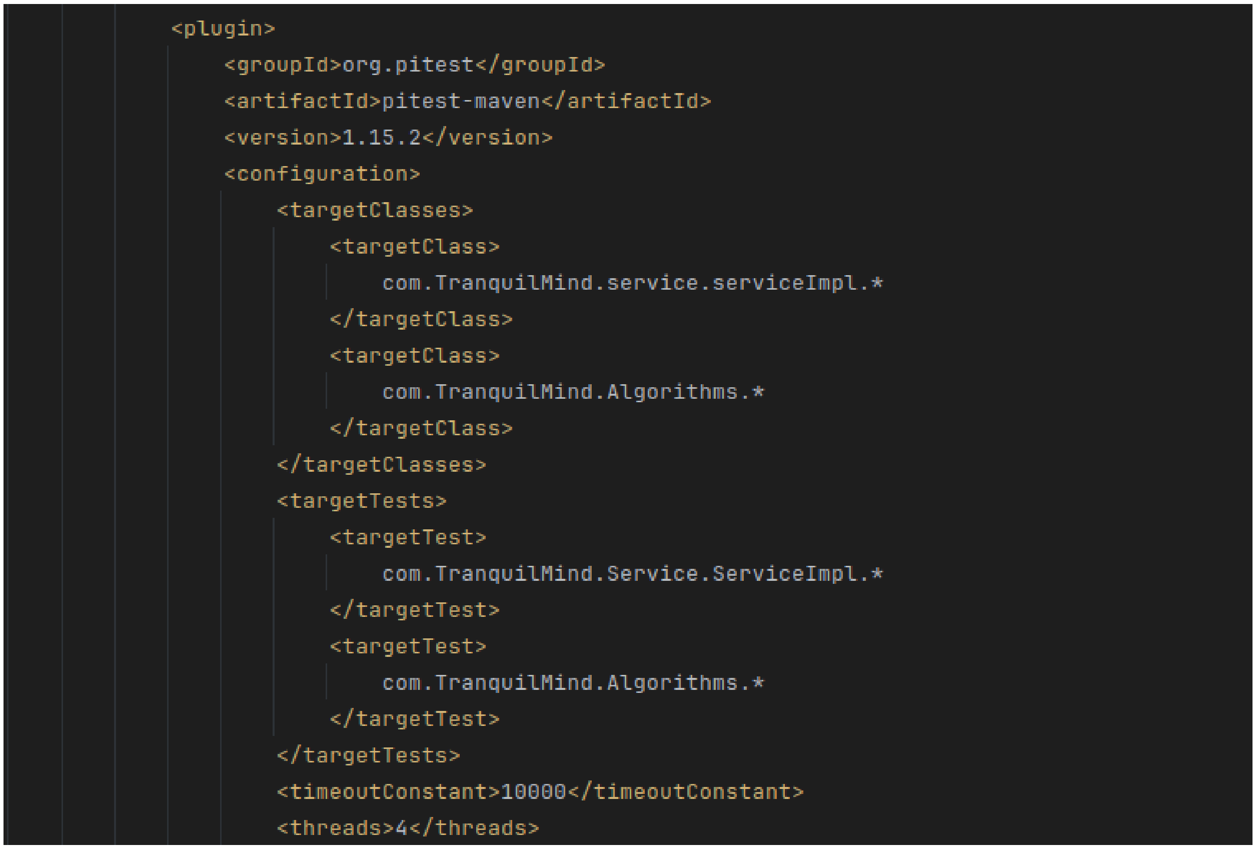
✓ View the reports

1. Add PIT plugin to your project: Add the PIT Maven plugin to your pom.xml file. Ensure that you specify the necessary configuration for your project. By default pitest will mutate all code in your project.

