

Raport Javor – Test Driven Development (TDD)

Informata të përgjithshme

Emri: Getoar Krasniqi

Java: 2 (e dytë)

Data e dorëzimit: 20/11/2025

Emri i projektit: Quiz Application

1 Përbledhje e javës

Në javën e dytë është implementuar kodi minimal i mjaftueshëm në mënyrë që testet e shkruara në javën e parë të kalojnë me sukses.

Përditësimi i README.md file-it në përputhje me ndryshimet e bëra në kod.

Gjenerimi i index.html me përqindje të mbulueshmërisë 96%.

Krijimi i "v1.0 Red Phase", i cili është release-i i parë i aplikacionit me kodin e javës së parë.

Krijimi i "v2.0 Green Phase", release-i i dytë me të gjitha testet të kalueshme me sukses dhe me CI pipeline të suksesshme.

2 Testet e zhvilluara

Nr	Përshkrimi i testit	Statusi (kalon/dështon)	Framework /
1	test_create_multiple_choice_question	kalon	pytest
2	test_check_correct_answer	kalon	pytest
3	test_question_requires_text	kalon	pytest
4	test_create_empty_quiz	kalon	pytest
5	test_add_question_to_quiz	kalon	pytest
6	test_cannot_add_duplicate_questions	kalon	pytest
7	test_get_question_by_index	kalon	pytest
8	test_submit_answer_and_track_score	kalon	pytest
9	test_perfect_score	kalon	pytest
10	test_passing_grade	kalon	pytest

11	test_result_summary	kalon	pytest
12	test_complete_quiz_flow	kalon	pytest

3 Implementimi i bërë pas testeve

Implemetimet e bëra pas testeve:

Moduli question.py

```
from typing import List

class Question:
    """Represents a single quiz question with multiple choice options"""

    def __init__(self, text: str, options: List[str], correct_answer: str) -> None:
        if not text or text.strip() == "":
            raise ValueError("Question text cannot be empty")

        self.text = text
        self.options = options
        self.correct_answer = correct_answer

    def check_answer(self, answer: str) -> bool:
        """Check if the provided answer is correct"""
        return answer == self.correct_answer

    def __eq__(self, other: object) -> bool:
        """Check equality based on question text and options"""
        if not isinstance(other, Question):
            return False
        return (
            self.text == other.text
            and self.options == other.options
            and self.correct_answer == other.correct_answer
        )

    def __hash__(self) -> int:
        """Make Question hashable for use in sets"""
        return hash((self.text, tuple(self.options), self.correct_answer))
```

Moduli quiz.py

```
from typing import Dict, List
from src.question import Question
from src.result import QuizResult


class Quiz:
    """Represents a quiz with multiple questions"""

    def __init__(self, title: str) -> None:
        self.title = title
        self.questions: List[Question] = []
        self.answers: Dict[int, str] = {} # Maps question index to submitted answer

    def add_question(self, question: Question) -> None:
        """Add a question to the quiz, avoiding duplicates"""
        if question not in self.questions:
            self.questions.append(question)

    def get_question(self, index: int) -> Question:
        """Get a question by its index"""
        return self.questions[index]

    def submit_answer(self, question_index: int, answer: str) -> None:
        """Submit an answer for a specific question"""
        self.answers[question_index] = answer

    def get_result(self) -> QuizResult:
        """Calculate and return the quiz result"""
        score = 0
        total = len(self.questions)

        for index, answer in self.answers.items():
            if index < len(self.questions):
                question = self.questions[index]
                if question.check_answer(answer):
                    score += 1

        return QuizResult(score, total)
```

Moduli result.py

```
class QuizResult:  
    """Represents the result of a completed quiz"""  
  
    def __init__(self, score: int, total: int) -> None:  
        self.score = score  
        self.total = total  
        self.percentage = (score / total * 100) if total > 0 else 0.0  
  
    def is_perfect(self) -> bool:  
        """Check if the score is perfect (100%)"""  
        return self.score == self.total  
  
    def is_passing(self, threshold: int = 60) -> bool:  
        """Check if the score meets the passing threshold"""  
        return self.percentage >= threshold  
  
    def get_summary(self) -> str:  
        """Get a formatted summary of the results"""  
        return f"Score: {self.score}/{self.total} ({self.percentage:.1f}%)"
```

4 Refaktorimi i bërë

Në javën e dytë nuk ka pasur nevojë për refaktorime të testeve të shkruara në javën e parë.

5 Probleme të hasura

Gjatë implementimit të kodit filletar të projektit në mënyrë që testet të kalojnë me sukses nuk kam hasur në ndonjë problem.

6 Plani për javën e ardhshme

Gjatë javës tjetër planifikoj të implementoj disa features të reja: Limiti i kohës për kuize, nivele të ndryshme të vështirësisë dhe mundësia e ndarjes së kuizeve sipas kategorive. Implementimi i këtyre funksionalitetave të reja do të bëhet në parim të metodologjisë TDD.

Gjatë javës së ardhshme po ashtu planifikohet të bëhet lidhja e aplikacionit me databazë dhe vurja në funksion e API-ve.

