РІВНЕНСЬКИЙ ДЕРЖАВНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ

Факультет математики та інформатики

Кафедра цифрових технологій та методики навчання інформатики

«До захисту допущено»
Завідувач кафедри ЦТ та МНІ
_____ Наталія ПАВЛОВА
«___»_____2025 р.
протокол №__

ВАДИМ ОЛЬХОВИК

КВАЛІФІКАЦІЙНА РОБОТА

за освітнім ступенем «магістр»
на тему:

МЕТОДИКА ЗАСТОСУВАННЯ GIT CLASSROOM ДЛЯ НАВЧАННЯ ПРОГРАМУВАННЮ У ЗАКЛАДАХ ЗАГАЛЬНОЇ СЕРЕДНЬОЇ ОСВІТИ

Виконав:
здобувач ІІ курсу
групи М-І-21
спеціальності 014
«Середня освіта
(Інформатика)»
Вадим ОЛЬХОВИК
Науковий керівник:
доктор педагогічних
наук, професор
кафедри цифрових
технологій та методики
навчання інформатики
Наталія ПАВЛОВА

2025

АНОТАЦІЯ

Ольховик В.Л. Методика застосування Git Classroom для навчання програмуванню у закладах загальної середньої освіти. − Кваліфікаційна робота на здобуття ступеня «Магістр» за спеціальністю 014 Середня освіта (Інформатика) – Рівненський державний гуманітарний університет – Рівне, 2025. – 65с.

У магістерській роботі проаналізовано сучасні підходи та інструменти для навчання програмуванню в старшій школі. Зокрема, розглянуто доцільність, переваги та недоліки впровадження GitHub Classroom в освітній процес з огляду на зміст та обсяг шкільної програми з предмету інформатики в 10-11 класах.

Аналіз актуальних методів викладання інформатики, дозволив нам сформувати висновок про необхідність та освітню цінність використання систем контролю версій при вивченні тем, що стосуються програмування. Також, було обрано систему управління навчанням, котра б дозволила провести експеримент.

У другому розділі проведено аналіз функціоналу GitHub Classroom, та розроблено методичні рекомендації щодо використання системи на уроках інформатики. Наведено приклади завдань, які демонструють основні концепти системи контролю версій Git, та дозволяють користуватися нею на базовому рівні.

Результати експериментальної частини роботи підтвердили зростання мотивації учнів 11 класу, було помічено позитивний вплив використання професійних інструментів для подальшої успішної профорієнтації учні.

Ключові слова: система управління навчанням, старша школа, системи контролю версій, інформатика, програмування, Git, GitHub Classroom.

ABSTRACT

Olkhovyk V.L. The Methodology of Applying GitHub Classroom for Teaching Programming in General Secondary Education Institutions. – Qualification work for the Master degree of specialty 014 Secondary education (Informatics) – Rivne State University of Humanities – Rivne, 2025. – 65 p.

The Master's thesis analyzes modern approaches and tools for teaching programming in upper secondary school. In particular, the feasibility, advantages, and disadvantages of implementing GitHub Classroom into the educational process are examined, taking into account the content and scope of the secondary school computer science curriculum for grades 10-11.

The analysis of current methods of teaching computer science allowed us to conclude that the use of version control systems is necessary and educationally valuable when studying programming-related topics. In addition, a learning management system was selected that enabled the implementation of an experimental study.

The second chapter analyzes the functionality of GitHub Classroom and develops methodological recommendations for its use in computer science lessons. Examples of tasks are provided that demonstrate the basic concepts of the Git version control system and allow students to use it at a basic level.

The results of the experimental part of the study confirmed an increase in the motivation of 11th-grade students. A positive impact of using professional tools on students' future successful career guidance was also observed.

Keywords: learning management system, high school, version control systems, Informatics, programming, Git, GitHub Classroom.

ЗМІСТ

ВСТУП

Актуальність дослідження: світ динамічно змінюється, і разом з цим зростає перелік компетенцій, котрими має оволодіти молода людина, щоб бути конкурентною. Особливо це помітно в наш час, коли робота може бути дистанційною, а компанії створюють багатонаціональні команди, аби мати доступ до найкращих кадрів. Тому, одним із ключових положень сучасної освітньої політики України є формування в учнів старшої школи комплексу ключових компетентностей, необхідних для життя в умовах цифрової економіки. Відповідно до статті 12 Закону України «Про освіту», повна загальна середня освіта має забезпечувати готовність особистості до «свідомого життєвого вибору та самореалізації, відповідальності, трудової діяльності та громадянської активності». А серед ключових компетентностей які необхідні для досягнення таких цілей згадуються «інноваційність; «інформаційно-комунікаційна компетентність» та «здатність спілкуватися рідною (у разі відмінності від державної) та іноземними мовами» (закон України «Про освіту» від 31.10.2025, ст. 12).

Також, на значні зміни в середній освіті, вказує презентований стратегічний план діяльності МОН до 2027 року, згідно з яким, в Україні впроваджується профільне навчання в старшій школі вже з 1 вересня 2027 року (згідно постанови КМУ від 25 липня 2024 р. № 851). З цієї дати набуває чинності Державний стандарт профільної середньої освіти, де окремо виділено інформатичну галузь. В цьому документі перелічено обов'язкові результати навчання у галузі, серед яких є те, що здобувач освіти «створює інформаційні продукти та програми для ефективного розв'язання задач/проблем, творчого самовираження індивідуально і у співпраці з іншими особами за допомогою цифрових пристроїв чи без них» («Державний стандарт профільної середньої освіти» від 25 липня 2024 р., п.19).

У контексті досліджуваної теми, ці документи є базовими, оскільки використання систем контролю версій (СКВ), зокрема Git у поєднанні з GitHub, передбачає роботу з цифровими інструментами, хмарними сервісами, алгоритмами співпраці та командній роботі, а також використання англійської мови у спілкуванні. Таким чином, інтеграція Git в освітній процес безпосередньо сприяє реалізації компетентнісного підходу та профільній орієнтації особистості.

Мета дослідження: визначити доцільність, переваги та недоліки використання в освітньому процесі старшої школи платформи GitHub Classroom з метою вивчення програмування.

Об'єкт дослідження: методи навчання програмуванню учнів старших класів у закладах загальної середньої освіти.

Предмет дослідження: інструменти та технології організації навчальної діяльності з вивчення програмування з використанням GitHub Classroom.

Завдання дослідження:

1. Проаналізувати сучасні підходи та інструменти для навчання програмуванню у старшій школі;
2. Дослідити функціональні можливості GitHub Classroom та його API;
3. Визначити педагогічні переваги та обмеження використання GitHub Classroom у шкільному курсі програмування;
4. Розробити методику використання GitHub Classroom у навчанні програмуванню старшокласників;
5. Оцінити ефективність запропонованої методики на основі результатів експерименту, шляхом анкетування учнів;
6. Сформулювати рекомендації щодо використання GitHub Classroom на уроках інформатики старшої школи;

Методи дослідження:

− аналіз науково-методичної літератури з методики навчання інформатики та програмування, використання цифрових освітніх інструментів;

− порівняльний аналіз сучасних платформ та сервісів для навчання програмуванню;

− анкетування та опитування учнів з метою оцінювання зручності, мотивації та ефективності використання Github Classroom;

− узагальнення й систематизація теоретичних відомостей, виокремлення головного у формі висновку.

Практична значущість дослідження: розроблені рекомендації та алгоритми щодо побудови ефективної навчальної діяльності з використання платформи GitHub Classroom, а саме алгоритми видачі завдань, впровадження автоматизованого тестування коду, взаємодії вчителя та учня через середовище розробки програмного забезпечення GitHub.

Апробація результатів: Основні положення і результати дослідження доповідалися та обговорювалися на:

− IV Всеукраїнській науково-практичній конференції «Підготовка педагогів до професійної діяльності в умовах змішаного навчання» (14 травня 2025 року, м. Рівне).

− XVIII Всеукраїнській науково-практичній конференції «Інформаційні технології у професійній діяльності» (10 листопада 2025 року, м.Рівне). Структура магістерської роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи складає 65 сторінок, основний зміст представлений на 49 сторінках. Список використаних джерел налічує 41 найменування.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ТА МЕТОДИЧНІ ОСНОВИ ЗАСТОСУВАННЯ GITHUB CLASSROOM У НАВЧАННІ ПРОГРАМУВАННЮ

1.1 Огляд сучасних методів навчання програмуванню учнів старших класів

Процес навчання учнів старшої школи програмуванню має свої особливості та виклики. До прикладу, згідно досліджень , існує проблема зниження інтересу до програмування учнів старшої школи у порівнянні з 4-5 класом (Семенихіна, Руденко, 2018). Причинами називається низька самооцінка здобувачів освіти, складність навчального матеріалу, відсутність інтересу й мотивації.

Щоб покращити мотивацію учнів та рівень засвоєння навичок з програмування наразі існує багато сучасних методик навчання , серед яких:

- компетентнісно-орієнтований підхід;
- проектне навчання (Project-based learning);
- перевернутий клас (Flipped classroom);
- змішане навчання (Blended learning);
- дослідницьке й проблемне навчання;
- колаборативне та парне навчання;
- гейміфікація та ігрові технології.

Перераховані методи не є взаємовиключними, їх можна і потрібно використовувати поєднуючи один з одним. Щоб з’ясувати, як краще їх застосовувати, оглянемо дані методики детальніше з прикладами.

Компетентнісно-орієнтований підхід. Такий підхід у навчанні спрямований не стільки на передачу знань, скільки на формування в учнів уміння ефективно застосовувати їх у реальних життєвих та навчальних ситуаціях (Горошкіна, 2022, с.89). Основний акцент робиться на розвитку ключових і предметних компетентностей, серед яких критичне мислення, здатність до

командної взаємодії, творчого розв'язання проблем, використання цифрових інструментів та здійснення самооцінювання власного поступу. Організація навчання в межах цього підходу базується на практично орієнтованих завданнях, міждисциплінарній інтеграції, дослідницькій та проєктній діяльності, що дає можливість учням здобувати досвід, наближений до реальних умов. **Приклад завдання:** створити калькулятор комунальних витрат сім'ї, чи калькулятор, який дозволяє розрахувати калорійність страви, пропорції інгредієнтів, тощо.

Проєктне навчання. В ході створення проєкту учні вирішують певну практичну проблему, або роблять дослідження, аналізуючи та синтезуючи інформацію, що подана у різних форматах. Написання проєекту супроводжується етапами − організацією проєкту (вибір теми), плануванням, реалізацією (пошукова діяльність) та презентацією результатів (Млавець , 2021, с.40). У рамках одного проєкту виникає багато завдань, що дає можливість застосувати роботу у групах , де кожен буде відповідати за свій напрямок (Сікора, 2022, с.287). Сікора робить висновок, що проєктна діяльність підвищує мотивацію та залученість учнів. **Приклад завдання:** проєкт сайту-вікторини з обраного предмету написаного за допомогою CSS/HTML. Учнів згруповано у робочі команди, де кожен учасник працює над своїм завданням, до прикладу: складання питань вікторини, написання структури HTML сторінки, написання форм із запитаннями, написання CSS стилів, створення скрипта з обробкою відповідей та підрахунком балів з використанням умовних операторів if. Під час виконання такого комплексного завдання, здобувачам освіти знадобиться вміння використовувати системи контроля версій Git та веб ресурсів онлайн репозиторіїв, таких як Github.

Перевернутий клас і змішане навчання. Технологія змішаного навчання передбачає винесення частини уроків з очного формату у дистанційну форму з використанням ІКТ, LMS систем, соціальних мереж, відеохостингів. Одними з перших хто впровадив таку методику на практиці були Дж. Бергман (Jonathan Bergmann) та А. Самс (Aaron Sams), які успішно випробували його 2007 році у

Вудлендській школі в штаті Колорадо (Бергман, Самс, 2012). Відтоді викладачі закладів освіти почали записувати лекції на відео та поширювати їх серед студентів в рамках освітньогопроцесу. Широкого застосування змішане навчання набуло під час епідемії COVID-19 та широкомасштабного нападу на Україну в 2022-му році, через об'єктивні причини.

Одним із видів змішаного навчання є методика **перевернутого класу (flipped classroom)**. Вона передбачає винесення опрацювання теоретичних матеріалів на роботу вдома, а у класі − виконання практичних завдань. Це дозволяє учневі по-перше, опрацювати матеріал у зручний час, а по-друге отримати корисні поради та вказівки з виконання лабораторних робіт у класі. Особливо корисним це є під час вивчення програмування, через низку можливих помилок у новачків, які потребують поради в реальному часі. При цьому вчитель виступає у ролі ментора-наставника, а не джерела інформації (Бобровський, 2016, с.3-6). В табл. 1.1 наведено основні переваги та недоліки змішаного навчання.

Таблиця 1.1

Аспект	Перевага	Недолік
Учень	Навчання у власному темпі, індивідуальна допомога, розвиток навичок цифрової грамотності, самостійність та відповідальність	Вимагає високої самодисципліни та доступу до технологій, ризик соціальної ізоляції, потреба в надійному Інтернет з'єднанні
Учитель	Більше часу для активної взаємодії, фокус на практиці, використання онлайн платформ для тестування та оцінювання, легкість	Високі витрати часу на створення контенту, зміна методології, необхідність освоювати нові технології, зростання вимог до рівня кваліфікації вчителя
Контент	Легкість надолуження матеріалу, інтеграція цифрових ресурсів в LMS, доступність навчального матеріалу в будь-який час доби.	Потреба в регулярному оновленні відео-лекцій та іншого цифрового контенту, ризик фрагментарного засвоєння матеріалу.

Гейміфікація та ігрові технології. Цей метод передбачає інтеграцію таких ігрових елементів як системи балів, значки (badges), таблиці лідерів та

рівні складності в процес навчання. У контексті програмування, гейміфікація перетворює теоретичні завдання на практичні місії або квести, де швидкість, якість та ефективність розв'язання задач винагороджуються віртуальними досягненнями. Завдяки миттєвому зворотному зв'язку та відчуттю прогресу (перехід на новий рівень), гейміфікація стимулює здорову конкуренцію, сприяє формуванню алгоритмічного мислення та заохочує учнів до самостійного й ітеративного вдосконалення своїх навичок програмування.

Для реалізації підходу доцільно використовувати різні онлайн платформи такі, як вікторини: Kahoot; Quiziz; Quizlet; Wordwall, Blooket; або платформи-симулятори: Code.org; Tynker; Tinkercad Circuits; Robocode;Kodable; BlocklyGames; CheckiO; CodeMonkey; CodeCombat; Vim Adventures (Тітова, 2024, с.106).

Можна використовувати платформу **codewars**, де зручно повправлятися в програмуванні C++, Python, та ще більше 50-ти мов програмування**.** Завдання поділені на рівні, котрі відкриваються після виконання відповідної складності. На платформі є онлайн редактор коду, поруч з яким описується суть завдання (рис. 1.1). В **codewars** можна отримувати два типи досягнень:

− рівень kyu («Кю» з японської - рівень рангу знань, від 10 до 1 у порядку зростання умінь), який можна підвищувати виконуючи складніші завдання;

− бали «Честі», які можна отримати, якщо запропонувати власне завдання, знайти помилку в іншому завданні, або записати в іншому середовищі.

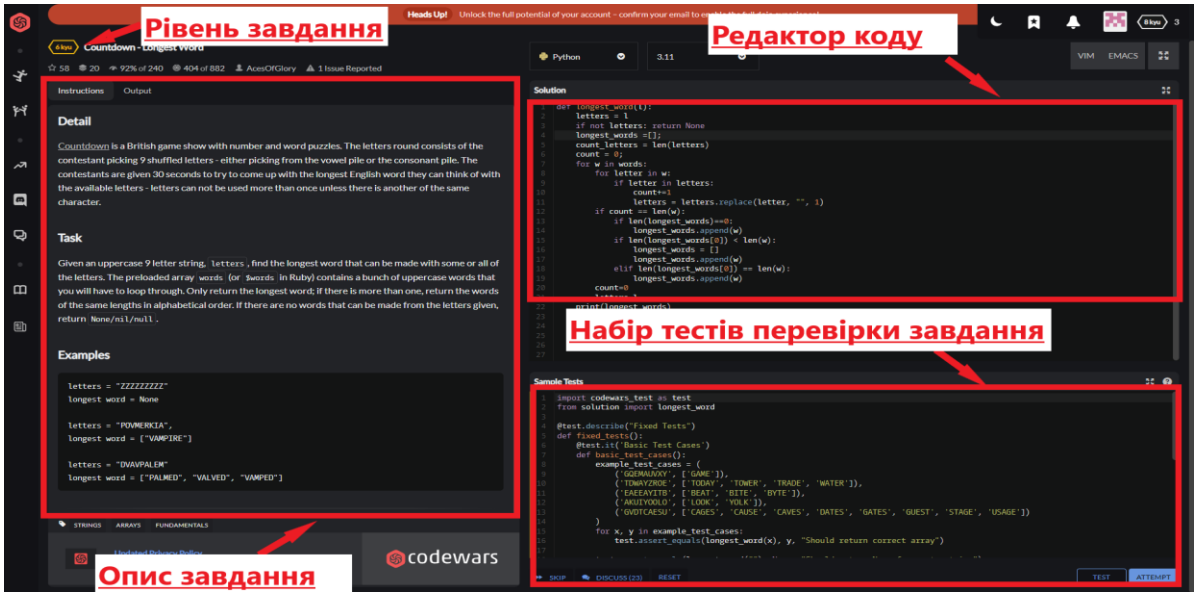


Рис. 1.1. Вигляд інтерфейсу **codewars** в режимі виконання завдання

Подібні платформи дають здобувачам освіти відчути себе цінним учасником спільноти, позмагатися між собою у створенні найбільш оптимізованого коду (тест виводить час роботи програми). Вчитель має можливість підібрати різнорівневі завдання за різними темами. Серед завдань є багато логічних ігор, що розроблені програмістами з різних країн. Перевагою є і оптимізація часу, адже у кожна ката містить вбудовані тести, тому вчитель може надати додаткові пояснення легше зробити підказку як виправити код (Тимощук, 2024, с.330).

Окрім готових онлайн ресурсів, в рамках гейміфікації навчального процесу, можна запропонувати учням написати свою гру. Таке завдання має бути виключно добровільним до виконання, бо має на меті підвищити мотивацію, зацікавленість у предметі та збільшення здорової конкуренції між учнями. Щоб здобувачі освіти були впевненими у своїх силах, їх потрібно плавно підводити до певного рівня, надаючи методички з розробки міні ігор (наприклад [https://ua.izzi.digital/DOS/1183864/1410274.html])

Звичайно, разом з великою кількістю переваг впровадження ігрових елементів у навчання, існує можливість перетворення такого процесу цілком на

гру, що не відповідатиме сформульованій освітній меті, тому їх варто використовувати виважено й доцільно (Медведєва , 2024, с.136).

1.2 Можливості використання систем контролю версій в контексті модельної програми з інформатики

Перші системи контролю версій (СКВ) з'явилися в 80-х роках минулого століття. Це СКВ з назвою RCS, котра розроблена в 1982 році та постачалася з ядром Linux. Такі системи слугують для збереження версій файлів та поділяються на три типи:

− локальні СКВ (копія файлів проєкту на одному локальному комп'ютері);

− централізовані (копія файлів на сервері);

− розподілені (копія файлів на сервері і на локальних комп'ютерах).

Незважаючи на те, що локальні СКВ забезпечують основну функцію збереження версій проєкту, вони не дозволяють працювати над одним проєктом команді розробників. Це вимагає створення резервних копій, щоб не втратити проєкт який буде в одному екземплярі на локальному комп'ютері. Саме таку архітектуру мала RCS як одна з перших СКВ.

Централізовані СКВ мають в своїй архітектурі головний сервер, який відповідає за збереження та надання потрібних версій проєкту. Це вже краще ніж у локальній версії, але також має свої недоліки. Так, якщо сервер буде недоступний (офлайн, або вимкнеться), робота над проєктом зупиниться також і для всіх його учасників. Безпека збереження історії файлів також залежить від резервного копіювання, оскільки на комп'ютерах учасників проєкту буде зберігатися лише його поточна версія.