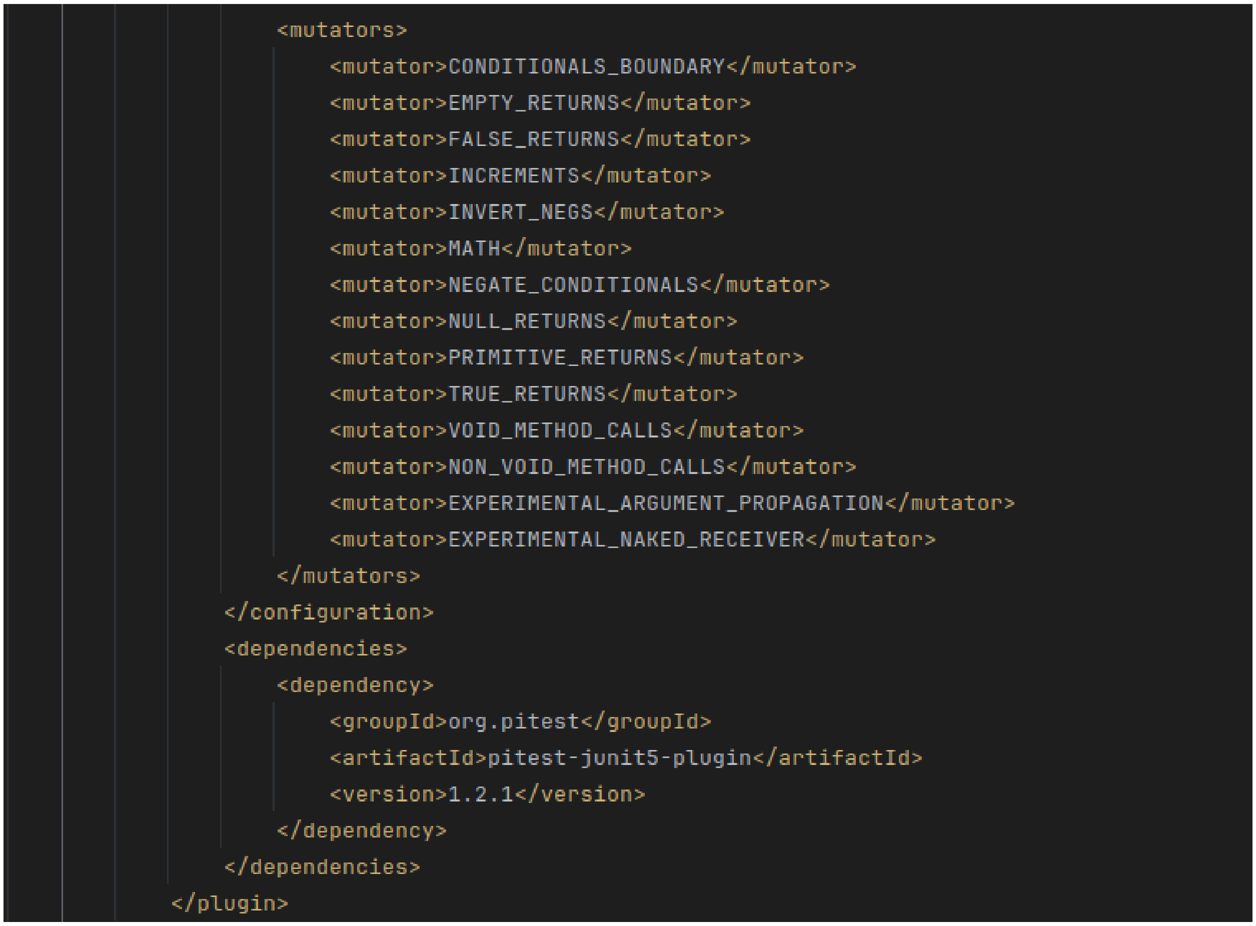


```
<plugin>
    <groupId>org.pitest</groupId>
    <artifactId>pitest-maven</artifactId>
    <version>LATEST</version>
</plugin>
```

You can limit which code is mutated and which tests are run using targetClasses and targetTests.



```
<plugin>
    <groupId>org.pitest</groupId>
    <artifactId>pitest-maven</artifactId>
    <version>1.15.2</version>
    <configuration>
        <targetClasses>
            <targetClass>
                com.TranquilMind.service.serviceImpl.*
            </targetClass>
            <targetClass>
                com.TranquilMind.Algorithms.*
            </targetClass>
        </targetClasses>
        <targetTests>
            <targetTest>
                com.TranquilMind.Service.ServiceImpl.*
            </targetTest>
            <targetTest>
                com.TranquilMind.Algorithms.*
            </targetTest>
        </targetTests>
        <timeoutConstant>10000</timeoutConstant>
        <threads>4</threads>
```