7 Mbulesa e testimit (nëse aplikohet)

- % e mbulesës aktuale: 96%
- Vegla e përdorur: pytest-cov

8 Komente / reflektimi të grupit

Para implementimit të kodit, mbulueshmëria e testeve ishte 100%, tanjë është 96% për shkak të metodave `_eq_` dhe `_hash_` të klasës `Question`, dhe mungesa e përdorimit adekuat të tyre në testet e Red Phase.

⌚ Ngjitjet (opcionale)

- Screenshots nga testet që kalojnë / dështojnë

```
PS C:\Users\DELL\Documents\GitHub\quiz-app> pytest --cov
===== test session starts =====
platform win32 -- Python 3.13.7, pytest-9.0.1, pluggy-1.6.0 -- C:\Users\DELL\AppData\Local\Programs\Python\Python313\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\DELL\Documents\GitHub\quiz-app
configfile: pyproject.toml
testpaths: tests
plugins: aioio-4.11.0, cov-7.0.0
collected 12 items

tests/test_integration.py::TestQuizIntegration::test_complete_quiz_flow PASSED
tests/test_question.py::TestQuestion::test_create_multiple_choice_question PASSED
tests/test_question.py::TestQuestion::test_check_correct_answer PASSED
tests/test_question.py::TestQuestion::test_question_requires_text PASSED
tests/test_quiz.py::TestQuiz::test_create_empty_quiz PASSED
tests/test_quiz.py::TestQuiz::test_add_question_to_quiz PASSED
tests/test_quiz.py::TestQuiz::test_cannot_add_duplicate_questions PASSED
tests/test_quiz.py::TestQuiz::test_get_question_by_index PASSED
tests/test_quiz.py::TestQuiz::test_submit_answer_and_track_score PASSED
tests/test_result.py::TestQuizResult::test_perfect_score PASSED
tests/test_result.py::TestQuizResult::test_passing_grade PASSED
tests/test_result.py::TestQuizResult::test_result_summary PASSED
[  8%]
[ 16%]
[ 25%]
[ 33%]
[ 41%]
[ 50%]
[ 58%]
[ 66%]
[ 75%]
[ 83%]
[ 91%]
[100%]

===== tests coverage =====
coverage: platform win32, python 3.13.7-final-0

Name     Stmts  Miss  Cover  Missing
src\__init__.py    4      0   100%
src\question.py   15      2   87%  19, 26
src\quiz.py       22      0   100%
src\result.py     11      0   100%
TOTAL        52      2   96%
Coverage HTML written to dir htmlcov
===== 12 passed in 0.39s =====
```

- Link i GitHub-it <https://github.com/Geti23/quiz-app>

- Raporti i CI/CD ose code coverage (në faqen tjetër):

CI pipeline ka rezultuar me sukses në të gjitha fazat e saj:

- Kontrollimi i kalueshmërisë së testeve në:
 - Tri sistemet operative Linux, Windows, MacOS;
 - Katër versione të python 3.9, 3.10, 3.11, 3.12.
- Kontrollimi i formatimit të kodit duke përdorur 'black',
- Kontrollimi i stilit të kodit duke përdorur 'flake8',
- Kontrollimi i tipeve të përdorura në metoda (Type checking) duke përdorur 'mypy'.

Code implemented + passing tests #3

Triggered via push 1 minute ago

Status: Success Total duration: 59s Artifacts: -

Jobs:

- test (ubuntu-latest, 3.9)
- test (ubuntu-latest, 3.10)
- test (ubuntu-latest, 3.11)
- test (ubuntu-latest, 3.12)
- test (windows-latest, 3.9)
- test (windows-latest, 3.10)
- test (windows-latest, 3.11)
- test (windows-latest, 3.12)
- test (macos-latest, 3.9)
- test (macos-latest, 3.10)
- test (macos-latest, 3.11)
- test (macos-latest, 3.12)
- lint

main.yml
on: push

lint 11s

Matrix test

12 jobs completed Show all jobs

File	function	statements	missing	excluded	coverage
src__init__.py	(no function)	4	0	0	100%
src\question.py	Question.__init__	5	0	0	100%
src\question.py	Question.check_answer	1	0	0	100%
src\question.py	Question.__eq__	3	1	0	67%
src\question.py	Question.__hash__	1	1	0	0%
src\question.py	(no function)	5	0	0	100%
src\quiz.py	Quiz.__init__	3	0	0	100%
src\quiz.py	Quiz.add_question	2	0	0	100%
src\quiz.py	Quiz.get_question	1	0	0	100%
src\quiz.py	Quiz.submit_answer	1	0	0	100%
src\quiz.py	Quiz.get_result	8	0	0	100%
src\quiz.py	(no function)	7	0	0	100%
src\result.py	QuizResult.__init__	3	0	0	100%
src\result.py	QuizResult.is_perfect	1	0	0	100%
src\result.py	QuizResult.is_passing	1	0	0	100%
src\result.py	QuizResult.get_summary	1	0	0	100%
src\result.py	(no function)	5	0	0	100%
Total		52	2	0	96%