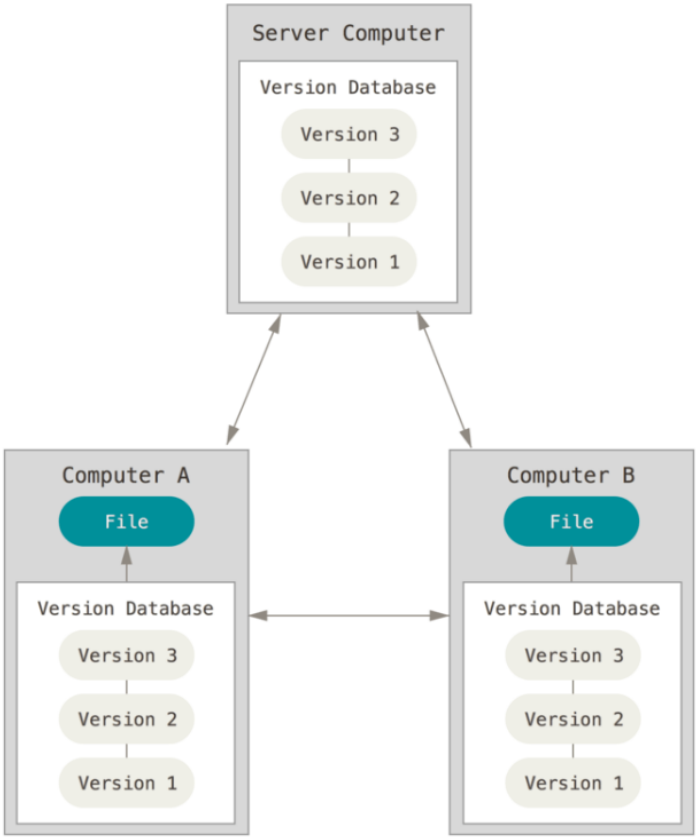


Рис. 1.2. Схема роботи розподіленої СКВ

Саме розподілена архітектура (рис 1.2.) лежить в основі таких популярних СКВ як **Git** та Mercurial. Розроблена Л. Торвальдсом (розробником ядра Linux) у 2005 році, **Git** є системою керування версіями, яку активно використовують професіонали. Її перевагами є сумісність з різними ОС, інтеграція з IDE та безкоштовність (ліцензія GNU GPL v2). Основними сценаріями використання СКВ є:

– **командна розробка :** кілька людей працюють над одним проектом, і потрібно об'єднувати їхні зміни, та уникнути конфліктів;

– **відстеження історії :** коли потрібно знати, хто, коли, і чому вніс певну зміну;

– **повернення до попереднього стану:** коли є необхідність скасувати помилкові зміни, або повернутися до стабільної версії (релізу);

– **розгалуження та експерименти:** можливість створювати ізольовані гілки для розробки нових функцій, виправлення помилок або експериментів, не впливаючи на основну робочу версію;

– **розгортання та релізи:** підготовка до хостингу сайтів, розгортання програм в різних середовищах.Тому доцільно відстежувати файли вихідного коду програм, конфігураційних файлів та текстової документації, текстові формати такі як CSV, файли векторної графіки SVG.

Проте, використання СКВ для великих бінарних файлів, таких як, відео та аудіофайли, великих зображень, баз даних не є виправданим, адже СКВ зберігає кожну змінену версію файлу в директорії. Саме тому потрібно визначити модулі та теми з програми Інформатики 10-11 класу, котрі підходять для впровадження використання СКВ.

Чинна модельна навчальна програма з інформатики для 10-11 класів затверджена Наказом Міністерства освіти і науки № 1407 від 23 жовтня 2017 року, передбачає версію для рівня стандарт та профільного рівня. Аналізуючи ці навчальні програми, можна зазначити, що рівень стандарту складається з базового модуля (35 годин) та варіативної складової (105 годин), котру обирає вчитель спираючись на інтереси учнів та спеціалізацію закладу. На вибір пропонуються наступні модулі:

– Графічний дизайн (35 годин);
– Комп’ютерна анімація (35 годин);
– Тривимірне моделювання (35 годин);
– Математичні основи інформатики(35 годин);
– Інформаційна безпека (35 годин);
– Веб-Технології (35 годин).

Отже, модулі, котрі передбачають написання програмного коду, та при вивченні яких доцільно використовувати СКВ, це насамперед “Веб-Технології”, та “Математичні основи інформатики”, де розглядаються алгоритми шифрування та архівації. Тому, зважаючи на малу кількість годин для вивчення

інформатики, і невеликий перелік тем в яких доцільно використовувати СКВ, можна зробити висновок що вводити її в освітній процес за програмою стандарт є суперечливим рішенням.

Профільний рівень чинної модельної програми передбачає обсяг в 350 годин, що майже втричі більше ніж у стандарту, і включає в себе вивчення мови програмування, алгоритмізації, баз даних, веб-технологій, парадигм та технологій програмування (об'єктно-орієнтоване програмування). Зважаючи на широкий спектр тем, де буде розглядатися програмний код, запроваджувати використання СКВ у профільному класі є доцільним та допоможе досягти учням високого рівня навчальних досягнень.

З 2027 року, Міністерство Освіти України в рамках реформи Нової Української школи (НУШ), запровадить курс "Інформатика" лише в рамках профілізації, тому він буде існувати лише у поглибленому варіанті. Тому, для підвищення актуальності, варто проаналізувати проєект модельної навчальної програми навчального предмета «Інформатика. 10–12 клас. Поглиблений рівень». Усі теми даної програми пов'язані з програмуванням, до прикладу в 11 класі їх перелік виглядає наступним чином:

1. Прості структури даних: список, стек, черга, дек;
2. Сортування та пошук;
3. Алгоритми теорії графів;
4. Алгоритми теорії чисел;
5. Комбінаторні алгоритми;
6. Дослідження алгоритмів. Проекти;

Бачимо, що пропонується вивчати програмування ґрунтовно та систематично. Це, в свою чергу, вимагає від вчителя більшої підготовки та використання актуальних інструментів, серед яких можуть бути СКВ git, та середовище GitHub Classroom.

1.3 GitHub Classroom − функціонал, архітектура та можливості для автоматизації навчальних завдань

Перш ніж описувати, що собою являє GitHub Classroom, варто згадати що таке GitHub та конкретизувати функції цієї платформи. Вище ми згадували, що розподілена архітектура СКВ передбачає наявність серверу, який ще називають віддалений репозиторій. Щоб зменшити витрати на налаштування та обслуговування такого серверу, існують сторонні хмарні рішення такі як **GitHub, BitBucket, GitLab**.

Першою на ринку виникла платформа Github у 2008-му році (Uddin, 2023, с.30), котра дуже швидко переросла у майданчик для обміну публічним кодом, де кожен міг зробити свій внесок у проєкти з відкритим кодом, запропонувавши зміни власнику. Більшість сервісів платформи надаються безкоштовно, і лише в комерційних проєктах може виникнути потреба збільшити об'єм сховища, або кількості приватних репозиторіїв за додаткову оплату. Саме такі обмеження не влаштовували українського розробника Д.Запорожця, який в 2011-му році заснував власну компанію GitLab.

Rонкурентом GitHub є сервіс BitBucket, що започаткований в 2008-му році. Він дозволяє працювати одразу з двома найбільш популярними СКВ - Mercurial і Git. Іншою перевагою BitBucket є інтеграція з Jira - популярною програмою для відстеження проблем та постановки задач (Dmytrenko, 2023, с. 15). Також можна відмітити наявність лімітів для безкоштовного аккаунту, до прикладу кількість контрибуторів для приватного репозиторію п'ять, проти необмеженої у GitHub. З іншого боку, Bitbucket має нижчі у кілька разів ціни для корпоративних аккаунтів (Dmytrenko, 2023, с. 18).

Не зважаючи на такі широкі й цікаві можливості у конкурентів, GitHub вирізняється більшою кількістю активних користувачів та багаточисельною спільнотою, яка здатна ефективно розвивати публічні проєкти. Так, згідно щорічного звіту за 2025, на GitHub 180 мільйонів активних розробників [5], а на рис.1.3. можна побачити рейтинг популярності пошуку в гугл трьох розглянутих хмарних сховищ коду.

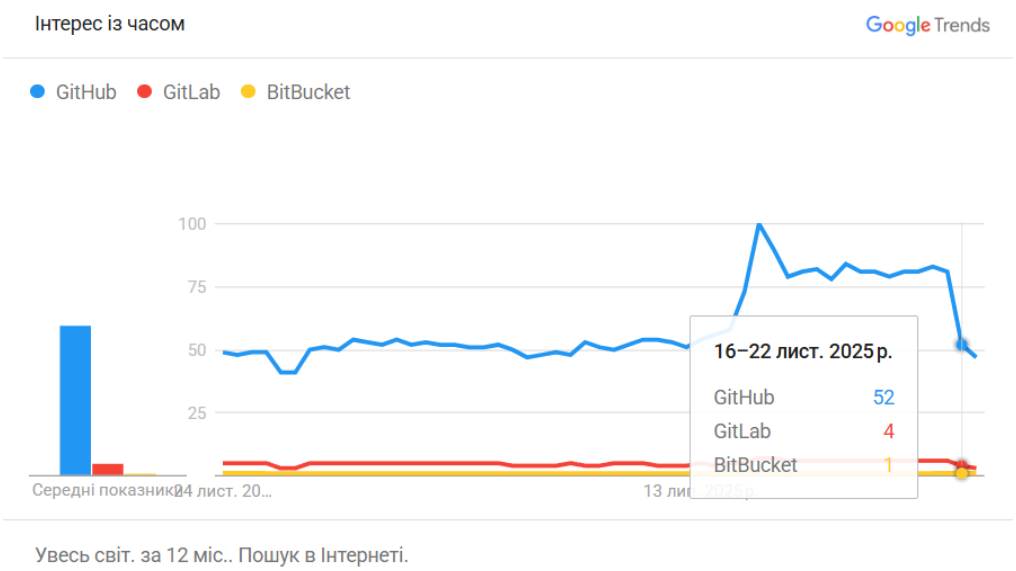


Рис. 1.3. Порівняння кількості пошукових запитів зі словами Github, GitLab, BitBucket на Google Trends
(умовна популярність, де 100 максимальний рейтинг)

Таким чином, можна робити висновок, що сукупність Git та GitHub стала галузевим стандартом у програмуванні. Це сприяло появі навчальних курсів програмування з використанням цих інструментів, на базі різних LMS (learning management system). Тому, щоб популяризувати платформу серед здобувачів освіти, в 2015 році компанія GitHub випускає свою LMS GitHub Classroom з відкритим кодом (Sprint., & Conci., 2019, с.77), що дозволила надавати завдання у вигляді репозиторіїв-заготовок, та автоматизовано перевіряти завдання.

Архітектура GitHub Classroom. Система управління навчанням GitHub Classroom не є повноцінною в тому розумінні, що без таких компонентів як Git та Github, не може в принципі використовуватись. У цьому можна пересвідчитись розглянувши блок-схему на рис. 1.4.

Вчитель спочатку створює шаблонний код завдання (template repository), потім формує assignment в GitHub Classroom, на який підписуються учні та отримують свої приватні репозиторії. Той факт, що репозиторії приватні, зменшує вірогідність плагіату, розвиває в учнів самостійність при виконанні завдань. Після того, як здобувач освіти підтвердив перші зміни, при наявності

автоматичних тестів коду, він спершу отримує відгук від системи, чи всі обов'язкові пункти завдання виконані, та може виправити помилки.

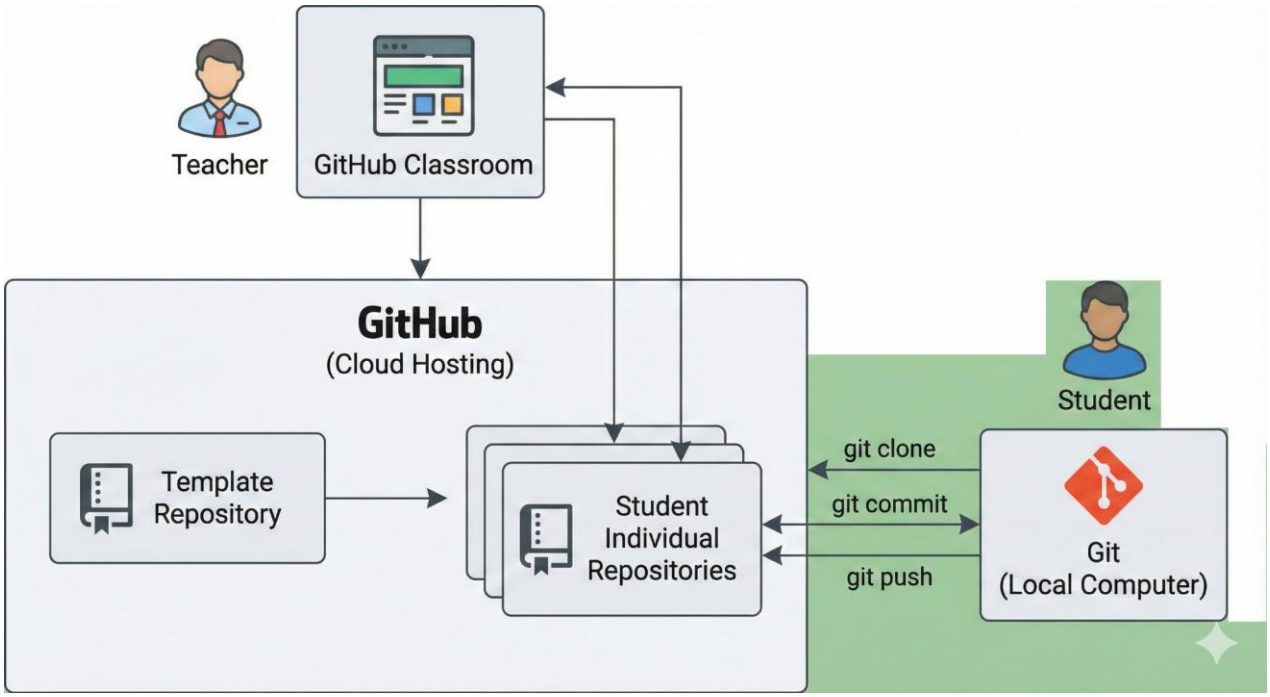


Рис.1.4. Схема взаємодії GitHub Classroom з Git та GitHub

На блок-схемі рис. 1.4 можна виділити три рівні компонентів, що забезпечують роботу Github Classroom, а саме, локальний рівень (Git), хмарний рівень (GitHub), та рівень керування (GitHub Classroom). Узагальнимо функції, що виконують ці компоненти:

1. Git (Локальний рівень):
   - **роль:** основна технологія системи контролю версій.
   - **функція:** учень використовує Git на власному локальному комп'ютері для клонування репозиторію, збереження змін (commit) та відправлення їх на сервер (push).
2. GitHub (Хмарний рівень):
   - **роль:** хмарний хостинг для репозиторіїв;
   - **функція:** зберігає віддалені репозиторії (Remote Repositories). Є центральним місцем, куди користувачі відправляють код, і звідки педагог отримує його для перевірки.

3. GitHub Classroom (Рівень керування):

- **роль:** інструмент, який автоматизує процеси видачі та збору завдань;
- **функція:** керує репозиторіями завдань, створюючи індивідуальні приватні репозиторії для кожного здобувача освіти чи /команди на GitHub.

Завдяки такій інтеграції сервісів, у вчителя що використовуватиме GitHub Classroom, відкривається доступ до інструментів автоматизації GitHub Actions (M. Meyer., 2014, с.15**),** що дозволяє створити власні сценарії перевірки завдань (функція Autograding), а також переглянути історію змін у проекті кожного окремого здобувача освіти, чи їх робочих команд (J. Loeliger, M. McCullough, 2012, с. 63). Це дозволяє на кожному етапі розробки проєкту залишити коментарі чи підказки.

Також, в GitHub Classroom вбудована інтеграція з редактором Visual Studio Code, де можна відправляти зміни коду і отримувати репозиторії з хмарного сховища GitHub, використовуючи графічний інтерфейс. Цей аспект надзвичайно важливий, адже нинішні здобувачі освіти як представники покоління альфа і зумерів, зростали, використовуючи саме програми зі зручним GUI (graphic user interface).

Таким чином, можна зробити висновок, що GitHub Classroom це комплексне рішення, що відповідає сучасним практикам програмування. Здобувачі освіти, що використовуватимуть такий інструментарій будуть не просто вчити алгоритми та синтаксис коду, а й способам взаємодії у командних проєктах: коментувати код, тестувати код, пропонувати власні зміни у спільний проєкт. Саме тому, так важливо познайомити учнів старших класів, з наближеним до реального, процесом розробки програмного забезпечення, щоб почати їх профорієнтацію раніше, та зростити мотивацію до здобуття фаху (Охріменко, 2021, с.17).

РОЗДІЛ 2.
ДОСЛІДЖЕННЯ ФУНКЦІОНАЛЬНИХ ПЕРЕВАГ, НЕДОЛІКІВ ТА МЕТОДИЧНЕ ОБҐРУНТУВАННЯ ВИКОРИСТАННЯ GITHUB CLASSROOM

2.1 Порівняльний аналіз GitHub Classroom з іншими системами управління навчанням (LMS)

Система управління навчанням(LMS) не має чітко окресленого визначення і кожен окремий екземпляр такої системи, може містити свій перелік функцій (Пригодій М., 2024, с. 3). Основними такими функціями можуть бути:

- створення навчальних курсів – відокремлених за змістом, чи за складом учасників логічної одиниці;
- управління навчальним контентом – додавання текстових документів, медіа даних в якості матеріалів курсу;
- відстеження прогресу – забезпечення постійного моніторингу результатів, прогресу опрацьованих занять, зворотній зв’язок про якість виконаних завдань;
- оцінювання – використовуються різні форми оцінювання з підрахунком балів і виставленням оцінок;
- інтерактивні функції – чати, блоги, спільна робота в середовищі, що об’єднують студентів і викладачів;
- інтеграції − можливість доєднати до використання популярну IDE, чи систему відео зв’язку для проведення конференцій, чи занять;

Сучасні події в Україні (перебої з енергопостачанням) висувають нові вимоги для LMS , наприклад, можливість працювати з нею через мобільні пристрої, а також офлайн функції.

Однією з найпопулярніших LMS є Google Classroom, завдяки десяткам інтегрованих сервісів Google для створення контенту. Для програмування онлайн можна використовувати сервіс Google Colab (Alves, & Cipriano, 2025), який має безкоштовні умови застосування. Програмний код написаний в Colab

зберігається у хмарі на Google диску, а редактор виконаний у веб версії, що дозволить писати програми, як з комп'ютера так і з мобільних пристроїв. Мобільний додаток Google Classroom здатен працювати у офлайн режимі. Таким чином, можна органічно поєднати Google Classroom та Colab. Недоліками такого поєднання можна назвати підтримку в Colab лише мови програмування Python, що не дозволить охопити усі теми шкільного курсу інформатики.

Ще однією LMS з вбудованим редактором коду, автоматизованою перевіркою виконаних завдань та можливостями командної роботи є Mimir Classroom (Loftsson., 2021). Перевагою цієї системи є закриті sandbox-и та вбудована перевірка на плагіат. Mimir є платною, тому використання у державних закладах середньої освіти є проблематичним, натомість у закладах вищої освіти може різноаспектно впроваджуватися в освітній процес.

У порівнянні з двома попередніми LMS (табл. 1.2), GitHub Classroom більше орієнтована на практичну роботу з написання проєктів. І, хоча, при створенні завдання можна зробити текстовий чи pdf файл з описом інструкцій та додатковими матеріалами, але такий медіа контент, як відео, доведеться поширювати через посилання, або ж іншу LMS. Очевидними перевагами є вбудовані інтеграції з сервісами GitHub, такими, як codespaces (онлайн IDE з інтерфейсом VS code), github Pages (дозволяє хостити створені html сторінки), github Actions (для написання гнучких тестів), а також можливість роботи офлайн, якщо завдання були попередньо завантажені. Основними недоліками можна назвати високий поріг входження, адже, потрібно знати workflow системи git, як учням так і вчителю, і вимоги до знань написання скриптів bash чи yaml, для функції автотестування. Також бракує можливості поділу завдань на змістовні модулі, але це можна обійти створенням нового Classroom.

Проведені експерименти показали, що використання GitHub Classroom у навчанні підвищує впевненість здобувачів освіти під час командної роботи (Nelson & Ponciano 2021, с.35)

Таблиця 1.2

Порівняння GitHub Classroom з іншими подібними платформами

Критерій	**GitHub Classroom**	**Google Classroom + Colab**	**Mimir Classroom**
Платформа	GitHub	Google	Amazon
Контроль версій (Git)	Вбудований	Через сторонні сервіси	Вбудований
Автоматичне створення репозитаріїв	Так	Немає	Так
Підтримка API	Потужне API	Часткова	Частково платна
Автоматичне тестування	Через CI/CD	Потрібна інтеграція	Вбудоване
Онлайн-кодування	Без компіляції (або через клієнт)	Є (тільки Python)	Є
Підтримка мов програмування	Будь-які (через Git)	Тільки Python	Java, C++, Python
Командна робота	Через Git	Через Workspace	Є
Інтерфейс для викладача	Зручний	Стандартний	Складний, функціональний
Навчальна аналітика	Через API	Через Google Sheets	Вбудована
Ціна	Безкоштовно	Безкоштовно	Платна частково

2.2 Робота з GitHub Classroom та використання вбудованих функцій

Розглянемо основні етапи роботи з GitHub Classroom з точки зору вчителя, а саме процес реєстрації, створення організації, створення завдань та тестів. Для початку роботи потрібно перейти за посиланням https://classroom.github.com, де з’явиться сторінка входу в систему (рис. 2.1). Вхід відбувається за нікнеймом та паролем з сервісу GitHub, тому при відсутності такого акаунту користуача буде перекинуто на сторінку https://github.com/signup для його створення. Мова інтерфейсу лише англійська, що можна рахувати як недоліком, так і перевагою, в залежності від підготовки вчителя та освітніх цілей. Так, в Законі України «Про застосування англійської мови в Україні», у статті 8, йдеться про те, що вивчення англійської мови є обов’язковим в закладах загальної середньої освіти, та дошкільних закладах, тому англійський інтерфейс не має стати перешкодою.