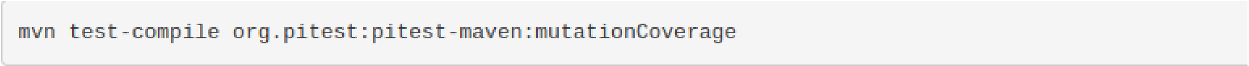
```
                    <mutators>
                        <mutator>CONDITIONALS_BOUNDARY</mutator>
                        <mutator>EMPTY_RETURNS</mutator>
                        <mutator>FALSE_RETURNS</mutator>
                        <mutator>INCREMENTS</mutator>
                        <mutator>INVERT_NEGS</mutator>
                        <mutator>MATH</mutator>
                        <mutator>NEGATE_CONDITIONALS</mutator>
                        <mutator>NULL_RETURNS</mutator>
                        <mutator>PRIMITIVE_RETURNS</mutator>
                        <mutator>TRUE_RETURNS</mutator>
                        <mutator>VOID_METHOD_CALLS</mutator>
                        <mutator>NON_VOID_METHOD_CALLS</mutator>
                        <mutator>EXPERIMENTAL_ARGUMENT_PROPAGATION</mutator>
                        <mutator>EXPERIMENTAL_NAKED_RECEIVER</mutator>
                    </mutators>
                </configuration>
                <dependencies>
                    <dependency>
                        <groupId>org.pitest</groupId>
                        <artifactId>pitest-junit5-plugin</artifactId>
                        <version>1.2.1</version>
                    </dependency>
                </dependencies>
            </plugin>
```

Configure Mutators:

- **CONDITIONALS_BOUNDARY**: Alters relational operators to test edge cases (e.g., < to <=).
- **EMPTY_RETURNS**: Forces methods to return default or empty values.
- **EXPERIMENTAL_ARGUMENT_PROPAGATION**: Propagates arguments to different places to test resilience.
- **EXPERIMENTAL_NAKED_RECEIVER**: Replaces method receiver with this in experimental ways.
- **FALSE_RETURNS**: Replaces return values with false.
- **INCREMENTS**: Modifies increment/decrement operations (e.g., + + to - -).
- **INVERT_NEGS**: Reverses the sign of numeric values.
- **MATH**: Changes arithmetic operations (e.g., + to -).
- **NEGATE_CONDITIONALS**: Negates conditional expressions
- **NON_VOID_METHOD_CALLS**: Removes calls to methods that return a value.
- **NULL_RETURNS**: Makes methods return null.
- **PRIMITIVE_RETURNS**: Substitutes primitive return values with defaults (e.g., 0).
- **TRUE_RETURNS**: Replaces return values with true.
- **VOID_METHOD_CALLS**: Removes calls to void methods.

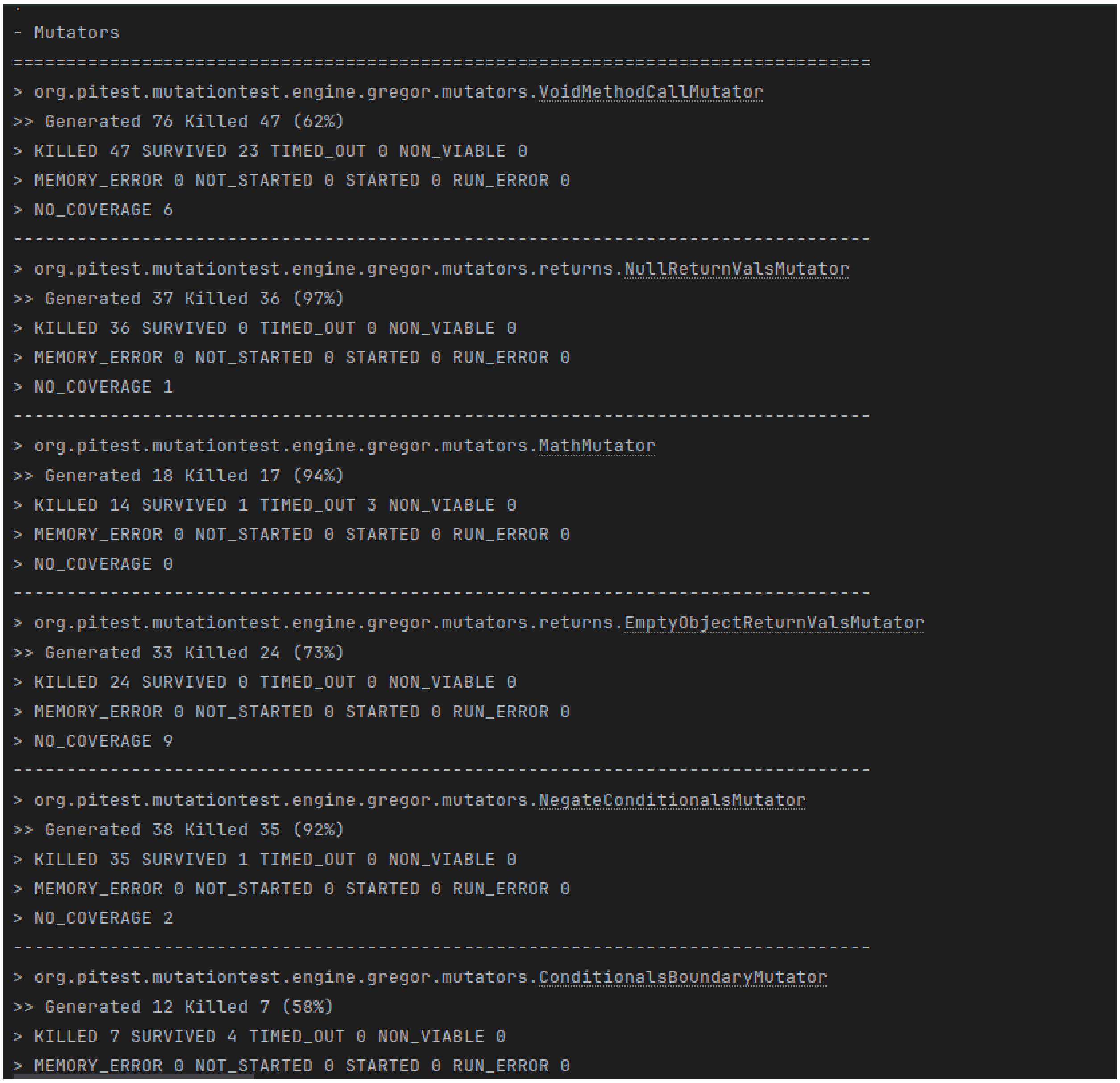
Run PIT runner:

