МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА



**АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ**

**КОМП’ЮТЕРНИХ СИСТЕМ**

Завдання 5: «Тестування гри Tic Tac Toe»

***Виконав:***

ст. гр. КІ-404

Зелений Т.Р.

***Прийняв****:*

Федак П.Р.

Львів – 2024

**ЗАВДАННЯ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Варіант** | **Ім’я студента** | **Група** | **Гра** | **Config формат** |
| **7** | Зелений Тарас Романович | КІ-404 | tik-tac-toe 3x3 | XML |

1. Впровадити або використати існуючий фреймворк тестування;

2. Створити набір автоматизованих тестів;

3. Тестовий звіт повинен містити кількість всіх тестів, пройдених тестів, не пройдених тестів,

покриття;

4. Покриття має бути більше 80%.

**ТЕОРЕТИЧНИЙ МАТЕРІАЛ**

**Автоматизоване тестування** — це процес використання спеціальних інструментів та програмного забезпечення для автоматичного виконання тестових сценаріїв з метою перевірки функціональності, продуктивності, безпеки чи інших аспектів роботи програмного забезпечення. На відміну від мануального тестування, де тестувальник вручну виконує тести, автоматизація дозволяє заощадити час, зменшити людський фактор і підвищити ефективність тестування.

**Юніт-тести (Unit Tests)** — це вид тестування, який перевіряє окремі модулі або компоненти програмного забезпечення в ізоляції. Кожен тест зосереджується на одній конкретній функції, методі або класі, щоб перевірити, чи працює він належним чином.

**Основні аспекти юніт-тестів:**

1. Цілі юніт-тестування:

* Переконатися, що окремі частини коду працюють правильно.
* Швидко виявляти помилки на ранніх етапах розробки.
* Забезпечити основу для подальшої інтеграції коду.

1. Особливості:

* Виконуються ізольовано, без взаємодії з іншими компонентами системи.
* Можуть використовувати заглушки (stubs) або моки (mocks) для імітації залежностей.
* Написання тестів зазвичай автоматизоване, що дозволяє швидко перевіряти їх багаторазово.

1. Обмеження:

* Юніт-тести перевіряють лише ізольовані частини системи й не враховують інтеграцію компонентів.
* Не гарантують, що вся система працюватиме правильно, навіть якщо всі юніт-тести успішно пройдені.

**ВИКОНАННЯ РОБОТИ**

**ЗАПУСК СЕРВЕРНОЇ СТОРОНИ(HW)**

1. Клонуйте репо за допомогою наступної команди нижче:

***Команда:*** *https://github.com/Taras-Zelenyy/csad2425ki404zelenyytr07.git*

1. Відкрийте git bash
2. Перейдіть до feature/develop/task5. Використовуйте наступну команду:

***Команда:*** git checkout feature/develop/task5

1. Знайдіть наступний файл за наступним шляхом: your\_path\server\server.ino
2. Відкрийте Arduino IDE, виберіть порт (у мене це COM3), плату та завантажте код.

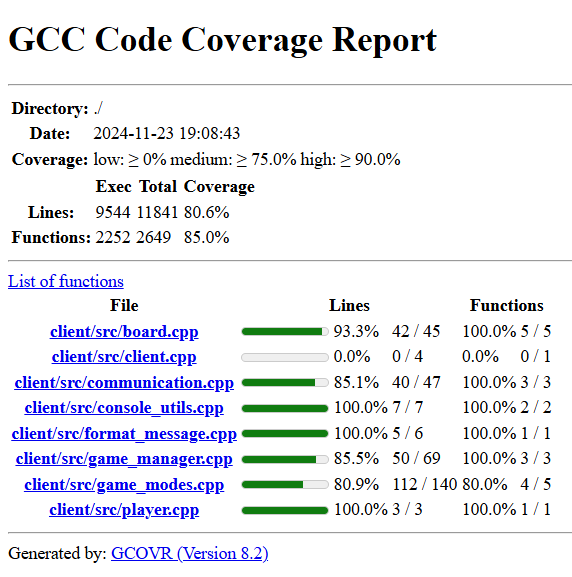
**ЗАПУСК КЛІЄНТСЬКОЇ СТОРОНИ(SW)**

1. Відкрийте pull feature/develop/task5
2. Перейдіть на вкладку Action
3. Виберіть останню збірку проекту
4. Завантажте артефакти
5. Розархівуйте завантажену папку
6. Перейдіть до «your\_path\build-artifacts\Debug\client.exe»
7. Двічі клацніть на client.exe

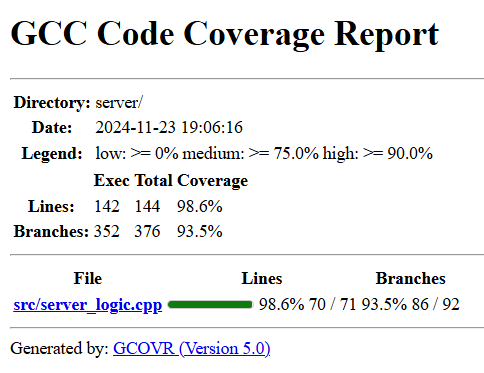
**ДОКУМЕНТАЦІЯ**

Репорти після тестування ви можете знайти в розділі GitHub Action для останнього пройденого білда. Там буде дві папки з артефактами (coverage-report та server-coverage-report) де знаходиться уся детальна інформація про тести та їх статус. Ось скрін який зображає відсоток покриття тестами клієнта і сервера відповідно:

***Відсоток покриття тестами клієнта:***



***Відсоток покриття тестами сервера:***



**ВИСНОВОК**

У ході виконання лабораторної роботи були успішно реалізовані всі поставлені завдання:

* Впроваджено та використано фреймворк Google Test для клієнтської та серверної частини гри Tic Tac Toe, що забезпечило стандартизоване середовище для написання та виконання тестів.
* Створено набір автоматизованих тестів для перевірки функціональності клієнта і сервера, що дозволило автоматично оцінити коректність роботи програми.
* Згенеровано тестовий звіт, який містить інформацію про загальну кількість тестів, кількість успішно пройдених та провалених тестів, а також рівень покриття. Звіти доступні у артефактах GitHub Actions.
* Забезпечено покриття коду тестами понад 80%, що відповідає вимогам і свідчить про високий рівень перевірки функціональності системи.

**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Шпіцер А.С. Instructions for practical tasks and coursework from «Computer systems automated design» – методичка. НУ «Львівська Політехніка», 12 с.