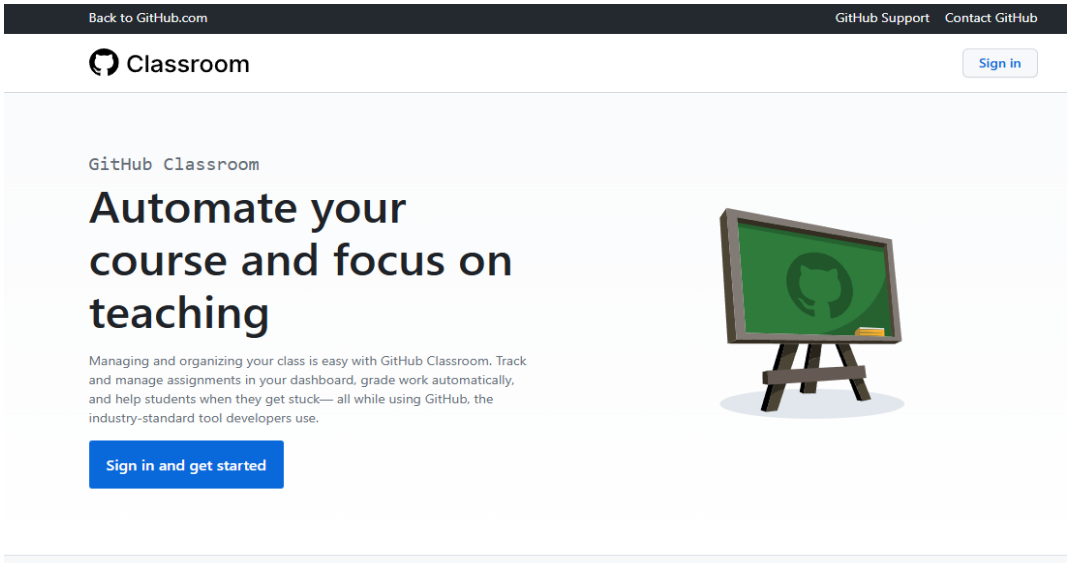


Рис 2.1. Домашня сторінка входу на платформі GitHub Classroom

Створення організації

Після реєстрації необхідно створити власну організацію, для цього на бічній панелі обрати пункт Organizations, та в новому вікні натиснути кнопку New organization (рис 2.2.). Далі відкривається форма де потрібно внести дані про назву організації, обраний тарифний план (наприклад Free), та тип власності - буде належати одному персональному акаунту, чи інституції [9].

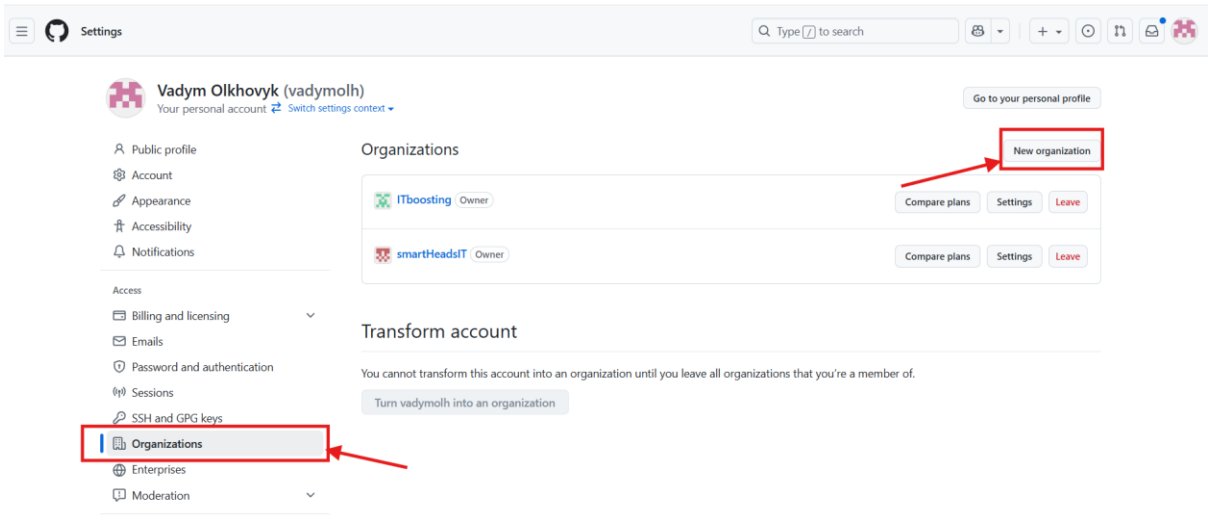


Рис 2.2. Створення організації на платформі GitHub

Наступними кроками може бути запрошення колег в організацію, якщо вчителів буде декілька, це робиться через інтерфейс email розсилання, або можна запросити їх за відомим нікнеймом у GitHub.

Створення шаблону коду

Кожній організації виділяється відокремлене сховище для репозиторіїв, і саме там логічно створювати шаблони для завдань, для чого потрібно перейти на вкладку Repositories. У формі що відкриється (Додаток Б), варто заповнити назву репозиторія латинськими літерами без пробілів, встановити видимість у режим Private, поставити галочку на пункті add README і натиснути Create Repository. Після цього, у вікні що відкриється (рис. 2.3), варто перейти на вкладку Settings та встановити прапорець Template repository. Тільки репозиторії з такою позначкою будуть видимі, при створенні завдання в Classroom.

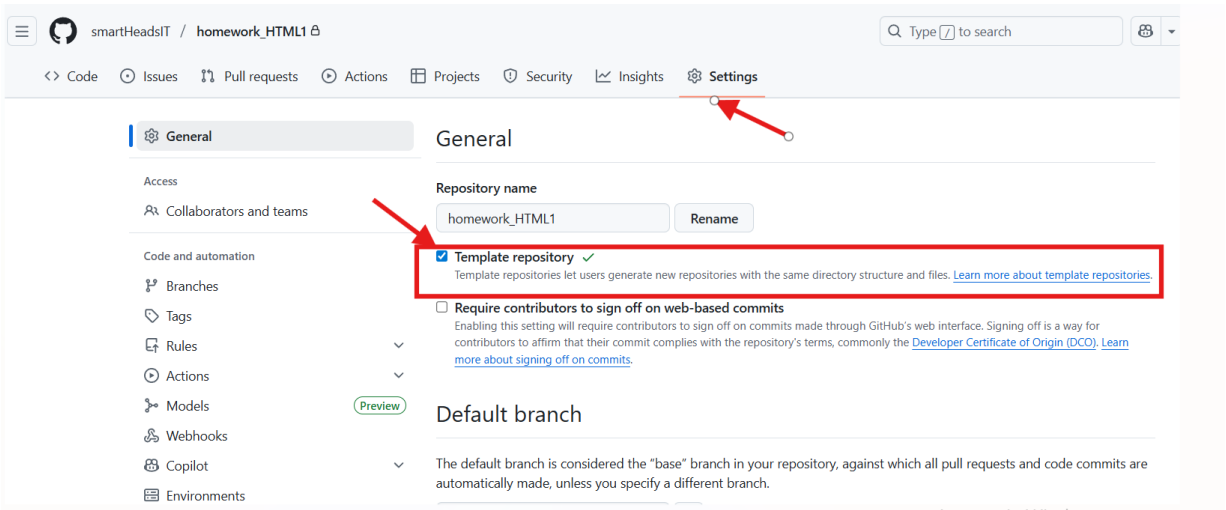


Рис 2.3. Створення шаблонного репозиторію завдання

Зазвичай шаблон передбачає низку початкових файлів та, за необхідністю, директорій. Такі файли можна додати за допомогою дії commit, або через графічний веб інтерфейс, дією Drag-&-Drop (Додаток Б), або через командну стрічку з локального комп'ютера, використавши Git клієнт.

Створення курсу

В кожній організації можна створити необхідну кількість курсів, або в термінології Classroom – кімнат. Для цього переходимо на https://classroom.github.com/, клікаємо на аватар (рис 2.4), меню «Your classrooms», та кнопку New classroom».

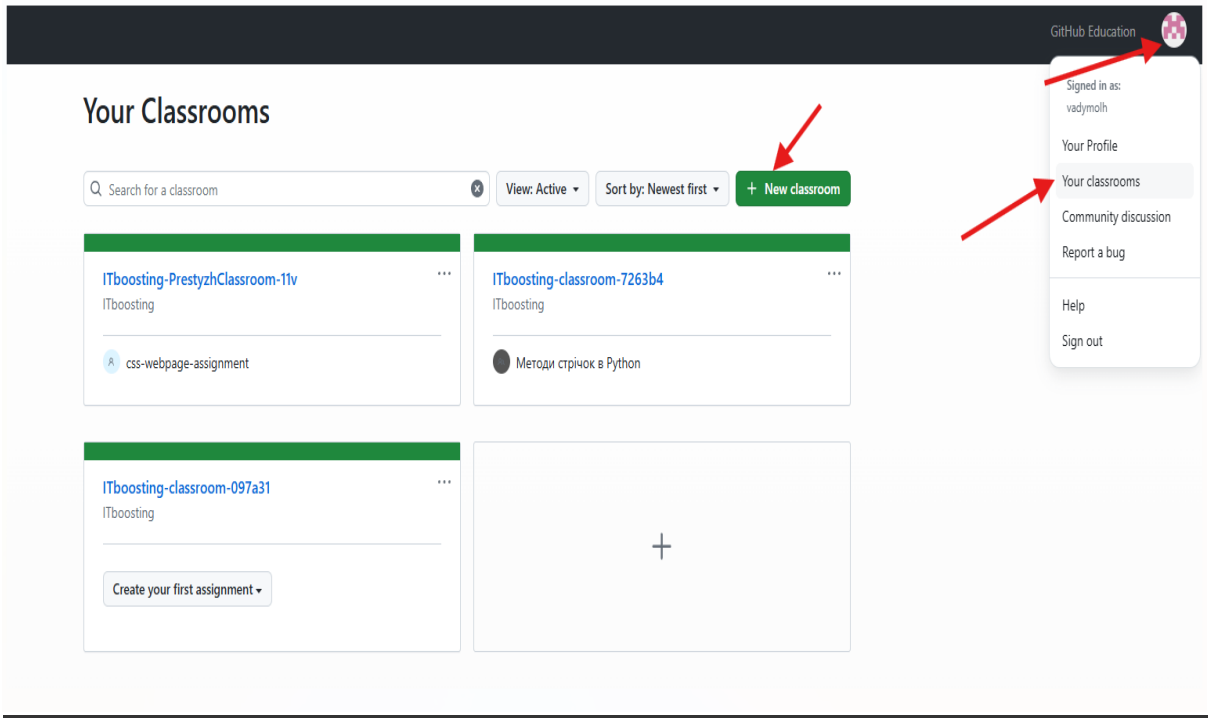


Рис. 2.4. Створення навчального курсу в GitHub Classroom

Одним із етапів створення курсу є додавання до нього учнів. В Github Classroom є одразу кілька опцій (рис. 2.5): можна додати імена вручну, або підвантажити здобувачів освіти існуючого курсу в одній з інтегрованих LMS, таких, як Google Classroom, Canvas, Moodle чи Sakai. Передбачається, що імена в списку (ростері) є унікальними, а в разі якщо будуть повні співпадіння, потрібно буде придумати інший або додатковий ідентифікатор, наприклад номер учнівського посвідчення. Для того, щоб пов'язати GitHub акаунт учня з іменем у списку групи, потрібно створити перше завдання, та поширити покликання-запрошення у чаті класу, чи в групі LMS, яку використовує клас.

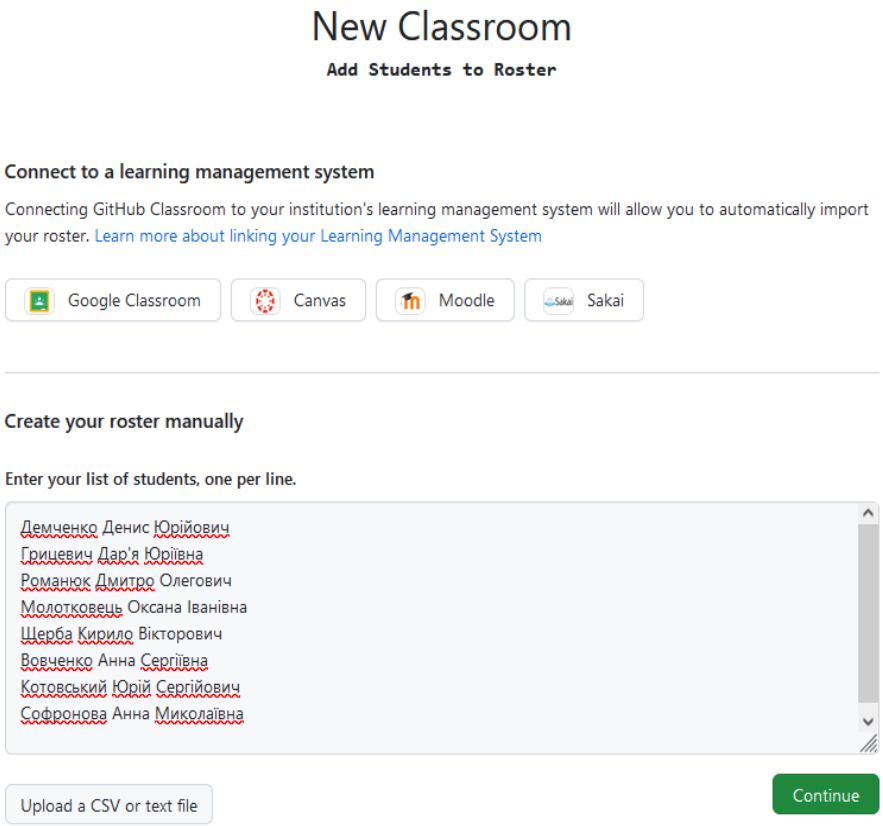


Рис. 2.5. Меню заповнення ростеру (списку) здобувачів освіти

Створення завдання

Відкривши меню курсу рис 2.6., зверху можна побачити планку з інформацією про кількість призначених завдань, кількість студентів і викладачів (вкладка TAs - від слів teaching assistants). В пункті меню «Settings», знаходиться можливість архівувати курс. Процес створення нового завдання містить три основні кроки:

1. заповнення базової інформації (назва, термін здачі, тип завдання);
2. задання початкового шаблону та середовища виконання (редактор коду);
3. визначення способу оцінювання та тестування результатів.

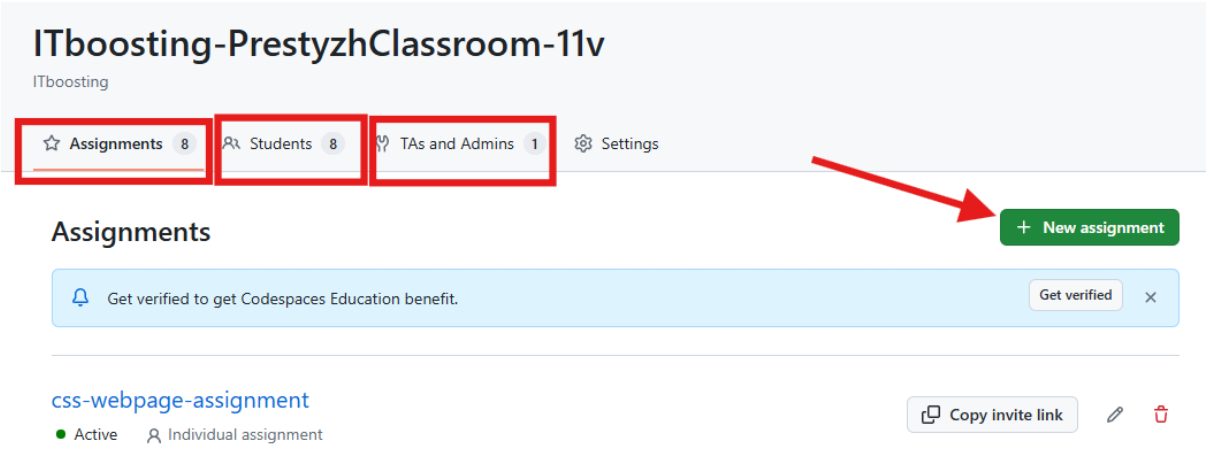


Рис. 2.6. Вигляд головної сторінки курсу в GitHub Classroom

Розглянемо ці етапи, розпочавши зі створення нового завдання кнопкою New assignment. У вікні що відкриється (рис. 2.7), потрібно заповнити назву завдання латинськими літерами, опціонально вказати псевдо для нього, крайній термін здачі та визначити, чи це будуть індивідуальні завдання, чи робота в групах.

На етапі задання шаблону, можна обрати раніше створені заготовки коду, редактор коду по замовчуванню («Додаток В»). Бажано обирати Visual Studio Code, щоб учні змогли користуватися зручним інтерфейсом та автоматичним завантаженням завдання. Серед опцій є прапорець, яким можна давати права адміністрування репозиторія, але зважаючи на недосвідченість здобувачів освіти у роботі з системою GitHub, його варто залишити не ввімкненим. Інакше, можлива втрата усього проєкту шляхом необачного видалення репозиторію.

Завершальним етапом є задання Autograding − «автоматичне виставлення оцінок». На цьому кроці додаються необхідні тести коду та умови спрацювання такого тестування. Скрипти перевірки можна задати двома різними способами − через сценарій .yaml файлу (Alves, P., & Cipriano, B., 2025, с. 9), і через заповнення форми. Так, якщо відмітити пункт «Every time a student submits an assignment» як на рис 2.8, то тест запускатиметься при кожній здачі здобувачем освіти завдання.

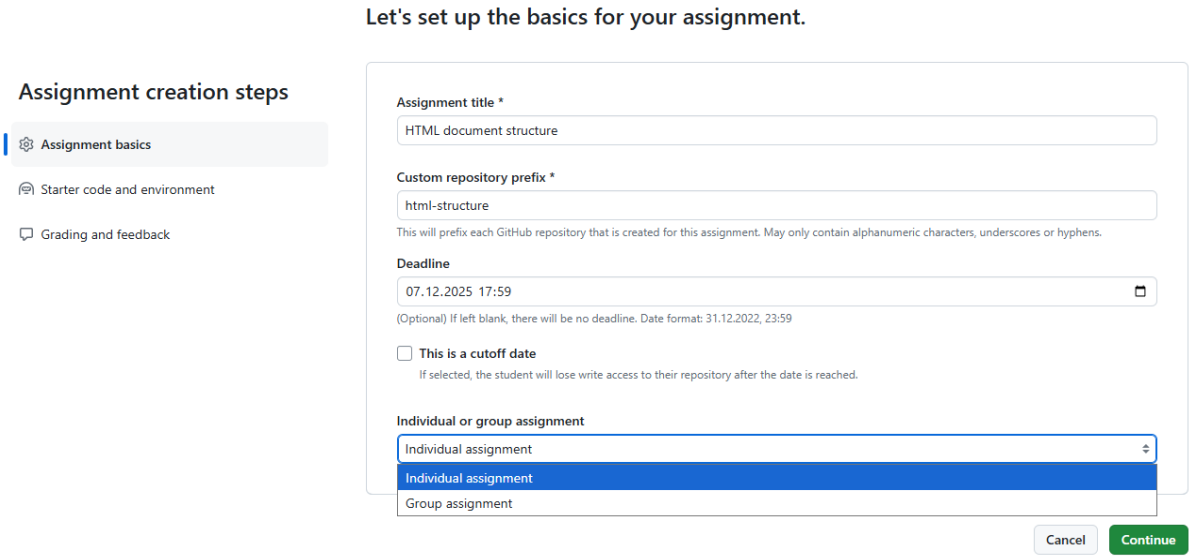


Рис 2.7 Створення нового завдання в GitHub Classroom

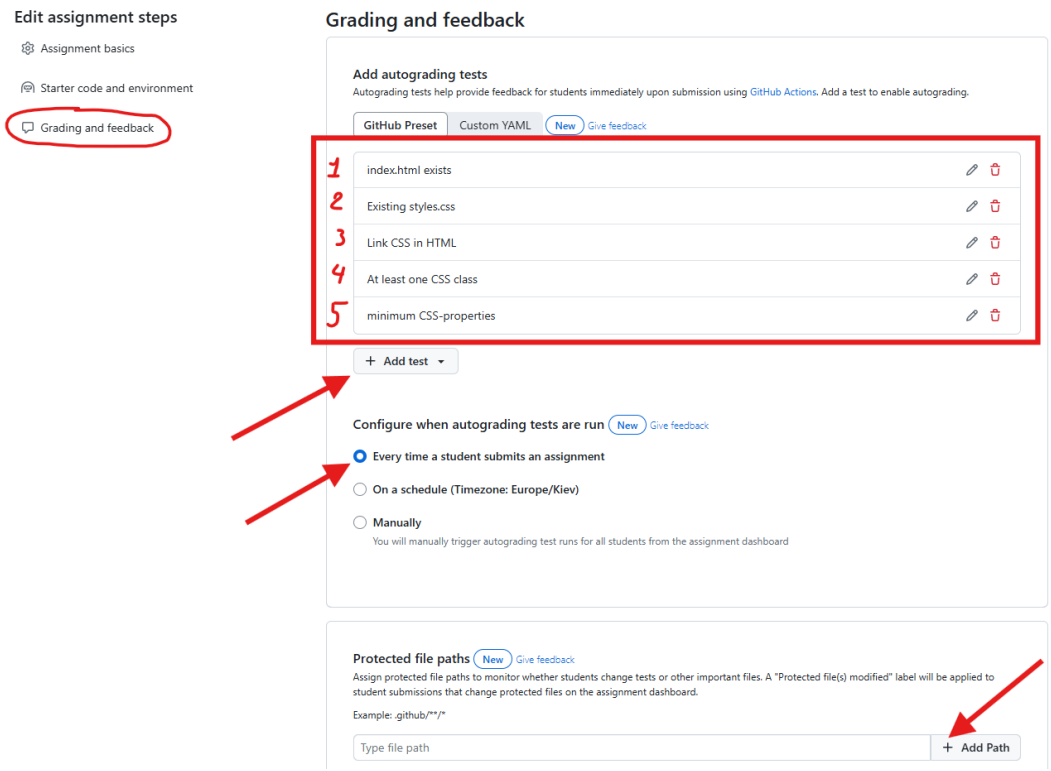


Рис. 2.8. Налаштування автоматичних тестів та оцінювання.