It can be run directly from the command-line:

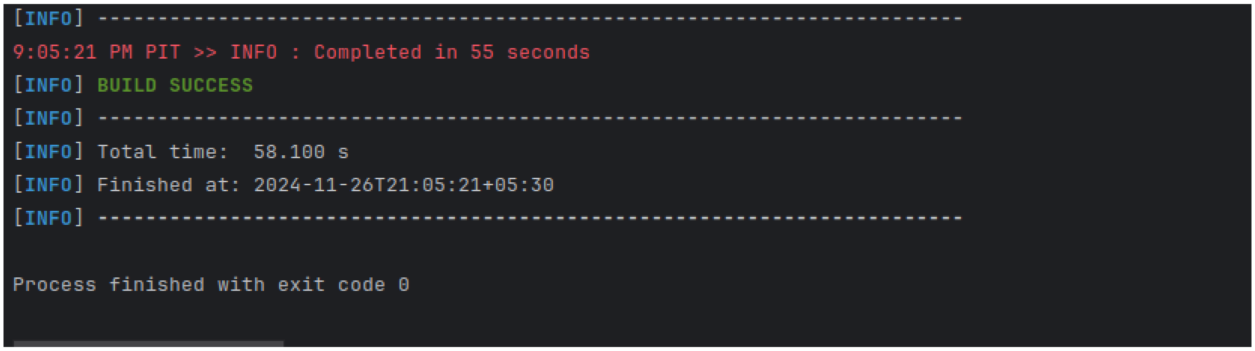


```
mvn test-compile org.pitest:pitest-maven:mutationCoverage
```

This command will create mutants and run the mutation testing on the codebase, the resultant report will be generated in the target folder as an HTML document.



```
- Mutators
================================================================================
> org.pitest.mutationtest.engine.gregor.mutators.VoidMethodCallMutator
>> Generated 76 Killed 47 (62%)
> KILLED 47 SURVIVED 23 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 6
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.returns.NullReturnValsMutator
>> Generated 37 Killed 36 (97%)
> KILLED 36 SURVIVED 0 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 1
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.MathMutator
>> Generated 18 Killed 17 (94%)
> KILLED 14 SURVIVED 1 TIMED_OUT 3 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 0
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.returns.EmptyObjectReturnValsMutator
>> Generated 33 Killed 24 (73%)
> KILLED 24 SURVIVED 0 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 9
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.NegateConditionalsMutator
>> Generated 38 Killed 35 (92%)
> KILLED 35 SURVIVED 1 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 2
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.ConditionalsBoundaryMutator
>> Generated 12 Killed 7 (58%)
> KILLED 7 SURVIVED 4 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
```