У пункті «Protected file path», варто задати службові директорії та файли, що не мають змінюватись студентом. Доцільно захистити від змін файл README.txt з переліком інструкцій до завдання, також скрипти автотестування з розширенням .yaml, та файл .gitignore.

2.3 Формулювання методичних рекомендацій щодо інтеграції інструментів Git та GitHub Classroom в освітній процес

Початок вивчення складних інструментів для роботи над завданнями є відповідальним моментом, і безсистемне чи надто швидке пояснення матеріалу, може відбити бажання в учнів вже на цьому етапі. Тому, далі в розділі, ми розглянемо як поступово можна підвести учнів до опанування навичок використання систем контролю версій.

Насамперед, потрібно познайомити слухачів з інтегрованим середовищем розробки, у якому будуть виконуватись завдання. У нашому випадку, обрано Visual Studio Code (VS Code), середовище, яке найбільш популярне серед розробників програмного забезпечення згідно опитування 2025-го року на сайті stackoverflow [12]. Для цього, ми пропонуємо завчасно дати домашнє завдання встановити VS Code, та переглянути відео з оглядом основних елементів інтерфейсу. Приклад інструкції зі встановлення на рис 2.9.

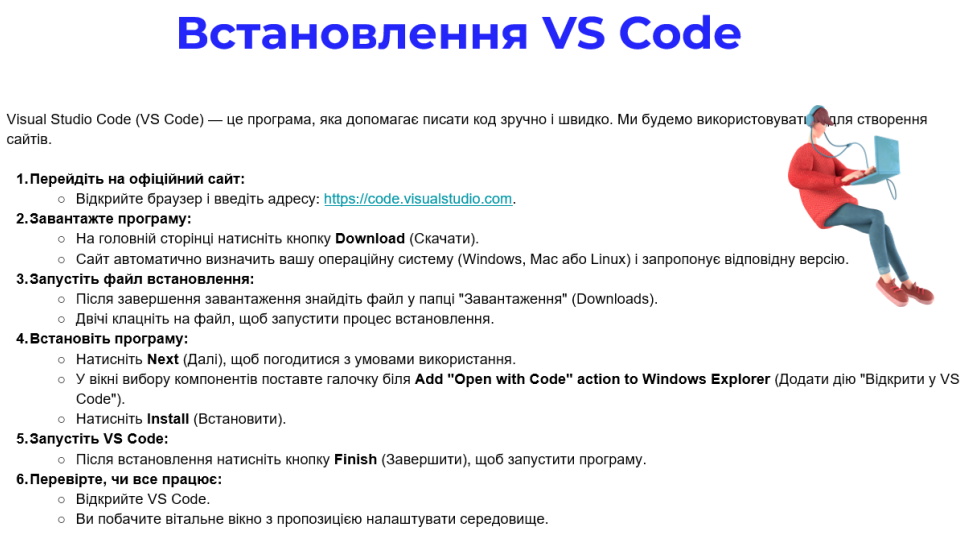


Рис. 2.9. Інструкція з встановлення редактора VS Code

Перший урок варто відвести цілком на вивчення концепцій Git, та створення учнями власних репозиторіїв коду. Мета такого уроку: дати школярам розуміння навіщо потрібні системи контролю версій, вивчити основні необхідні команди Git. Щоб «підвищити мотивацію та залученість учнів» (Романюк, О. Н.,

Романюк, О. В., & Величко, Н. П., 2024, с.15), варто додати до плану уроку використання ігрових форм навчання. Урок має містити такі частини як:

- **Постановка проблеми** − ставимо питання «Хто з вас коли-небудь робив подібні дії?», наводимо приклади назв файлів Referat.docx, Referat_final.docx, Referat_tochno_final.docx, Referat_FINAL_V2_DRUKUVATY.docx. Далі варто пояснити, що у програмуванні цей підхід є шкідливим, адже якщо зробити помилку в коді, то не вийде відмінити зміни, чи знайти причину в версіях файлу;
- **Вирішення проблеми** − запропонувати використати СКВ Git та на прикладах пояснити як її використання допомагає у розробці проєкту;
- **Огляд концептів** − показати підписану «Repository» дошку, розділену на дві частини з написами «Working directory» та «Staging Area» із хаотично розміщеними на ній фігурами героїв аніме, чи відомих кіноакторів. Пояснити, що кожній фігурі відповідає файл в робочій директорії, а у «Staging Area» будемо відбирати лише цікаві для нас файли (дія команди Add в термінології Git). Запропонувати учням, виходячи до дошки, обрати персонажів до «Staging Area». Після цього, щоб зафіксувати зміни в проекті, робиться знімок колажу з обраних фігур (файлів) − цей знімок називатимемо коміт (Commit). Далі знімок можна роздрукувати, або закинути на Google disk, з поясненням, що кожен знімок потрапляє у історію комітів. Підсумок підводиться на основі обговорення блок-схеми рис. 2.10;

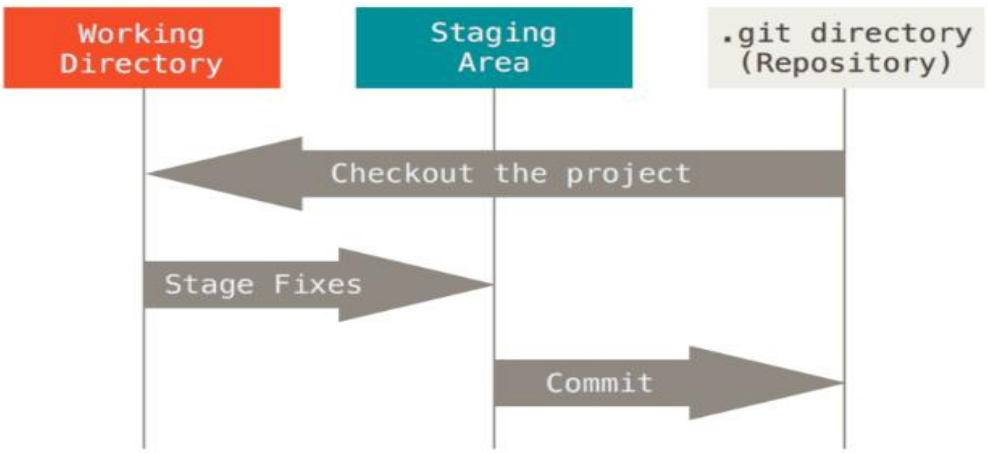


Рис. 2.10. Три можливі стани файлів що відслідковуються системою Git

- **Практична частина** – учні переходять за комп’ютери, створюють робочу директорію та відкривають її у VS code, заходять у вбудований термінал і починають вводити команди за вчителем.

Вчитель демонструє команди налаштування Git, а саме задання глобальних налаштувань імені контрибутора(програміста) та його email адреси (рис. 2.11).

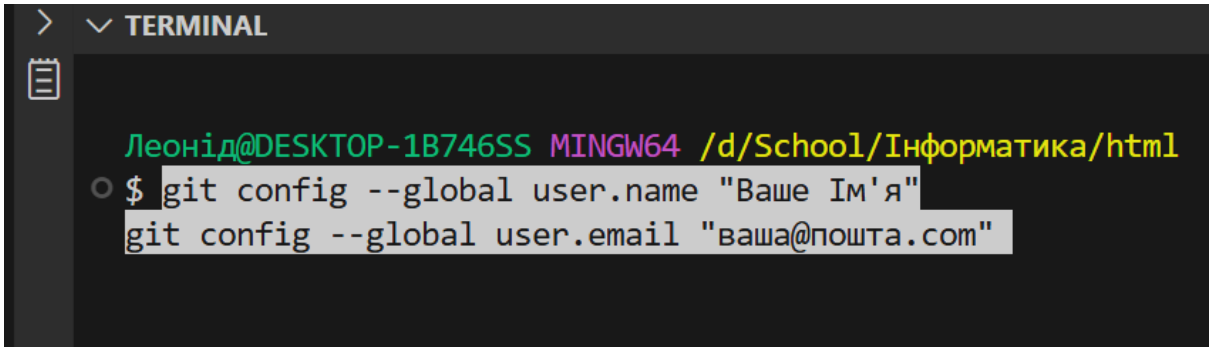


Рис. 2.11. Глобальні налаштування Git перед роботою

Командою «git init» у терміналі кожен ініціалізує свій репозиторій, можна звернути увагу учнів, що у папці проекту створюється при цьому прихована директорія з назвою «git». Наступний крок це створення першого файлу і для початку підійде .txt файл щоб спростити завдання. Кожен створює файл з іменем «story.txt» де починає оповідь реченням, до прикладу: «Жили-були котики!». За допомогою команди «status» учні відслідковують зміни у потрібних файлах (рис. 2.12).

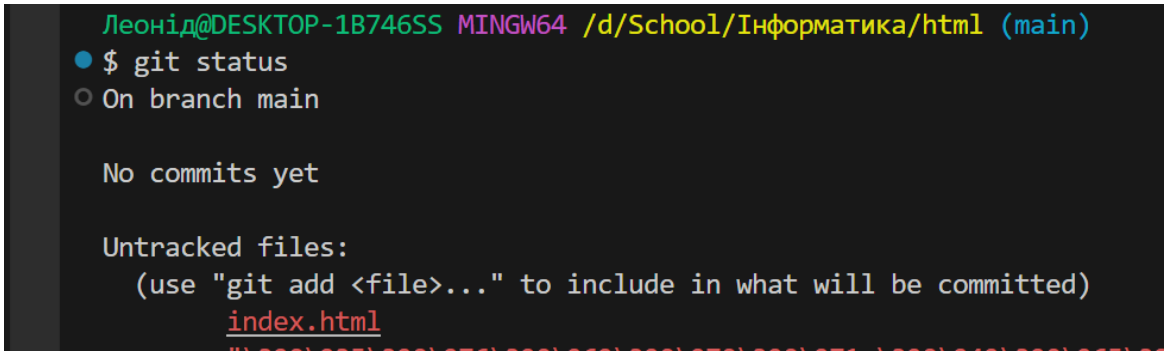


Рис. 2.12 Перевірка поточного статусу файлів

Під керівництвом учителя, учні додають файли до «Staging Area», використавши команду «git add story.txt», і знову перевіряють статус щоб побачити зміни. На останньому етапі циклу роботи з файлами робиться знімок за допомогою команди commit (рис. 2.13), тут потрібно загострити увагу на тому, що кожен знімок мусить супроводжуватись коментарем, у консолі за це відповідає прапорець «-m».

```
Леонід@DESKTOP-1B746SS MINGW64 /d/School/Інформатика/html (main)
$ git commit -m "Початок історії"
[main (root-commit) 933ce25] Початок історії
 create mode 100644 index.html
```

Рис. 2.13. Застосування команди commit для створення знімку

Використання гілок в Git бажано розглянути через демонстрацію вчителем створення нових гілок та переходу між ними. Акцентувати увагу варто на тому, як в одних гілках є код та файли, а в інших гілках - немає. В кінці уроку потрібно показати процес відправки коду на віддалений репозиторій GitHub за допомогою команди «push».

РОЗДІЛ 3.
РОЗРОБКА ІНСТРУМЕНТАРІЮ, АПРОБАЦІЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ НАВЧАННЯ З ВИКОРИСТАННЯМ GIT CLASSROOM

3.1. Розробка програмно-методичного комплексу на базі GitHub Classroom та його API

Оскільки сценарій використання Git Classroom найбільше підходить для тем, де передбачається написання коду, методичне наповнення розроблялося з наступних тем змістового модуля Веб-технології: мова гіпертекстової розмітки, каскадні таблиці стилів, проектування та верстка веб-сторінок, адаптивна верстка (Руденко, Речич & Потієнко, 2019, с.255).

В якості додаткового джерела інформації, було створено блог зі зручною навігаційною панеллю, де розмістилися теорія до уроків, та вказівки до практичних занять (рис 3.1.). Також є необхідна інформація для встановлення і налаштування Git та приєднання до GitHub. Відвідати даний ресурс можна за покликанням https://you-style-html.blogspot.com/.

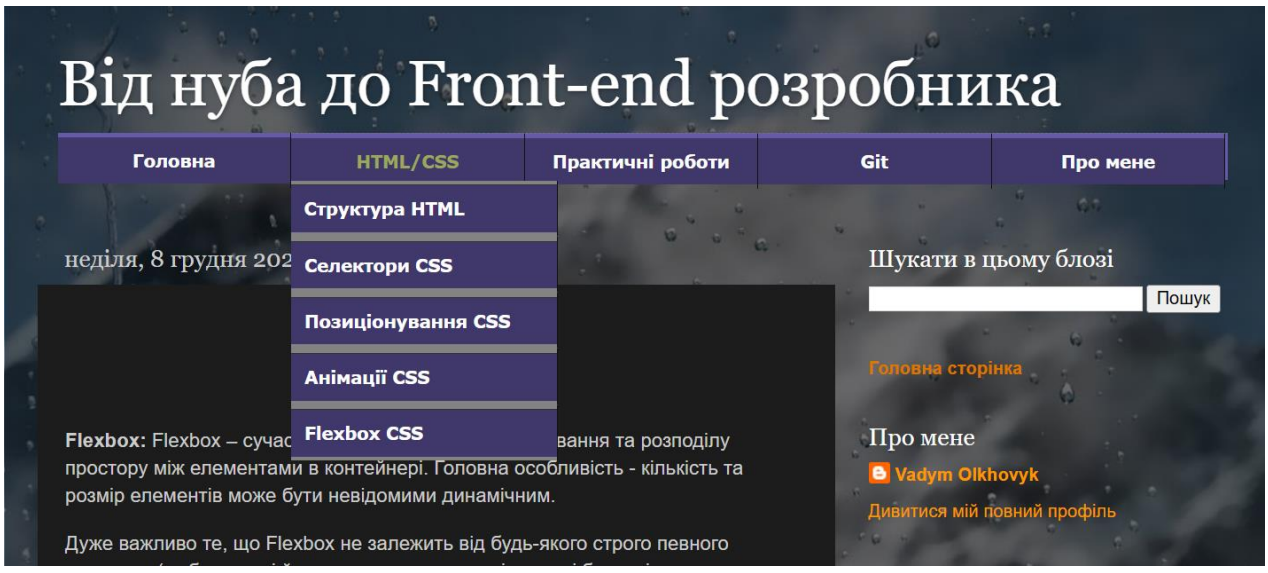


Рис. 3.1. Вигляд сторінки блога з навчальними матеріалами

На ресурсі GitHub Classroom було створено чотири домашніх роботи та дві практичні роботи (рис. 3.2.). Метою виконання домашніх завдань було

використати теоретичні знання для набуття практики без стресу отримати погану оцінку, а виконання практичних оцінювалося та впливало на підсумок за тему.

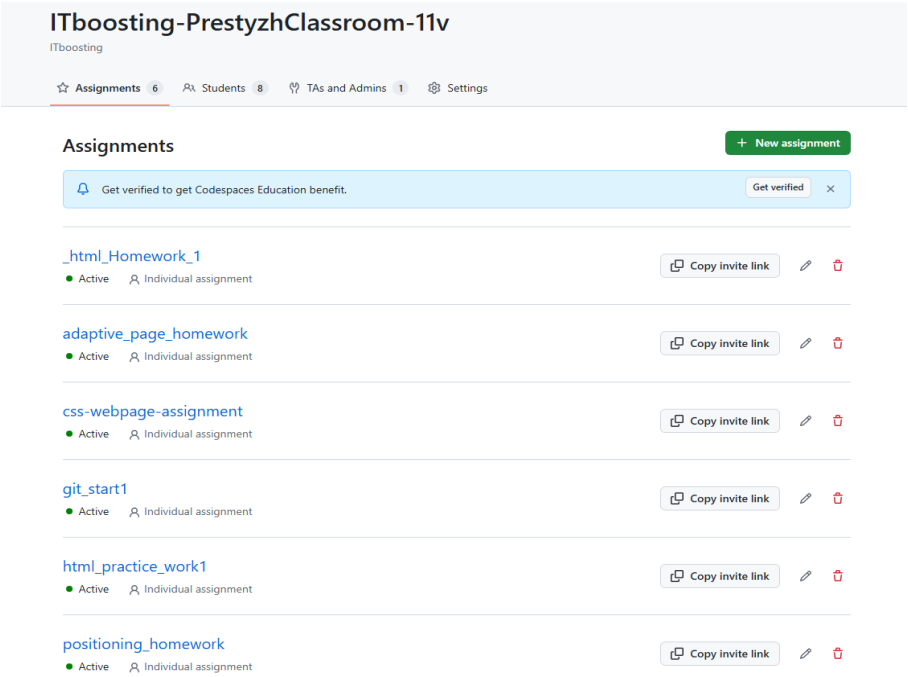


Рис. 3.2. Вигляд списку завдань у навчальному курсі Github Classroom

До розроблених завдань, вчитель розсилає учням посилання-запрошення. Цю процедуру можна зробити кількома способами. При переході, учень вибирає своє ім’я зі списку, після чого для нього генерується окремий приватний репозиторій на базі шаблону. На сторінці є можливість в один клік перейти до редагування коду в редакторі VS code (рис 3.3.).

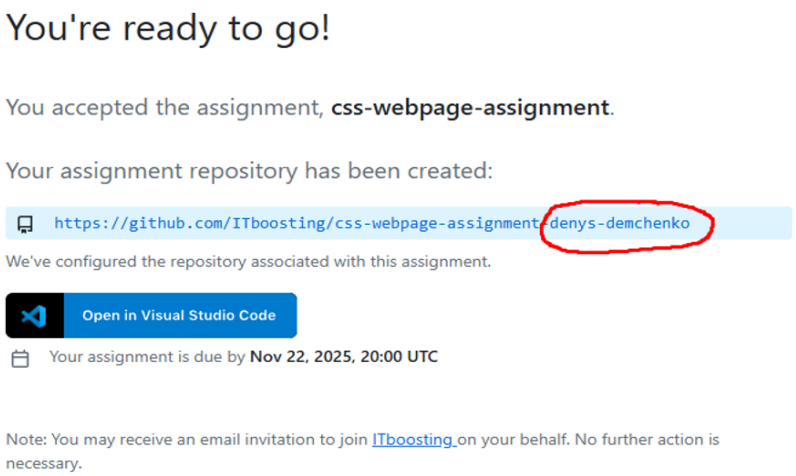


Рис. 3.3. Вигляд вікна підтвердженого учнем завдання в GitHub Classroom

З VS Code можна робити коміти та завантажувати зміни використовуючи виключно GUI. Після виконання учнем над репозиторієм дії «push», відбувається відправка знімку проекту на сервер, після чого вчитель може робити коментарі, та вносити правки до коду(рис.3.4) через механізм «pull request» (Fiksel, etc, 2019, с.117). Окрім редактора VS Code, GH Classroom вчені пропонують використовувати онлайн версію під назвою Codespaces. Його інтерфейс повторює VS Code і дає можливість почати роботу без налаштування середовища, та з мобільних пристроїв. У безкоштовній версії виділяється до 15Гб місця у хмарі для проекту та 120 хвилин процесорного часу на одне ядро (Snowberger., You, 2024, с. 269). Враховуючи, що процесорний час не дорівнює часу відкритого у браузері середовища, а використовується лише коли відбуваються обчислення, можна з впевнено стверджувати, що його вистачить для роботи в рамках шкільного курсу Інформатики.

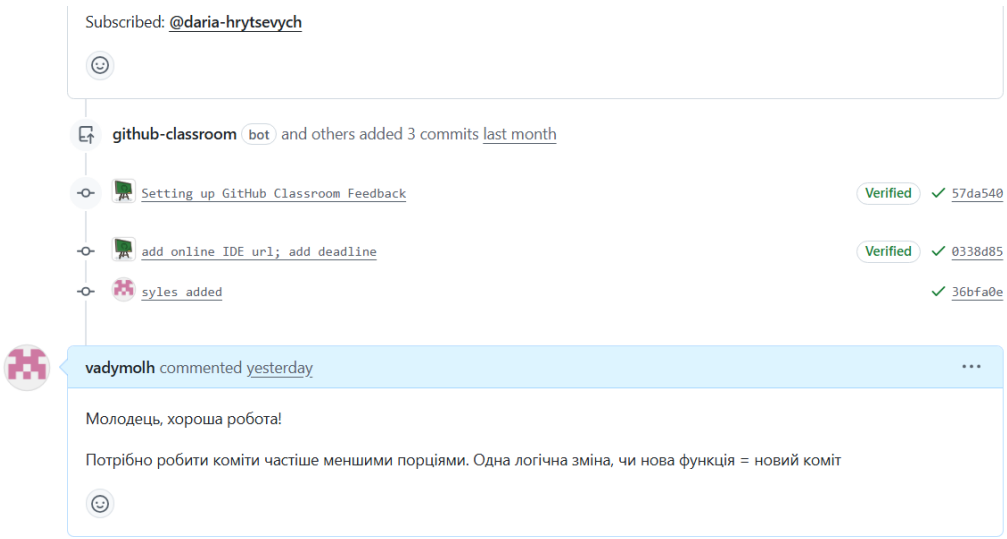


Рис.3.4. Функціонал відгуків у GitHub Classroom

Інструкції до завдання прописані у файлі «Readme.txt», учень може їх переглянути як у редакторі коду, так і в своєму, автоматично створеному, репозиторії. GitHub вміє гарно відображати простий текстовий файл «readme» у вигляді відформатованої веб сторінки (рис. 3.5.).

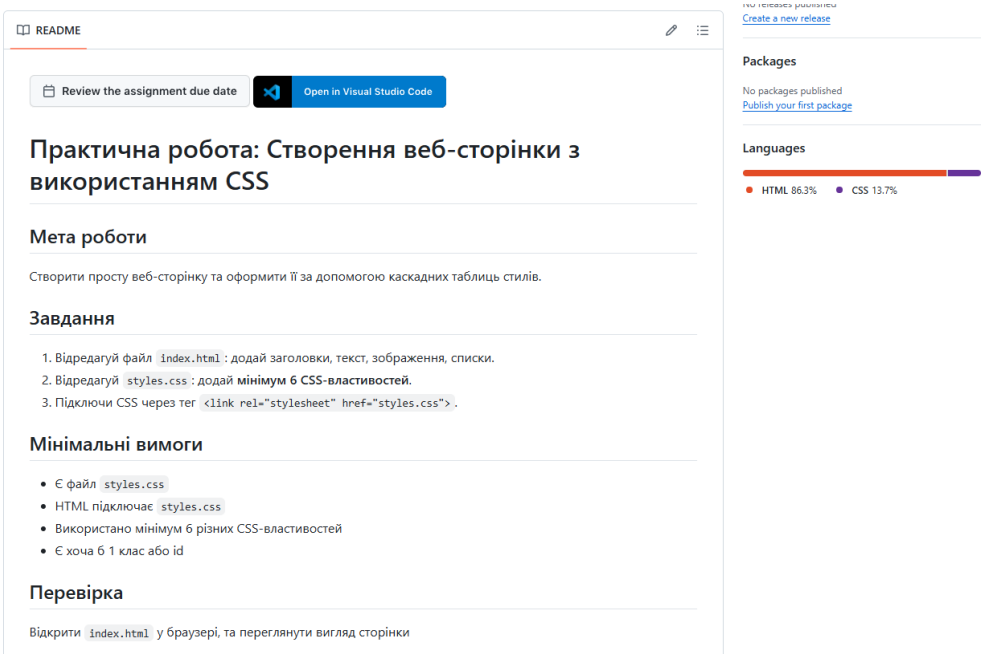


Рис. 3.5. Вигляд інструкції з Readme файлу до практичної роботи

Для оцінки практичних завдань було використано функцію «Autograding». Її можна налаштувати при створенні завдання, ми обрали для цього спосіб через заповнення форми, щоб протестувати можливість створення тестів для людей, що не знайомі з мовою розмітки yaml. Для прикладу розглянемо, як протестувати наявність селектора класу в CSS файлі (рис 3.6.). У полі «Test name» вводимо ім’я тесту, в поле «Setup command» можна вказати шлях до скрипта який запускатиметься перед перевіркою коду (залишаємо порожнім). Поле «Run command» містить умову успішного проходження тесту, у нас це лінія коду grep -g “\.” styles.css, що шукає у файлі стилів слова що починаються з крапок, коли знайдеться хоч одне співпадіння − тест вважається пройденим. Наступні поля з «Timeout» та «Points» визначають максимальний час виконання скрипту (процесорний час) і кількість можливих балів за успішно пройдений тест.

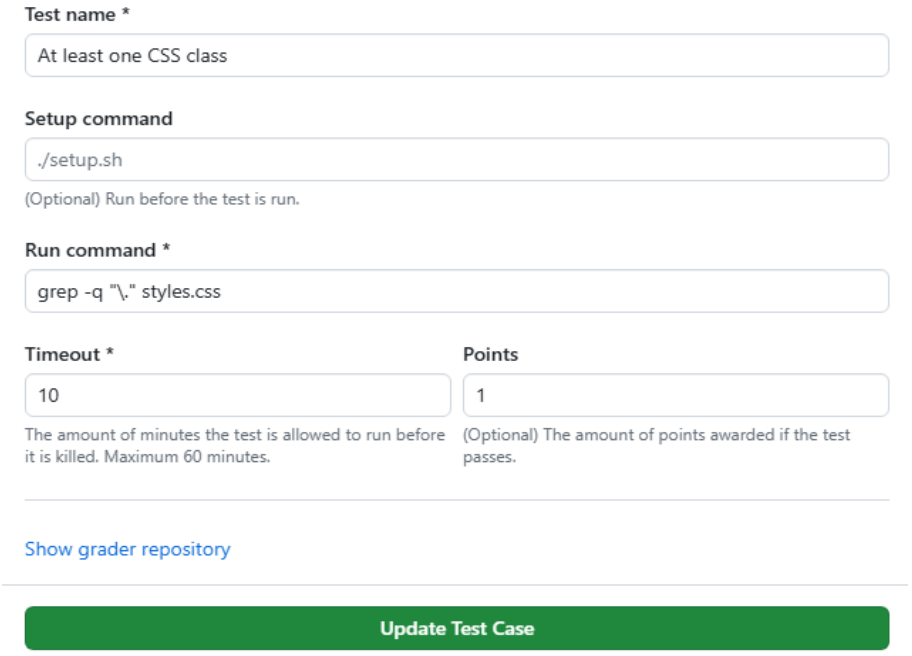


Рис. 3.6. Вікно створення автоматичного тестування коду

Для учнів процес тестування буде запускатися при кожній події «push» – завантаження свого нового коду на GitHub репозиторій. Учень отримує зворотній зв'язок з результатами тесту (рис. 3.7.) у вигляді детального списку і підсумкової таблиці. Кількість перездач не регламентується, але таку умову можна додати скриптом.

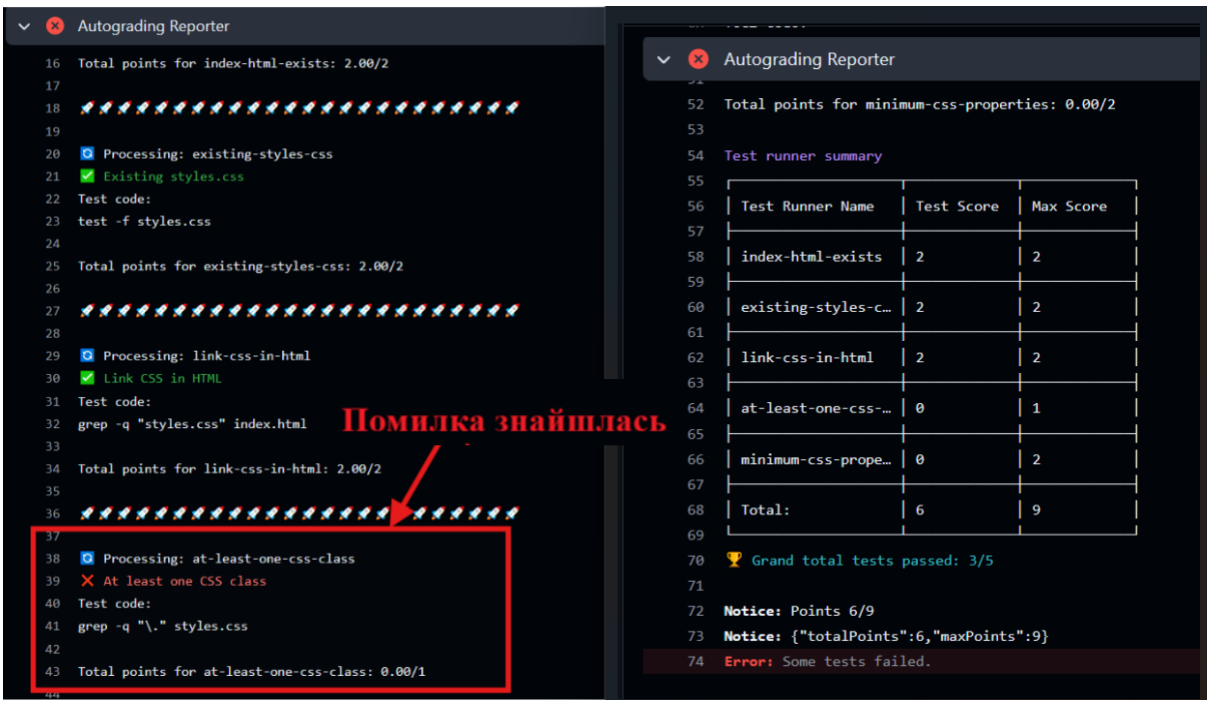


Рис. 3.7. Вигляд вікна тестування коду

У ході випробовування функції Autograding було з’ясовано, що використання кирилиці у назві тесту генерує помилку, через яку тест не спрацьовує.

3.2 Аналіз результатів та оцінка ефективності використання GitHub Classroom

Впровадити вивчення та використання Git та GitHub Classroom на уроках Інформатики в 11 класі для апробації було запропоновано в Рівненському академічному ліцеї «Престиж» ім. Лілії Котовської. Під час практики, в другій половині першого семестру 11 клас вивчав теми змістового модуля Веб-програмування. З дозволу адміністрації школи для реалізації експерименту, було виділено підгрупу з 9 учнів, котрі раніше обрали вивчати інформатику на профільному рівні. Метою даного експерименту було дослідити зміну в мотивації у вивченні програмування в результаті використання Git та GitHub Classroom при виконанні завдань, оцінити зручність інструментів як для вчителя, так і для учнів. По завершенні експерименту учням було надано анонімну анкету, з питаннями про те, чи корисним на їхню думку було вивчення Git, про рівень задоволеності навчальним процесом, і рівень засвоєння та розуміння матеріалу.

Проаналізувавши стовпчасту діаграму на рис. 3.8. можна стверджувати, що учні позитивно сприйняли нововведення і тому негативних оцінок не було.

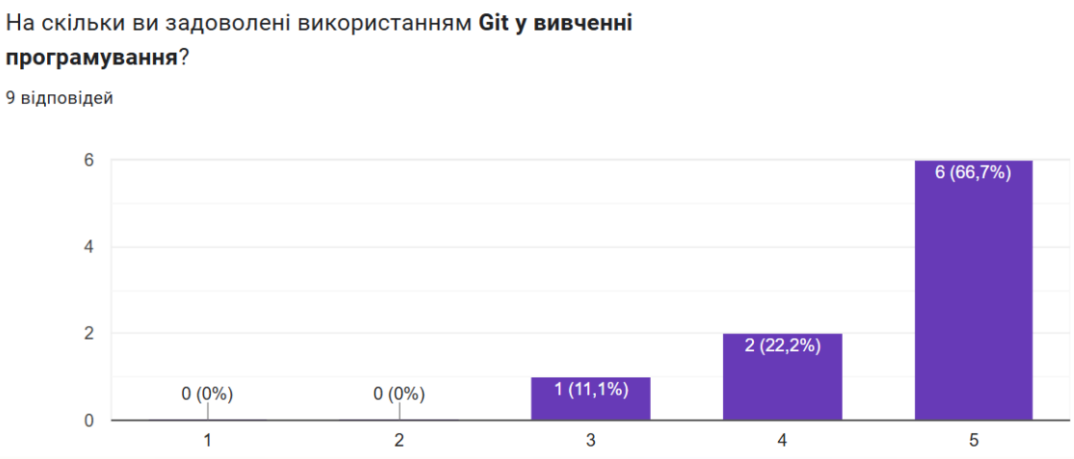


Рис. 3.8. Відповіді щодо загальних вражень використання Git

Рівень засвоєння навичок використання Git є невисоким, але в той же час 55,6% учнів навчилася застосовувати базові команди при розробці проєктів рис. 3.9. Позитивним є і значення в нуль голосів тих, хто не справлявся з використанням та розумінням системи Git взагалі.

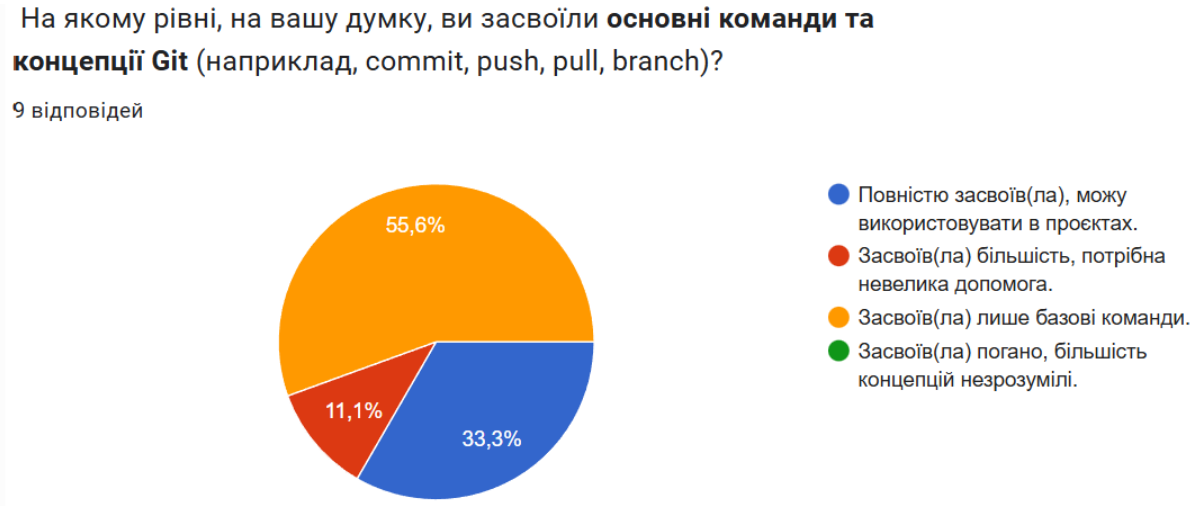


Рис.3.9 Діаграма самооцінки учнями своїх навичок володіння Git

На запитання «Чи покращило використання Git якість ваших навчальних проектів?» 88,9% респондентів відповіли що так, решті важко визначитись однозначно. Також здобувачі освіти вважають уміння працювати з Git цінним для їхнього майбутнього працевлаштування (рис. Д.1). Серед питань були і запитання з відкритою відповіддю, до прикладу на питання «Яка тема або функція Git виявилася для вас найбільш корисною?» (рис. Е.1) учні частіше згадували про створення окремих гілок версій коду, для експериментів без страху зіпсувати робочий основний код, а також про спільне написання коду.

Щодо кількості відведеного часу на засвоєння теми роботи з Git та GitHub, третина учнів побажала мати більше часу, а отже виділених занять, яких у нас було два повноцінні уроки. Рівень сприйняття складності освоєння технології Git лежить в середині діапазону між оцінкою 2 та 3 із 5-ти, де 1 легко, а 5 це складно, рис. 3.11.

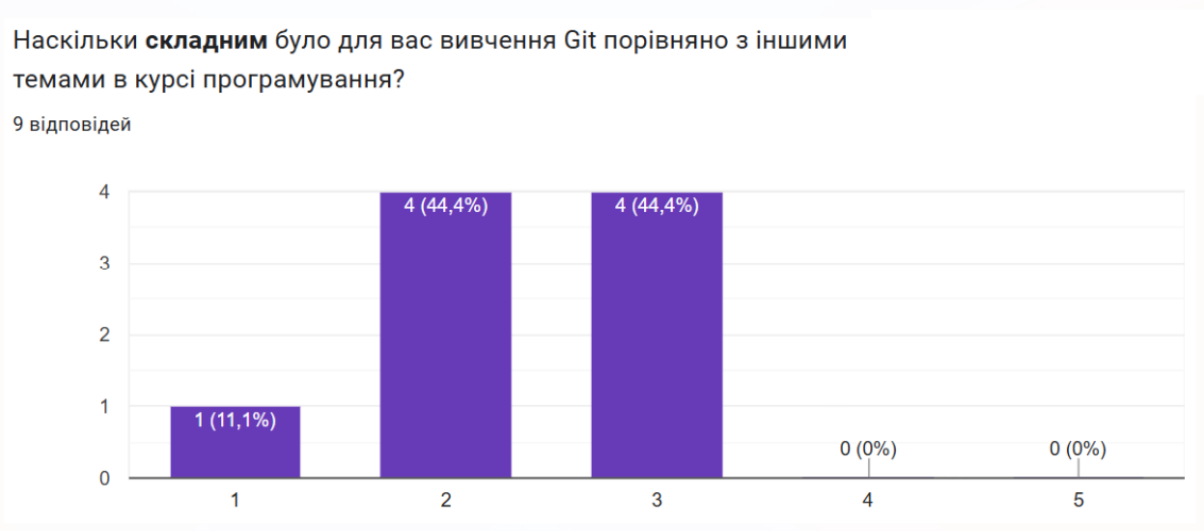


Рис. 3.10. Оцінка складності для вивчення технології Git

З опитування можна зробити висновок, що використання Git у навчальній діяльності 11 класів профільного рівня не створює перепон у навчанні, а навпаки, мотивує здобувачів освіти.

ВИСНОВКИ

Освітня політика України в наш час спрямована на якісні зміни, а навчальний предмет “Інформатика” є одним з найбільш вимогливим до регулярних оновлень навчальних програм. Останні досягнення в області ШІ чітко показують, що той хто не знатиме актуальних технологій, і не зможе їх застосувати в професійній діяльності, ризикує опинитися не затребуваним на ринку праці. Саме тому, опанування актуальних технологій та інструментів програмування має відбуватися ще у середній школі.

Дана робота була спрямована на огляд систем контролю версій коду, вирішення питань доцільності та складності впровадження вивчення цих систем у рамках курсу з інформатики для закладів загальної середньої освіти. Було чи не вперше запропоновано ввести в навчальний процес середньої школи використання системи контролю навчанням GitHub Classroom. Усі проаналізовані літературні джерела свідчать, що GitHub Classroom досі використовувався здебільшого у закладах вищої освіти. За висновками наукових робіт та статей], це підвищило мотивацію здобувачів освіти до вивчення програмування, схожу тенденцію було помічено і в результатах нашого експерименту з апробації GitHub Classroom.

У рамках підготовки до апробації курсу з використанням GitHub Classroom, було розроблено методичний комплекс, який складається з інструкцій по налаштуванню та користування системою, теоретичних матеріалів, практичних робіт, та автоматизованих тестів оцінювання коду з використанням Github Actions. Така автоматизація дозволила учням самостійно опрацьовувати допущені помилки, адже здобуття усіх «зелених» відміток підвищують рівень задоволеності процесом навчання. У той же час, для вчителя практично зникає потреба перевіряти роботи які не пройшли базове тестування, він може зосередитись на порадах зі стилістики коду, підказках до складніших завдань, оцінювання творчого компоненту роботи чи обговоренні з учнями групових проектів. Корисними виявилась функція створення завдань за шаблонним репозиторієм, завдяки чому, усі учні працювали з

однаковою ієрархією файлів у проекті. Це розвиває культуру написання коду, спрощує його перевірку.

За результатами експерименту, використання GitHub Classroom для вивчення теми Веб-технології 11 класу у ліцеї, було відмічено покращення якості коду, здобуття учнями базового рівня володіння системою контролю версій Git, отримання навичок роботи з професійними інструментами, такими як редактор VS Code та Codespaces. Усе перелічене підтверджує ефективність обраної методики. Разом з тим, варто констатувати, що вимоги до кваліфікації вчителя при використанні GitHub Classroom зростуть.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Горошкіна О. / та ін. (2022). Компетентнісно орієнтоване навчання: сутність, форми і методи: *навч. посіб. Київ : Пед. думка*. 221 с.

2. Semenykhina, Olena & Rudenko, Yuliia. (2018). Проблеми навчання програмувати учнів старших класів та шляхи їх подолання. *Information Technologies and Learning Tools*.

3. Млавець Ю.Ю. Методика навчання інформатики *(конспект лекцій для студентів факультету суспільних наук).* – Ужгород: ДВНЗ "УжНУ", 2021. – 57 с.

4. Сікора Я., Карплюк С., Грінчук І., Оленюк Д.(2022) Використання методу проектів на уроках інформатики в закладах загальної середньої освіти як одна із ефективних педагогічних технологій. Перспективи та інновації науки. *Серія: «Педагогіка», «Психологія», «Медицина». № 8. 2022. № 8.* С. 278 – 288.

5. Грибик Т. Т. (2024). *О*собливості використання системи контролю версій при розробці освітніх проєктів з інженерії програмного забезпечення. *Інноваційна педагогіка*. Т. 1, № 70.

6. Mosiiuk O. (2025). Особливості навчання майбутніх учителів інформатики роботі із системою контролю версій Git. *Науковий вісник Ужгородського університету. Серія: «Педагогіка. Соціальна робота»*, (2(49), 107–110).