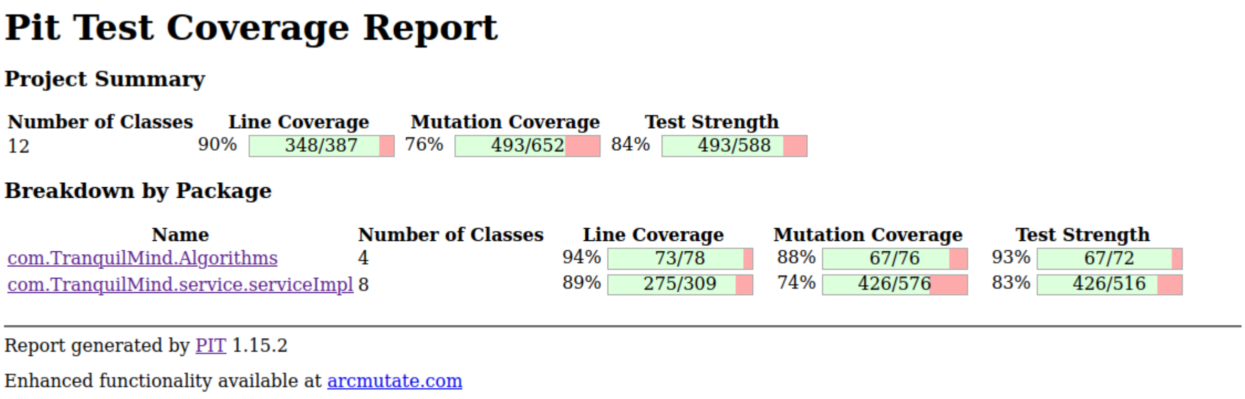


```
[INFO] ------------------------------------------------------------------------
9:05:21 PM PIT >> INFO : Completed in 55 seconds
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  58.100 s
[INFO] Finished at: 2024-11-26T21:05:21+05:30
[INFO] ------------------------------------------------------------------------

Process finished with exit code 0
```

Checking the reports:

After the execution is complete, we can find the HTML report in the target/pit-reports directory. Open the index.html file in a browser to view the detailed mutation testing report.



Pit Test Coverage Report

Project Summary

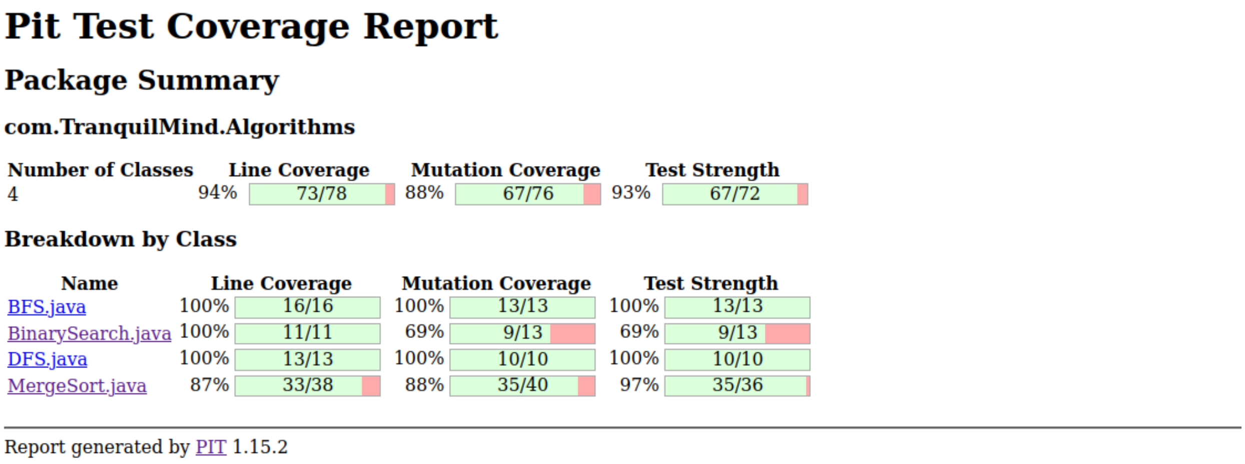
Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
12	90%	348/387	76%	493/652	84%	493/588

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
com.TranquilMind.Algorithms	4	94%	73/78	88%	67/76	93%	67/72
com.TranquilMind.service.serviceImpl	8	89%	275/309	74%	426/576	83%	426/516

Report generated by PIT 1.15.2

Enhanced functionality available at arcmutate.com



Pit Test Coverage Report

Package Summary

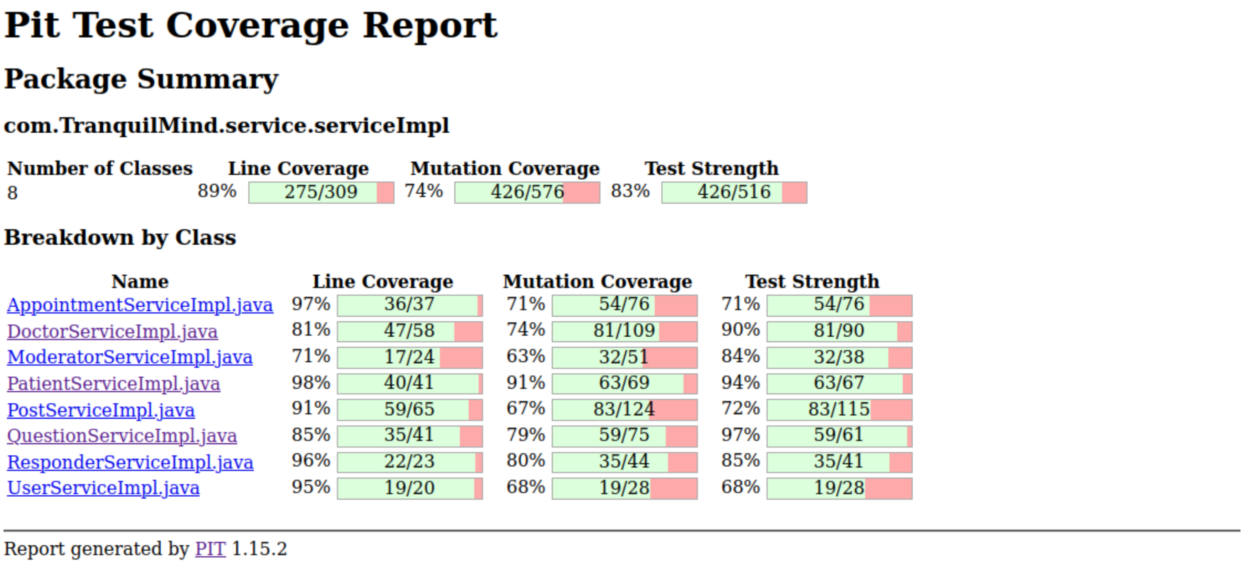
com.TranquilMind.Algorithms

Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
4	94%	73/78	88%	67/76	93%	67/72

Breakdown by Class

Name	Line Coverage		Mutation Coverage		Test Strength	
BFS.java	100%	16/16	100%	13/13	100%	13/13
BinarySearch.java	100%	11/11	69%	9/13	69%	9/13
DFS.java	100%	13/13	100%	10/10	100%	10/10
MergeSort.java	87%	33/38	88%	35/40	97%	35/36