7. Bergmann, J. & Sams, A. (2012). *Flip Your Classroom: Reach Every Student in Class Every Day. Washington, DC : ISTE.*

8. Бобровський М. (2016). Технологія «перевернутих» уроків та можливості її впровадження у навчальних закладах Києва. Проблеми та перспективи управління сучасною столичною школою: *зб. наук. ст. за матер. регіон. наук.-практ. конф. Київ : Київ. ун-т ім. Б. Грінченка*. С. 3–6.

9. Бобровський М., Якубов С. (2017). Експеримент з дистанційного навчання у школах м. Києва, перехід до змішаного навчання з елементами персоналізації на основі платформи Moodle 3.2*//П'ята міжнародна науково-*

практична конференція «MoodleMootUkraine Теорія і практика використання системи управ-ління навчанням Moodle»: тези доповідей. - К.: КНУБА. – с. 8.

10. Titova, Liubov. (2024). *Гейміфікація у навчанні програмування// Комп'ютерні ігри та мультимедіа як інноваційний підхід до комунікації -/ Матеріали IV Всеукраїнської науково-технічної конференції молодих вчених, аспірантів і студентів, Одеса, 26-27 вересня 2024 р. - Одеса, Видавництво ОНТУ*, 2024 р. – с.105-107.

11. М. Медведєва, Є. Богульська,(2024) *«Методика використання гейміфікації на уроках інформатики», у Сучасні інформаційні технології в освіті і науці, Умань, 25-26 квіт. 2024. Умань*, (с. 136–140).

12. Тимощук, Г. В. (2024). *Гейміфікація освітнього процесу: сучасний погляд. Наукові записки. Серія: Педагогічні науки*, (216), 328-332.

13. Uddin, M. S., Hasan, K. J., & Tasnima, R. (2023). A Comparative Analysis of the Rise of Git-Based Distributed Collaborative Hosting Platforms: Survey, Performance Test, and Comparison. *Asian Journal of Engineering and Applied Technology*, *12*(2), 29-33.

14. Dmytrenko, D., Alieksieieva, H., & Khomenko, S. (2023). BITBUCKET як рішення для організації командної роботи над IT проектом. *ScienceRise: Pedagogical Education*, (2 (53)), 14-19.

15. Sprint, G., & Conci, J. (2019). Mining github classroom commit behavior in elective and introductory computer science courses. *The Journal of Computing Sciences in Colleges*, *35*(1).

16. Mathias Meyer. (2014). *Continuous integration and its tools. IEEE software 31, 3* (2014), 14–16.

17. Jon Loeliger and Matthew McCullough. 2012. Version Control with Git: Powerful tools and techniques for collaborative software development. " O’Reilly Media, Inc.".

18. Охріменко, З. В. (2021). Профорієнтація як умова ціннісного професійного самовизначення учнів. *Науковий вісник Національного еколого-натуралістичного центру*, *1*(11), 16-27.

19. Пригодій, М. А. (2024). *Функціональні особливості сучасних систем управління навчанням.*

20. Loftsson, H. (2021, January). Moving Classes in a Large Programming Course Online: An Experience Report. In *Proceedings of the 2nd International Computer Programming Education Conference*.

21. Alves, P., & Cipriano, B. P. (2025). *Automated Assessment in Mobile Programming Courses: Leveraging GitHub Classroom and Flutter for Enhanced Student Outcomes. arXiv preprint arXiv:2504.04230*.

22. Nelson, M. A., & Ponciano, L. (2021). Experiences and insights from using Github Classroom to support Project-Based Courses. In *2021 Third International Workshop on Software Engineering Education for the Next Generation (SEENG)* (pp. 31-35). IEEE.

23. Романюк, О. Н., Романюк, О. В., & Величко, Н. П. (2024). *Особливості гейміфікації навчання. Advanced top technology.№ 2: 15-18.*

24. Руденко, В. Д., Речич, Н. В., & Потієнко, В. О. (2019). *Інформатика (профільний рівень): Підручник для 11 класу закладів загальної середньої освіти.* Ранок.

25. Fiksel, J., Jager, L. R., Hardin, J. S., & Taub, M. A. (2019). Using GitHub classroom to teach statistics. *Journal of Statistics Education*, *27*(2), 110-119.

26. Snowberger, A. D., & You, K. (2024). Creating a standardized environment for efficient learning management using github codespaces and GitHub Classroom. *Journal of Practical Engineering Education*, *16*(3_spc), 267-274.

27. Tu, Y. C., Terragni, V., Tempero, E., Shakil, A., Meads, A., Giacaman, N., & Blincoe, K. (2022, February). Github in the classroom: Lessons learnt. In *Proceedings of the 24th Australasian Computing Education Conference* (pp. 163-172).

28. Angulo, M. A., & Aktunc, O. (2019, April). Using GitHub as a teaching tool for programming courses. In *2018 Gulf Southwest Section Conference*.

29. Snowberger, A. D., & Lee, C. H. (2023). A workflow for practical programming class management using github pages and github classroom. *Journal of Practical Engineering Education*, *15*(2), 331-339.

30. Gunnarsson, S., Larsson, P., Månsson, S., Mårtensson, E., & Sönnerup, J. (2017). Enhancing student engagement using GitHub as an educational tool.

31. Zagalsky, A., Feliciano, J., Storey, M. A., Zhao, Y., & Wang, W. (2015, February). The emergence of github as a collaborative platform for education. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing* (pp. 1906-1917).

32. Puryear, B., & Sprint, G. (2022). Github copilot in the classroom: learning to code with AI assistance. *Journal of Computing Sciences in Colleges*, *38*(1), 37-47.

33. Čižmešija, A., Stapić, Z., & Bubaš, G. (2018). Using github in software engineering course: analysis of students'acceptance of collaborative coding platform. In *ICERI2018 Proceedings* (pp. 5857-5866). IATED.

34. Delgado, D. C. (2025, June). Innovating computer science education with github integration. In *Conference Proceedings. The Future of Education 2025*.

35. Marquadson, J. (2019). Learning by teaching through collaborative tutorial creation: Experience using GitHub and AsciiDoc. *Journal of Information Systems Education*, *30*(1), 10-18.

36. Ying, K. M., & Boyer, K. E. (2020, April). Understanding students' needs for better collaborative coding tools. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1-8).

37. Rahman, M. M., Paudel, R., & Sharker, M. H. (2019, October). Effects of infusing interactive and collaborative learning to teach an introductory programming course. In *2019 IEEE Frontiers in Education Conference (FIE)* (pp. 1-8). IEEE.

38. Valdivia, R. G. B. (2019). Collaborative learning using git with gitlab in students of the engineering programming course In *CISETC* (pp. 92-101).

39. Яцько, О. (2023). Аналіз навчання вибіркового модуля «Веб-технології» у 10-11 класах ЗЗСО. *Освіта. Інноватика. Практика*, *11*(3), 52-59.

40. Навчальна програма з інформатики (профільний рівень) для 10-11 класів загальноосвітніх шкіл. URL: https://mon.gov.ua/static-

objects/mon/sites/1/zagalna%20serednya/programy-10-11-klas/prof-riven.pdf (дата звернення: 11.11.2025)

41. Навчальна програма з інформатики (рівень стандарту) для 10-11 класів загальноосвітніх шкіл URL: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-10-11-klas/2018-2019/informatika-standart-10-11.docx (дата звернення: 11.11.2025).

Інтернет ресурси:

1. Про затвердження Державного стандарту профільної середньої освіти https://zakon.rada.gov.ua/laws/show/851-2024-%D0%BF#n10

2. ClassCraft. URL: https/www.classcraft.com (дата звернення 1.12.2025)

3. CodeCombat. URL: https://codecombat.com (дата звернення 1.12.2025)

4. Codingame. URL: https://www.codingame.com/start (дата звернення 1.12.2025)

5. GitHub 2025 щорічний звіт . URL: https://github.blog/news-insights/octoverse/octoverse-a-new-developer-joins-github-every-second-as-ai-leads-typescript-to-1/ (дата звернення 15.11.2025)

6. Практична робота зі створення ігор https://ua.izzi.digital/DOS/1183864/1410274.html (дата звернення 1.12.2025)

7. GitLab URL: https://web.archive.org/web/20211027181044/https://medium.com/@jhchen/building-an-open-source-company-interview-with-gitlabs-ceo-290180d172da (дата звернення 1.11.2025)

8. Asplund, J. E., & Ballve, M. (2025). Replit. URL: https://sacra-pdfs.s3.us-east-2.amazonaws.com/replit.pdf (дата звернення 1.12.2025)

9. Використання Replit у Вищій школі. Блог URL: https://scottie.is/writing/repl-it-classroom-documentation/ (дата звернення 1.12.2025)

10. Закон України «Про застосування англійської мови в Україні» https://zakon.rada.gov.ua/laws/show/3760-20#Text (дата звернення 1.12.2025)

11. Документація GitHub по створенню організації https://github.com/rzn-example-classroom/tutorial (дата звернення 1.12.2025)

12. Щорічне опитування розробників на Stackoverflow за 2025 рік URL: https://survey.stackoverflow.co/2025/ (дата звернення 1.12.2025)

13. GitHub Classroom Documentation. URL: https://classroom.github.com (дата звернення 1.12.2025)

Додатки

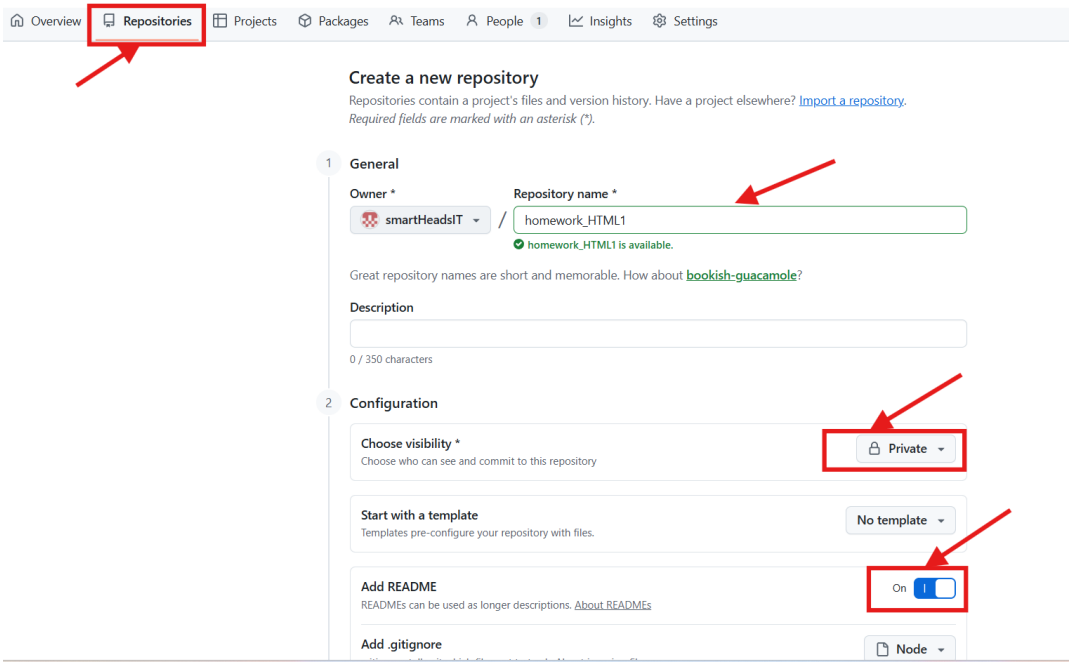
ДОДАТОК А

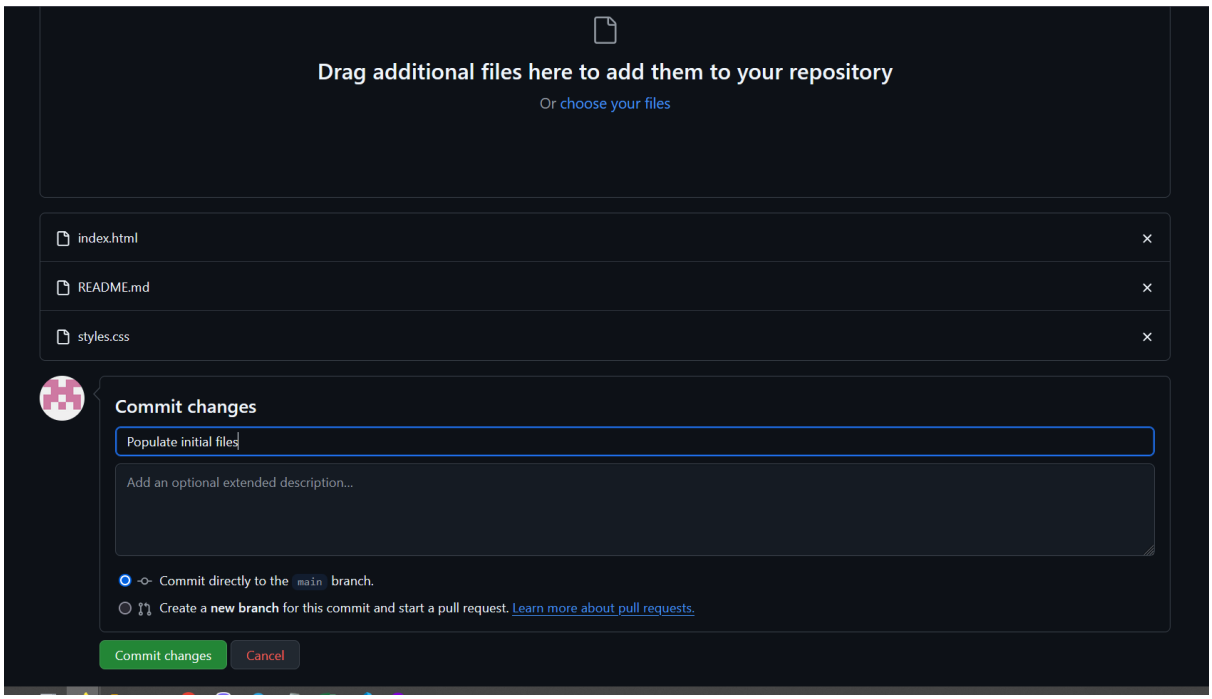
Порівняння сервісів хмарного сховища коду

Параметр	GitHub	GitLab	Bitbucket
Підтримувані системи контролю версій (СКВ)	Лише Git	Лише Git	Git і Mercurial
Публічні репозиторії	Необмежені, Free	Необмежені, Free	Необмежені, Free
Приватні репозиторії	Необмежені, безкоштовні для користувачів; додаткові можливості в платних планах	Необмежені приватні репозиторії навіть у безкоштовній версії	Приватні репозиторії доступні у free-версії, але з обмеженням користувачів у робочому просторі
Навігація	Функції навігації по коду присутні, гарна структура	Добра навігація по коду, особливо в self-hosted	Достатня, але простіша за GitHub
Інтерфейс користувача	Функціональний, складний	Технічно гнучкий; багато налаштувань	Зрозумілий інтерфейс, простіший за GitHub і GitLab
Wiki	Є wiki для кожного репозиторію	Є wiki, з розширеним форматом та історією	Є wiki з вибором доступності
Підтримка та співпраця	PR, Code Review, Projects, Discussions. підходить для open-source команд	Merge Requests, Issue Boards, Milestones, прогресивна робота з DevOps	Орієнтація на продукти Atlassian (Jira, Confluence). Зручний для корпоративних команд
Аналіз проєкту	Базовий перегляд історії, граф комітів, insights	DevOps-аналітичний набір: CI/CD pipelines, тестові звіти, аналітика	Візуалізація комітів, історії, інтеграція з Jira
Інтеграції	Marketplace з тисячами інтеграцій; GitHub Actions для CI/CD	Інтеграції з Jira, Slack, VS Code, Kubernetes; GitLab CI/CD вбудований	Тісна інтеграція з Atlassian (Jira, Trello, Confluence), CI/CD через Bitbucket Pipelines
Ціна	Free, Team $4/корист., Enterprise $21/корист.	Free, Premium $29/корист., Ultimate — для enterprise	Free до 5 користувачів, Standard $3/корист., Premium $6/корист.

ДОДАТОК Б

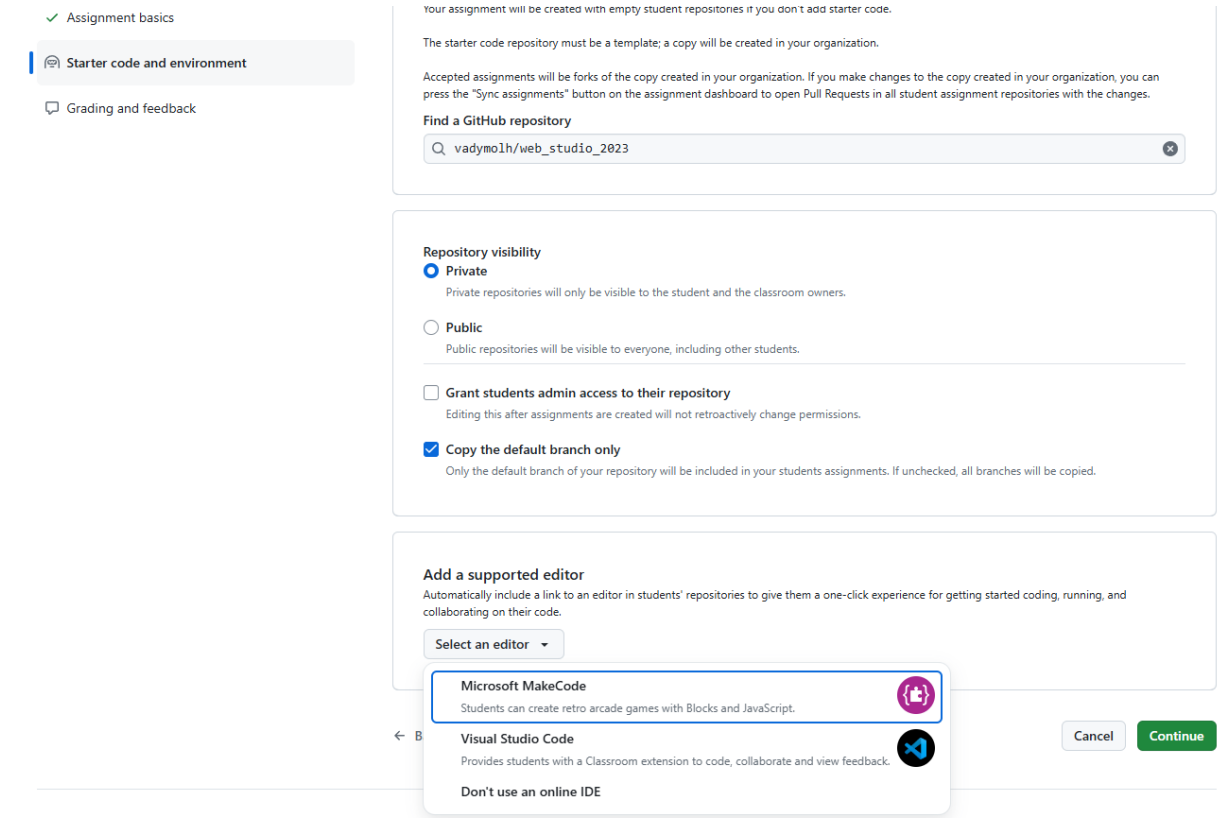
Створення приватного репозиторію організації

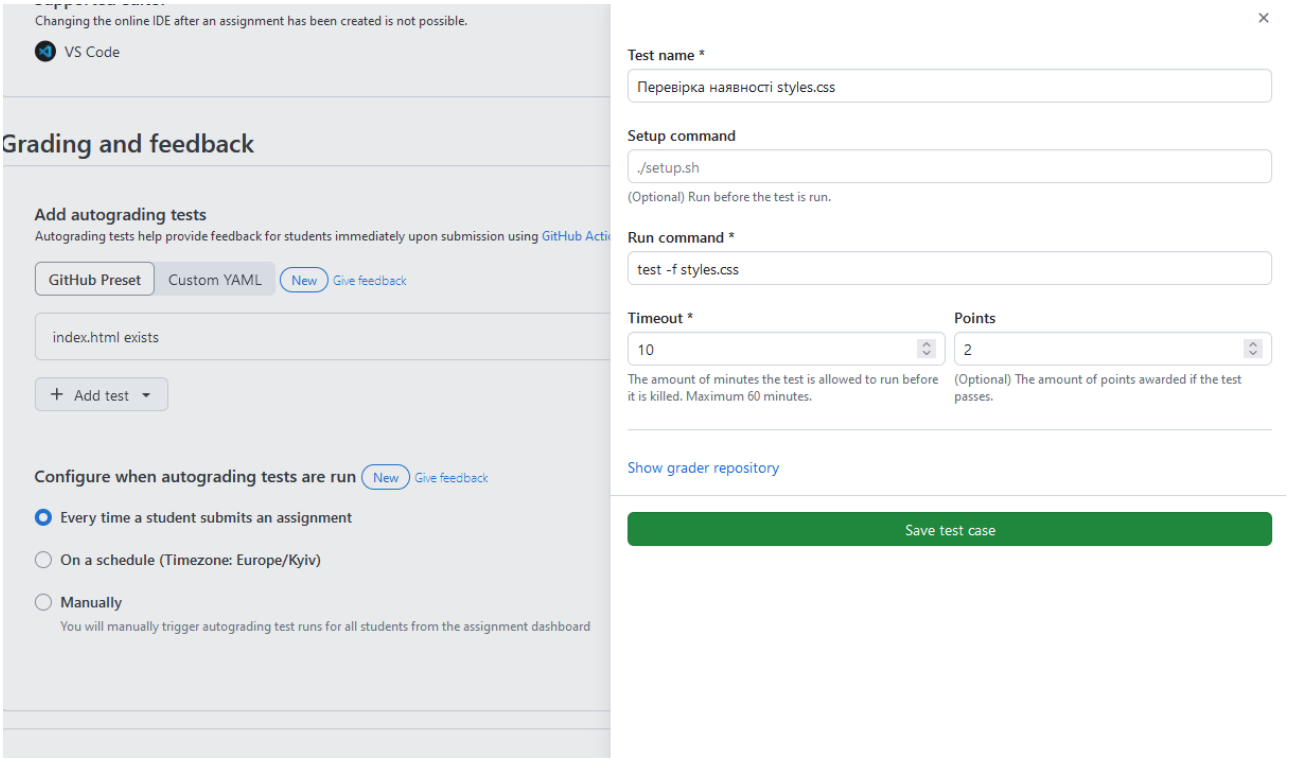




ДОДАТОК В

Створення нового завдання в GitHub Classroom





ДОДАТОК Г

Міністерство освіти і науки України
Департамент освіти і науки Рівненської ОДА
Рівненський державний гуманітарний університет