Report generated by PIT 1.15.2



Pit Test Coverage Report

Package Summary

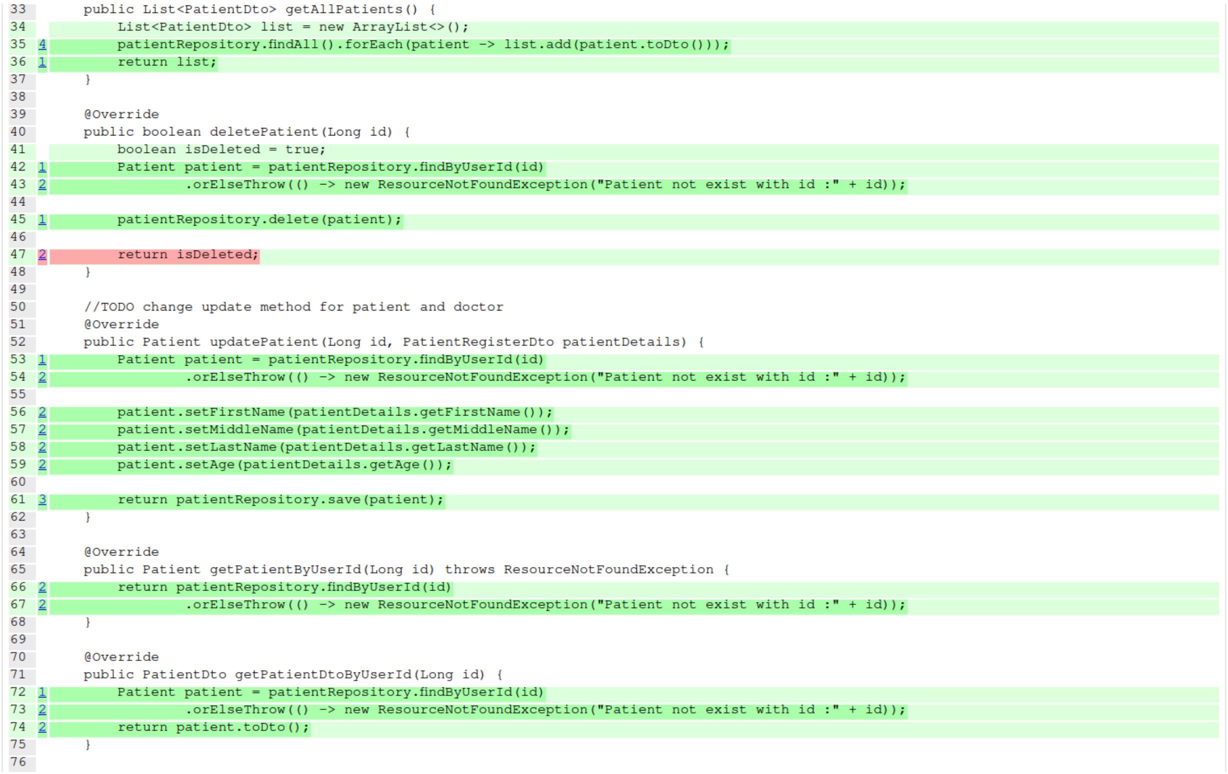
com.TranquilMind.service.serviceImpl

Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
8	89%	275/309	74%	426/576	83%	426/516

Breakdown by Class

Name	Line Coverage		Mutation Coverage		Test Strength	
AppointmentServiceImpl.java	97%	36/37	71%	54/76	71%	54/76
DoctorServiceImpl.java	81%	47/58	74%	81/109	90%	81/90
ModeratorServiceImpl.java	71%	17/24	63%	32/51	84%	32/38
PatientServiceImpl.java	98%	40/41	91%	63/69	94%	63/67
PostServiceImpl.java	91%	59/65	67%	83/124	72%	83/115
QuestionServiceImpl.java	85%	35/41	79%	59/75	97%	59/61
ResponderServiceImpl.java	96%	22/23	80%	35/44	85%	35/41
UserServiceImpl.java	95%	19/20	68%	19/28	68%	19/28

Report generated by PIT 1.15.2



```
       public List<PatientDto> getAllPatients() {
           List<PatientDto> list = new ArrayList<>();
4          patientRepository.findAll().forEach(patient -> list.add(patient.toDto()));
1          return list;
       }

       @Override
       public boolean deletePatient(Long id) {
           boolean isDeleted = true;
1          Patient patient = patientRepository.findByUserId(id)
2                  .orElseThrow(() -> new ResourceNotFoundException("Patient not exist with id :" + id));

1          patientRepository.delete(patient);

2          return isDeleted;
       }

       //TODO change update method for patient and doctor
       @Override
       public Patient updatePatient(Long id, PatientRegisterDto patientDetails) {
1          Patient patient = patientRepository.findByUserId(id)
2                  .orElseThrow(() -> new ResourceNotFoundException("Patient not exist with id :" + id));

2          patient.setFirstName(patientDetails.getFirstName());
2          patient.setMiddleName(patientDetails.getMiddleName());
2          patient.setLastName(patientDetails.getLastName());
2          patient.setAge(patientDetails.getAge());

3          return patientRepository.save(patient);
       }

       @Override
       public Patient getPatientByUserId(Long id) throws ResourceNotFoundException {
2          return patientRepository.findByUserId(id)
2                  .orElseThrow(() -> new ResourceNotFoundException("Patient not exist with id :" + id));
       }

       @Override
       public PatientDto getPatientDtoByUserId(Long id) {
1          Patient patient = patientRepository.findByUserId(id)
2                  .orElseThrow(() -> new ResourceNotFoundException("Patient not exist with id :" + id));
2          return patient.toDto();
       }