СЕРТИФІКАТ № 2025-062

Учасника
XVIII Всеукраїнської науково-практичної конференції

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРОФЕСІЙНІЙ ДІЯЛЬНОСТІ

10 листопада 2025 року, м. Рівне

Ольховик Вадим

Обсяг - 15 годин (0,5 кредиту ЄКТС)

Завідувачка кафедри цифрових технологій
та методики навчання інформатики РДГУ,
голова програмного комітету конференції

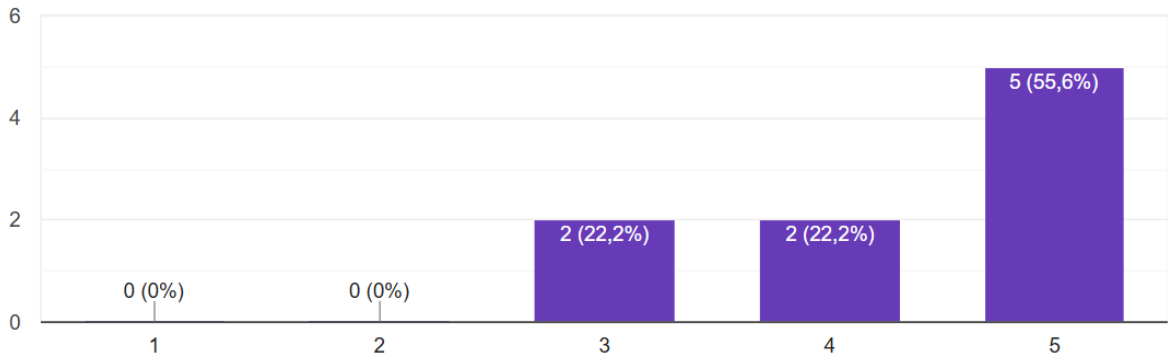
проф. Наталія ПАВЛОВА

Додаток Д

Результати анкетування учнів після експерименту з використання Git та GitHub Classroom

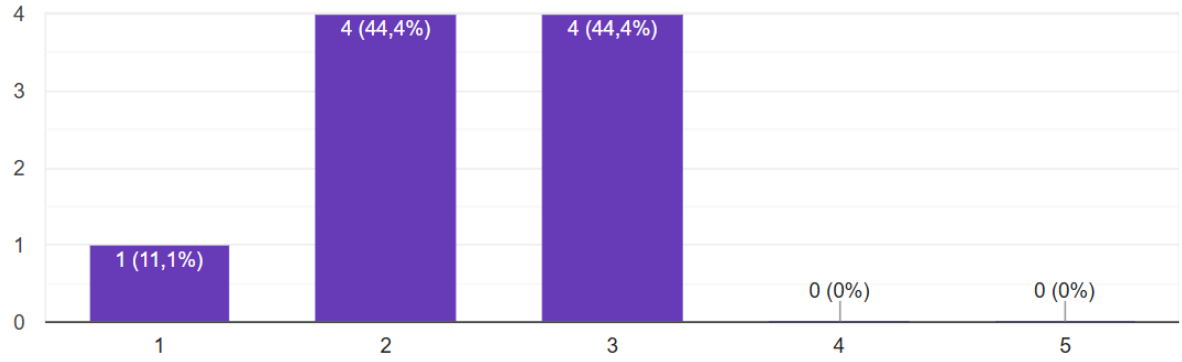
Наскільки ви згодні з твердженням, що знання та вміння працювати з Git **допоможе вам у майбутньому працевлаштуванні** у сфері IT?

9 відповідей



Наскільки **складним** було для вас вивчення Git порівняно з іншими темами в курсі програмування?

9 відповідей



Додаток Е

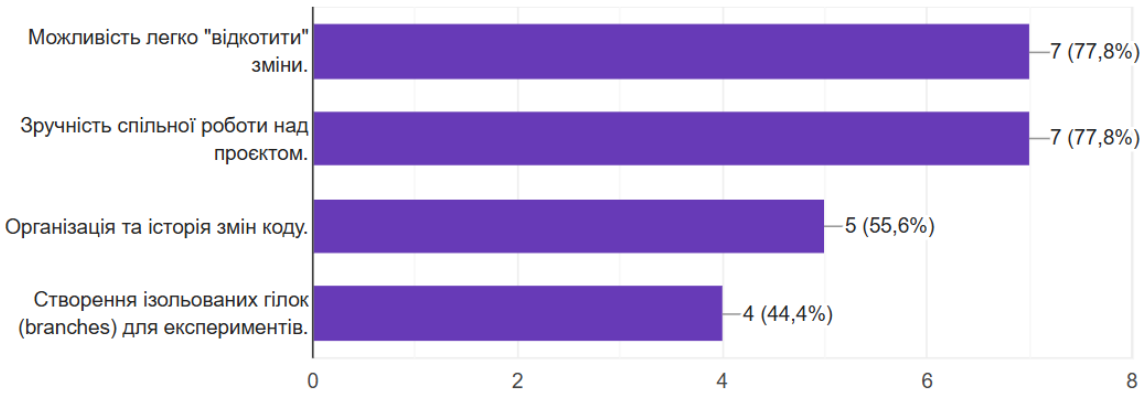
Яка **тема або функція Git** виявилася для вас **найбільш корисною** або цікавою? (Наприклад, робота з гілками, вирішення конфліктів, використання GitHub)

6 відповідей

- Розгалуження
- Можливість створення гілок до вже існуючого проекту
- можливість створювати інші гілки для експериментів без впливу на основний код
- Користування GitHub, Поширення та спільне написання коду
- Вирішення конфліктів
- Функція branch, тобто створення гілок та робота з ними

Які **найбільші переваги** використання систем контролю версій (Git) ви можете виділити для себе у навчальному процесі та поза ним? (Можна обрати декілька або вписати свою)

9 відповідей



Додаток Є

Результати анкетування учнів після експерименту з використання Git та GitHub Classroom

Які додаткові **інструменти, ресурси або матеріали** (окрім тих, що використовувалися) могли б покращити ваше розуміння Git?

9 відповідей

На якому рівні, на вашу думку, ви засвоїли **основні команди та концепції Git** (наприклад, commit, push, pull, branch)?

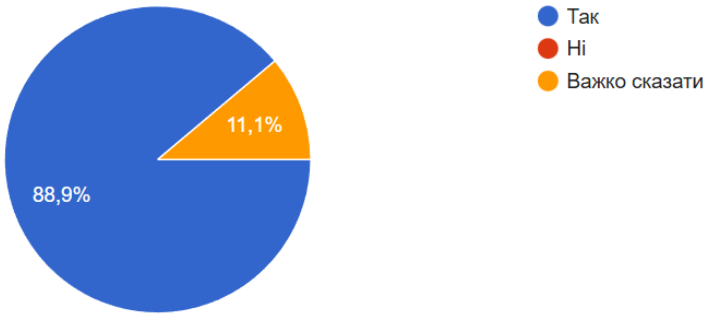
9 відповідей

Додаток Ж

Результати анкетування учнів після експерименту з використання Git та GitHub Classroom

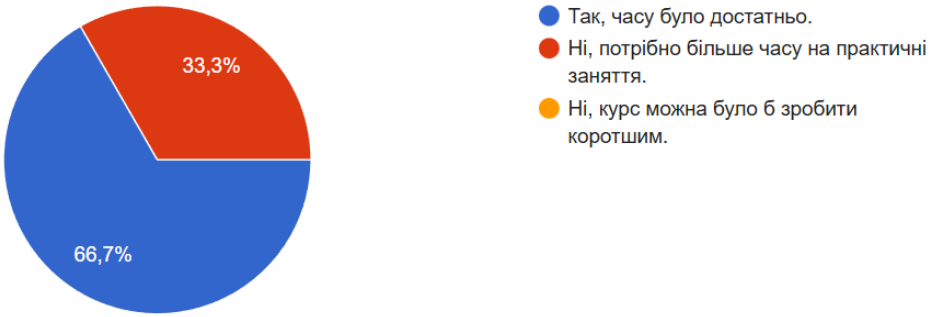
Чи вважаєте ви, що використання Git **покращило якість** ваших навчальних проєктів та/або спільної роботи з однокласниками?

9 відповідей



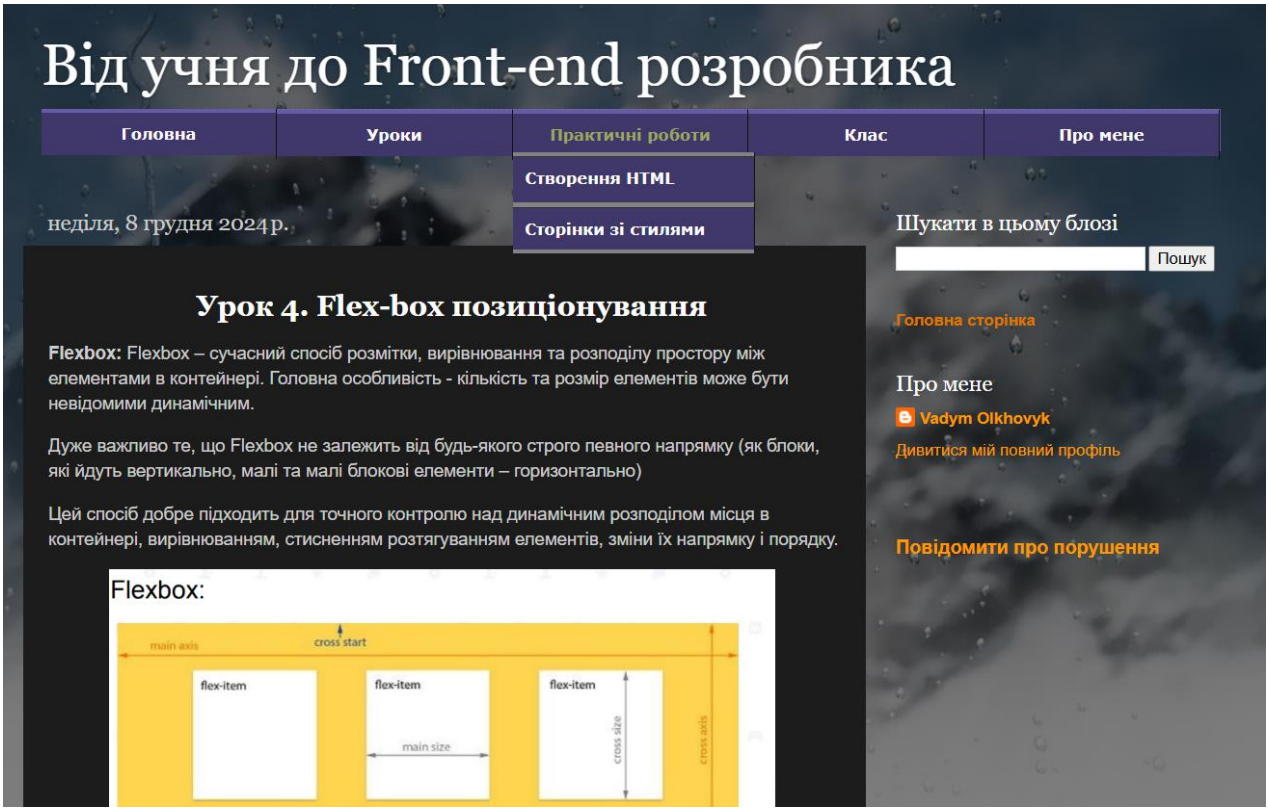
Чи вважаєте ви, що на вивчення Git у межах курсу було відведено **достатньо часу**?

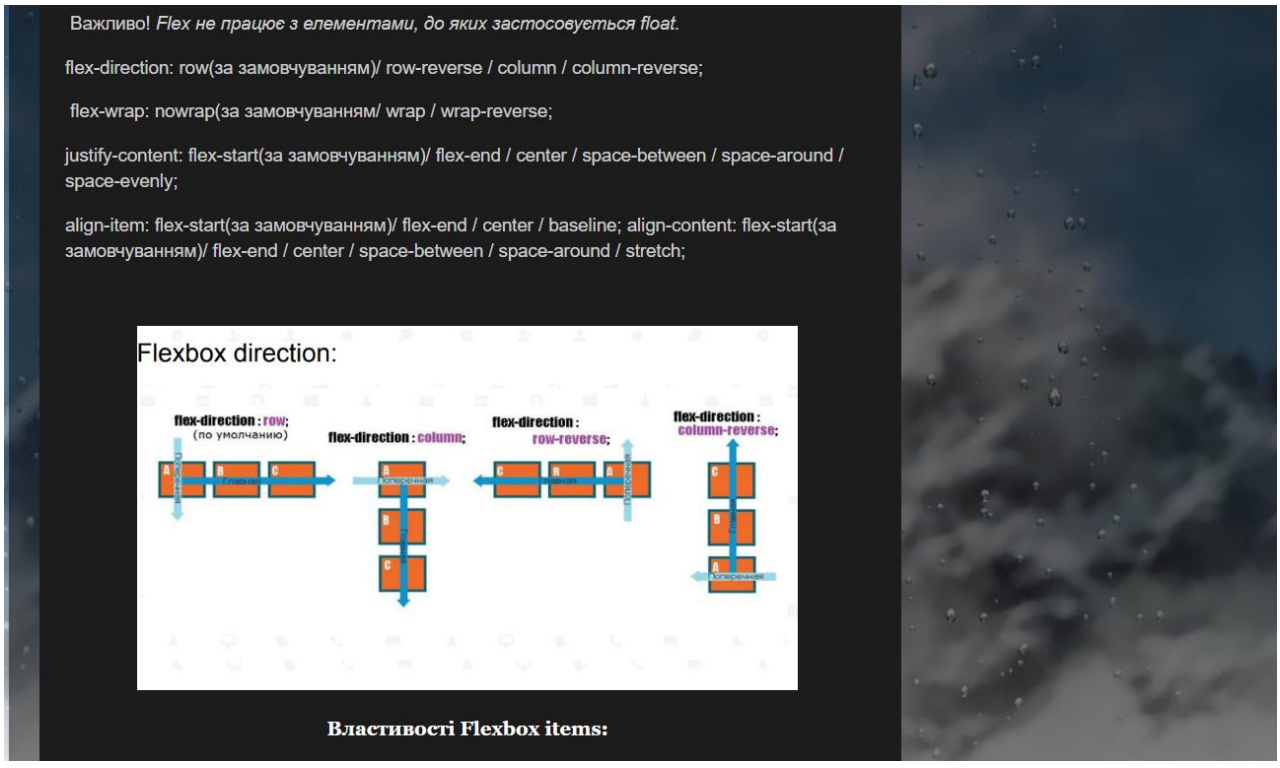
9 відповідей



Додаток З

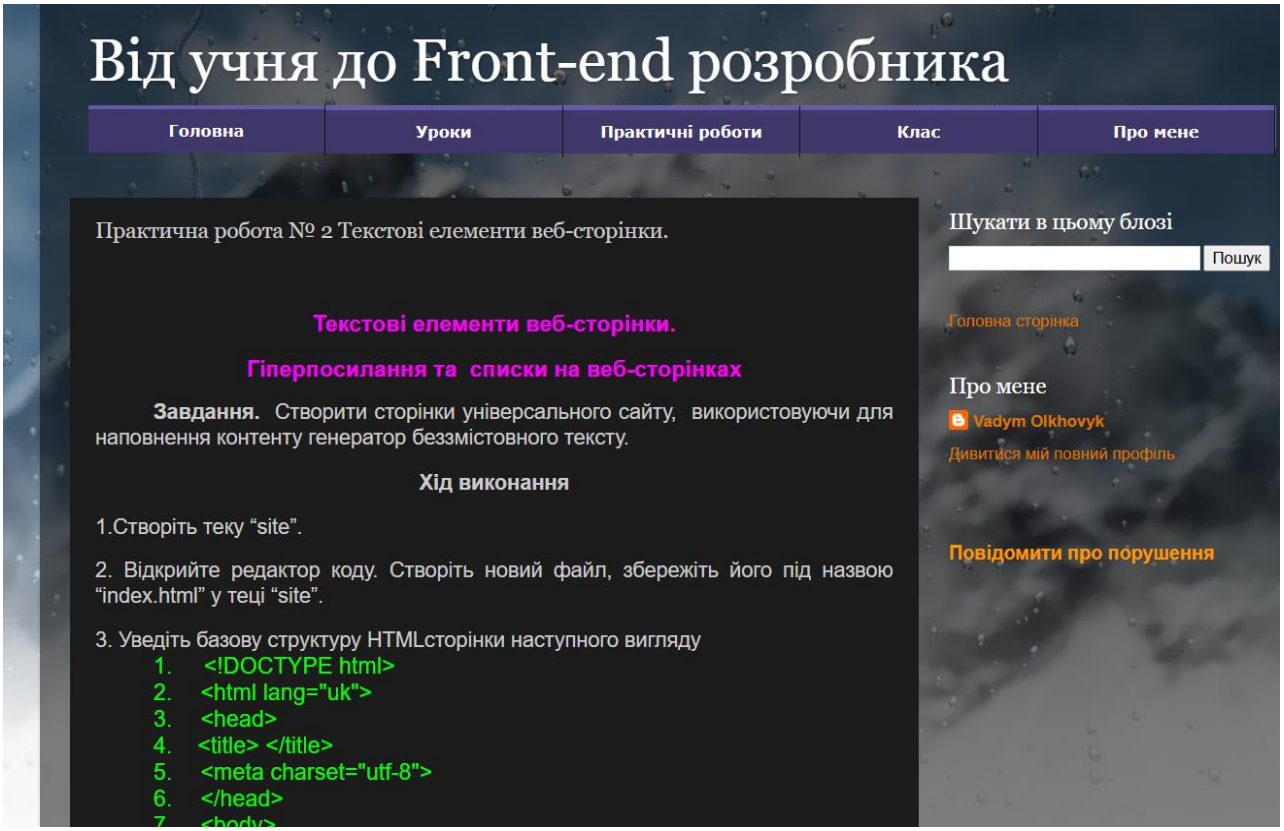
Створений сайт з методичними матеріалами за покликанням https://you-style-html.blogspot.com/