```

35	2. removed call to java/util/List::forEach → KILLED 3. removed call to com/TranquilMind/model/Patient::toDto → KILLED 4. removed call to com/TranquilMind/repository/PatientRepository::findAll → KILLED
36	1. replaced return value with Collections.emptyList for com/TranquilMind/service/serviceImpl/PatientServiceImpl::getAllPatients → KILLED
42	1. removed call to com/TranquilMind/repository/PatientRepository::findByUserId → KILLED
43	1. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::lambda$deletePatient$1 → KILLED 2. removed call to java/util/Optional::orElseThrow → KILLED
45	1. removed call to com/TranquilMind/repository/PatientRepository::delete → KILLED
47	1. replaced boolean return with false for com/TranquilMind/service/serviceImpl/PatientServiceImpl::deletePatient → KILLED 2. replaced boolean return with true for com/TranquilMind/service/serviceImpl/PatientServiceImpl::deletePatient → SURVIVED
53	1. removed call to com/TranquilMind/repository/PatientRepository::findByUserId → KILLED
54	1. removed call to java/util/Optional::orElseThrow → KILLED 2. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::lambda$updatePatient$2 → KILLED
56	1. removed call to com/TranquilMind/dto/PatientRegisterDto::getFirstName → KILLED 2. removed call to com/TranquilMind/model/Patient::setFirstName → KILLED
57	1. removed call to com/TranquilMind/dto/PatientRegisterDto::getMiddleName → KILLED 2. removed call to com/TranquilMind/model/Patient::setMiddleName → KILLED
58	1. removed call to com/TranquilMind/model/Patient::setLastName → KILLED 2. removed call to com/TranquilMind/dto/PatientRegisterDto::getLastName → KILLED
59	1. removed call to com/TranquilMind/dto/PatientRegisterDto::getAge → KILLED 2. removed call to com/TranquilMind/model/Patient::setAge → KILLED
61	1. replaced call to com/TranquilMind/repository/PatientRepository::save with argument → KILLED 2. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::updatePatient → KILLED 3. removed call to com/TranquilMind/repository/PatientRepository::save → KILLED
66	1. removed call to com/TranquilMind/repository/PatientRepository::findByUserId → KILLED 2. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::getPatientByUserId → KILLED
67	1. removed call to java/util/Optional::orElseThrow → KILLED 2. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::lambda$getPatientByUserId$3 → KILLED
72	1. removed call to com/TranquilMind/repository/PatientRepository::findByUserId → KILLED
73	1. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::lambda$getPatientDtoByUserId$4 → KILLED 2. removed call to java/util/Optional::orElseThrow → KILLED
74	1. removed call to com/TranquilMind/model/Patient::toDto → KILLED 2. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::getPatientDtoByUserId → KILLED
80	1. removed call to com/TranquilMind/dto/PatientRegisterDto::getPassword → KILLED 2. removed call to com/TranquilMind/dto/PatientRegisterDto::getEmail → KILLED
82	1. removed call to com/TranquilMind/service/UserService::register → KILLED
84	1. removed call to org/springframework/http/HttpStatusCode::is2xxSuccessful → KILLED 2. removed call to org/springframework/http/ResponseEntity::getStatusCode → KILLED 3. removed call to com/TranquilMind/dto/RegisterDto::getResponse → KILLED 4. negated conditional → KILLED
85	1. removed call to com/TranquilMind/service/serviceImpl/PatientServiceImpl::getPatient → KILLED
86	1. removed call to com/TranquilMind/repository/PatientRepository::save → KILLED 2. replaced call to com/TranquilMind/repository/PatientRepository::save with argument → KILLED
87	1. replaced return value with null for com/TranquilMind/service/serviceImpl/PatientServiceImpl::createPatient → KILLED
95	1. removed call to com/TranquilMind/dto/RegisterDto::getUser → SURVIVED 2. removed call to com/TranquilMind/model/Patient::setUser → SURVIVED
96	1. removed call to com/TranquilMind/model/Patient::setFirstName → KILLED 2. removed call to com/TranquilMind/dto/PatientRegisterDto::getFirstName → KILLED
	1. removed call to com/TranquilMind/dto/PatientRegisterDto::getMiddleName → KILLED

Active mutators

- CONDITIONALS_BOUNDARY
- EMPTY_RETURNS
- EXPERIMENTAL_ARGUMENT_PROPAGATION
- EXPERIMENTAL_NAKED_RECEIVER
- FALSE_RETURNS
- INCREMENTS
- INVERT_NEGS
- MATH
- NEGATE_CONDITIONALS
- NON_VOID_METHOD_CALLS
- NULL_RETURNS
- PRIMITIVE_RETURNS
- TRUE_RETURNS
- VOID_METHOD_CALLS

Tests examined

- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:createPatient_shouldCreatePatientSuccessfully()] (85 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getAllPatients_shouldReturnListOfPatients()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:testGetAllPatients()] (6 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:updatePatient_shouldUpdateExistingPatient()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getPatientByUserId_shouldReturnPatientWhenExists()] (3 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:testGetPatientByUserIdNotFound()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:testGetPatientByUserId()] (6 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getPatientByUserId_shouldThrowExceptionWhenPatientDoesNotExist()] (6 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:updatePatient_shouldThrowExceptionIfPatientDoesNotExist()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:createPatient_shouldThrowExceptionIfRegistrationFails()] (6 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:deletePatient_shouldThrowExceptionWhenPatientDoesNotExist()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:updatePassword_shouldUpdatePasswordSuccessfully()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getPosts_shouldReturnPostsForUser()] (4 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getPatientDtoByUserId_shouldThrowExceptionWhenPatientDoesNotExist()] (4 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:getPatientDtoByUserId_shouldReturnPatientDtoWhenExists()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:testDeletePatient()] (5 ms)
- com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest.[engine:junit-jupiter]/[class:com.TranquilMind.Service.ServiceImpl.PatientServiceImplTest]/[method:deletePatient_shouldDeletePatientWhenExists()] (5 ms)