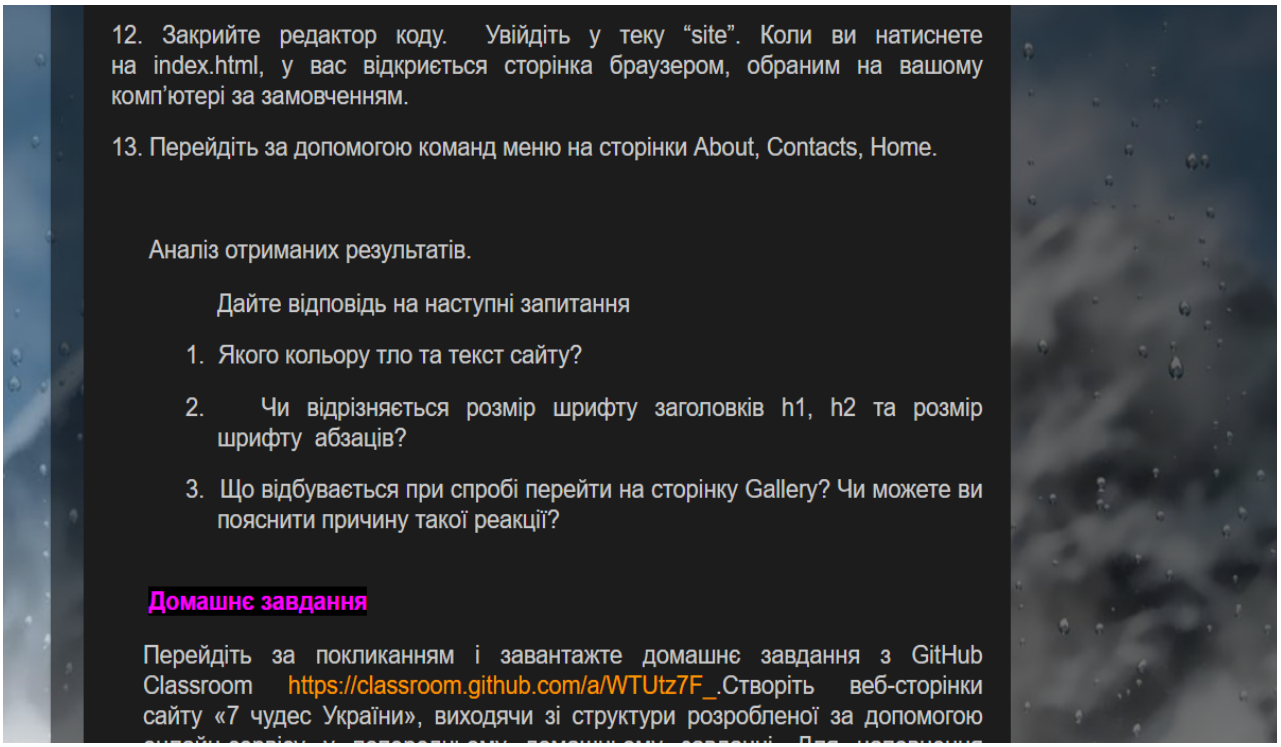




Додаток И

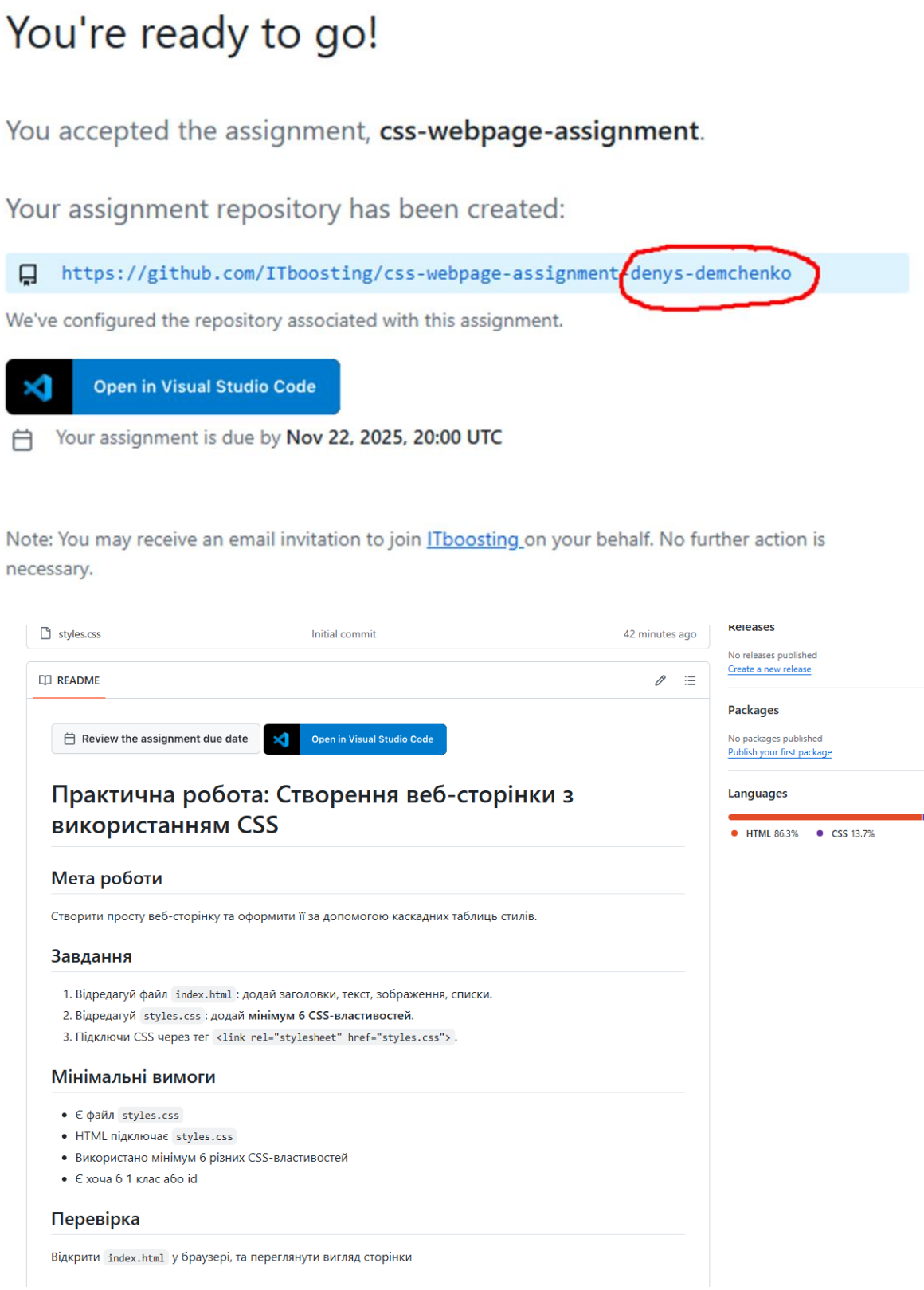
Вигляд сторінок сайту з методичними вказівками





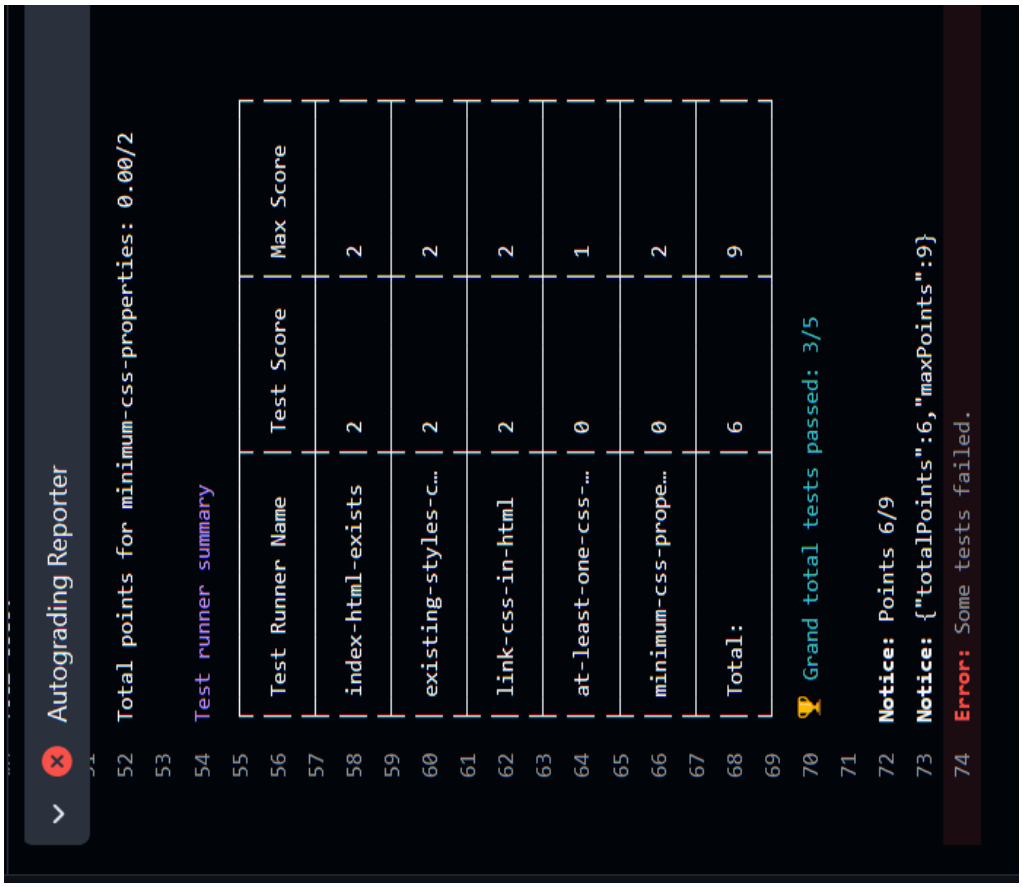
Додаток І

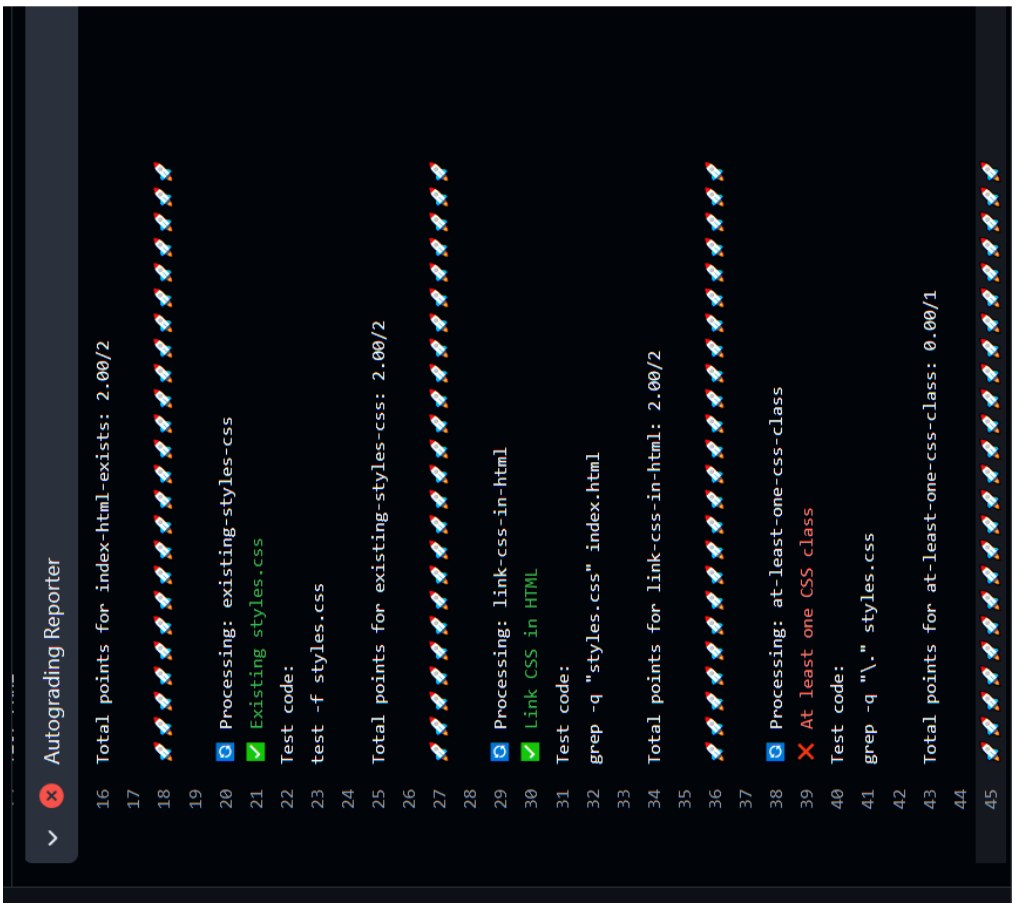
Вигляд завдання та вказівок для студента в GitHub Classroom



Додаток Ї

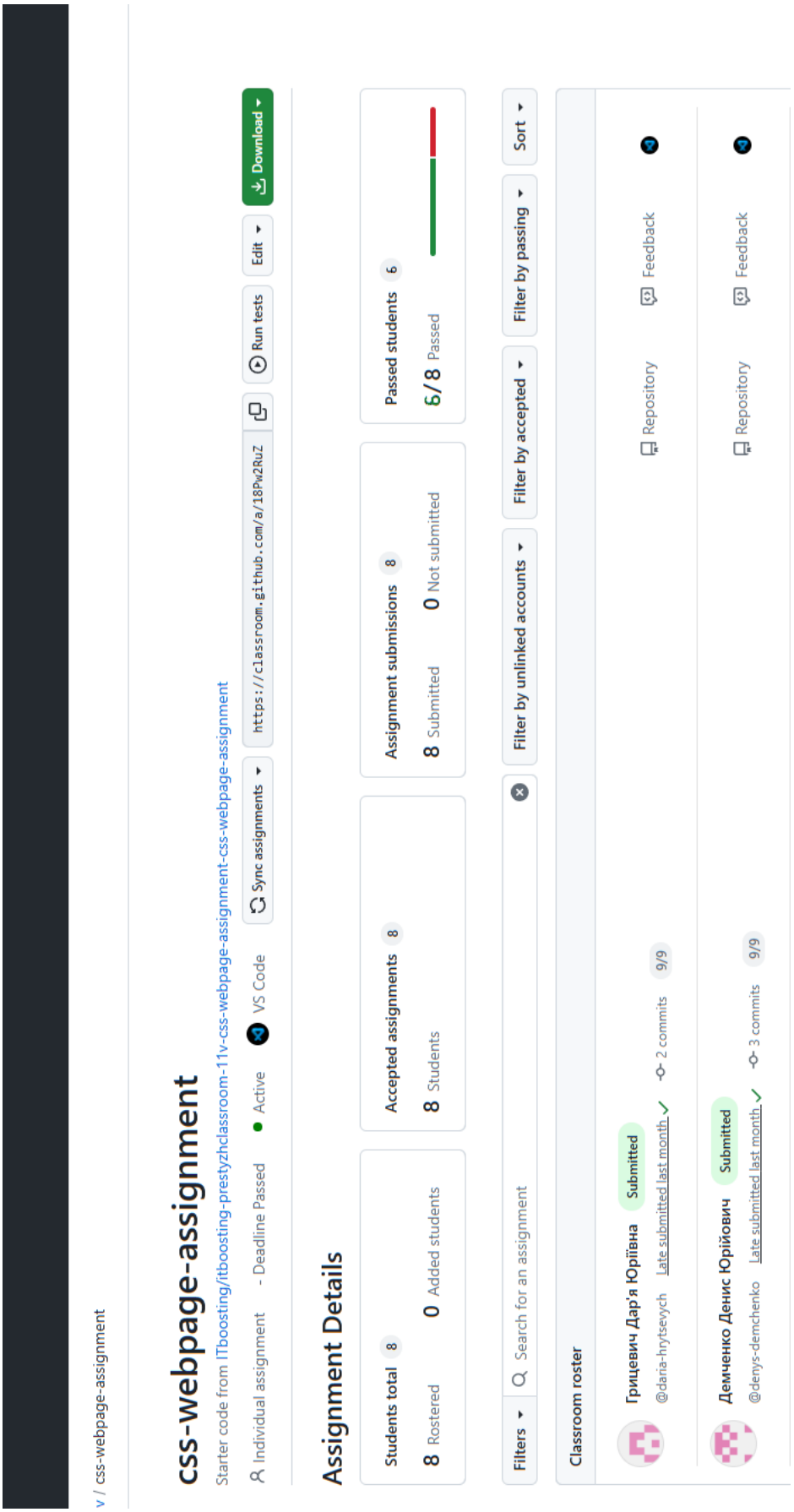
Вигляд процесу автоматизованого тестування коду





Додаток Й

Панель статистики та перевірки виконання завданнь, вигляд для вчителя



Додаток К

ВИКОРИСТАННЯ GIT CLASSROOM ТА ЙОГО API ДЛЯ НАВЧАННЯ ПРОГРАМУВАННЮ УЧНІВ СТАРШИХ КЛАСІВ

Вадим ОЛЬХОВИК,
здобувач другого (магістерського) рівня вищої освіти
Науковий керівник: ***Наталія ПАВЛОВА***
професор, доктор педагогічних наук
Рівненський державний гуманітарний університет

Анотація. *У роботі досліджено використання платформи GitHub Classroom та її API як інструменту для навчання програмуванню учнів старших класів. Проаналізовано переваги Git Classroom у формуванні навичок командної роботи, контролю версій, а також можливості автоматизації освітнього процесу. Запропоновано практичний підхід до впровадження Git Classroom у шкільний курс інформатики.*
Ключові слова: *GitHub Classroom, API, навчання програмуванню, старша школа, цифрова компетентність.*

VADYM OLKHOVYK, NATALIA PAVLOVA. USING GIT CLASSROOM AND ITS API FOR TEACHING PROGRAMMING TO HIGH SCHOOL STUDENTS

Abstract. *The paper explores the use of the GitHub Classroom platform and its API as a tool for teaching programming to high school students. The advantages of Git Classroom in developing teamwork and version control skills, as well as its potential for automating the educational process, are analyzed. A practical approach to integrating Git Classroom into the school computer science curriculum is proposed.*
Keywords: *GitHub Classroom, API, programming education, high school, digital competence.*

Стрімкий розвиток технологій змінює те, як ми навчаємо та вчимося. Програмування стало важливою навичкою для кожного, тому так важливо озброїти учнів старших класів відповідними актуальними інструментами та знаннями. Git Classroom є інноваційною платформою, яка поєднує систему контролю версій з освітніми засобами, надаючи учням та вчителям середовище для співпраці.

Git Classroom побудований на базі Git – системи контролю версій, що широко використовується розробниками професіоналами. Саме Git дозволяє організовувати виконання завдань з програмування у форматі індивідуальної або командної роботи (GitHub документація [1]). Система покращує навчальний процес, дозволяючи

викладачам легко створювати та керувати завданнями, відстежувати прогрес учнів та надавати зворотний зв'язок (Di Stasio 2020 [2]).

У табл. 1 наведено порівняння можливостей та обмежень подібних систем. Всі дані були отримані через відвідування сайтів цих проектів.

Таблиця 1

Порівняння GitHub Classroom з іншими подібними платформами

Критерій	**GitHub Classroom**	**Repl.it Teams for Education**	**Google Classroom + Colab**	**Mimir Classroom**
Платформа	GitHub	Replit	Google	Amazon (раніше незалежна)
Контроль версій (Git)	Вбудований	Обмежений	Через сторонні сервіси	Вбудований
Автоматичне створення репозиторіїв	Так	Немає	Немає	Так
Підтримка API	Потужне API	Немає	Часткова	Частково платна
Автоматичне тестування	Через CI/CD	Вбудоване	Потрібна інтеграція	Вбудоване
Онлайн-кодування	Без компіляції (або через клієнт)	Є	Є (тільки Python)	Є
Підтримка мов програмування	Будь-які (через Git)	Python, Java, JS	Тільки Python	Java, C++, Python
Командна робота	Через Git	Є	Через Workspace	Є
Інтерфейс для викладача	Зручний	Інтуїтивний	Стандартний	Складний, функціональний
Навчальна аналітика	Через API	Мінімальна	Через Google Sheets	Вбудована
Ціна	Безкоштовно	Безкоштовно	Безкоштовно	Платна (частково)

З табл.1 видно, що той час, як Replit Teams більше підходить для молодших учнів, а Mimir Classroom – в якості професійного платного рішення, GitHub Classroom найкращий варіант для викладання реальних IT-практик.

Отже, GitHub Classroom забезпечує, створення приватних репозиторіїв для учнів, централізоване управління завданнями та дедлайнами, доступ до історії змін у проєктах,

автоматизований зворотний зв'язок і перевірку робіт через API (Petrosino, 2019 [3]). Тому добре підходить для викладання актуальних IT практик. Інтеграція API Git Classroom дозволяє викладачеві, створювати репозиторії та налаштовувати завдання автоматично, реалізовувати автоматичне тестування коду (CI), генерувати звіти про активність учнів (GitHub Docs, 2024 [1]).

Випробування сервісу GitHub Classroom було проведено під час навчальної практики на базі 11-го класу. У ході експерименту учні отримували індивідуальні завдання з веб програмування при вивченні HTML/CSS, виконання яких відбувалося у приватних репозиторіях, створених через GitHub Classroom. Результати випробування засвідчили підвищення рівня мотивації учнів, розвиток навичок самостійної роботи та ознайомлення з сучасними інструментами розробки. Крім того, використання GitHub Classroom сприяло формуванню цифрової компетентності та розумінню принципів командної взаємодії у середовищі розробників.

Список використаних джерел

1. GitHub Classroom Documentation. URL: https://classroom.github.com
2. Di Stasio, M., Lombardi, L., & Nanni, U. (2020). *Using GitHub Classroom in Higher Education: An Experience Report*. ACM International Conference on Software Engineering Education and Training.
3. Petrosino, A. J. (2019). *Collaborative Learning in Coding Environments: A Study on GitHub Classroom*. Journal of Computer Science Education, 29(4), 422–439.

Рис. К.1 Тези автора дослідження

Міністерство освіти і науки України
Департамент освіти і науки Рівненської ОДА
Рівненський державний гуманітарний університет

СЕРТИФІКАТ № 2025-062

Учасника
XVIII Всеукраїнської науково-практичної конференції

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРОФЕСІЙНІЙ ДІЯЛЬНОСТІ

10 листопада 2025 року, м. Рівне

Ольховик Вадим

Обсяг - 15 годин (0,5 кредиту ЄКТС)

Завідувачка кафедри цифрових технологій
та методики навчання інформатики РДГУ,
голова програмного комітету конференції — проф. Наталія ПАВЛОВА

Рис. К. 2 Сертифікат учасника конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

РІВНЕНСЬКИЙ ДЕРЖАВНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ

СЕРТИФІКАТ

засвідчує, що

Вадим ОЛЬХОВИК

взяв(ла) участь у роботі

IV Всеукраїнської науково-практичної конференції
“ПІДГОТОВКА ПЕДАГОГІВ ДО ПРОФЕСІЙНОЇ ДІЯЛЬНОСТІ В УМОВАХ ЗМІШАНОГО НАВЧАННЯ”

14 травня 2025 року
м. Рівне

Завідувач кафедри
цифрових технологій та методики
навчання інформатики Рівненського
державного гуманітарного університету

кандидат педагогічних наук,
доцент
Наталія ПАВЛОВА

Рис. К.3 Сертифікат учасника конференції