Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ЦТЕ

Кафедра цифрових технологій в енергетиці

ТЕХНОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	**Перший (бакалаврський)**
Галузь знань	12 Інформаційні технології
Спеціальність	122 Комп'ютерні науки
Освітня програма	Цифрові технології в енергетиці
Статус дисципліни	Нормативна
Форма навчання	Очна (денна)
Рік підготовки, семестр	3 курс осінній семестр
Обсяг дисципліни	5 кредитів ECTS / 150 годин: 72 аудиторні години (36 лекційних, 36 лабораторних), 78 годин СРС
Семестровий контроль/ контрольні заходи	Екзамен, МКР
Розклад занять	https://schedule.kpi.ua/
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: к.т.н., доцент каф. ЦТЕ, Тихоход Володимир Олександрович. Лабораторні роботи: к.т.н., доцент каф. ЦТЕ, Тихоход Володимир Олександрович, асистент, Пасічнюк Антон Олексійович
Розміщення курсу	https://campus.kpi.ua

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

Мета дисципліни — сформувати здатність обґрунтовано вибирати й застосовувати процеси, методи та інструменти розроблення програмного забезпечення, створювати й супроводжувати програмний продукт у команді та забезпечувати його якість упродовж життєвого циклу.

Предмет дисципліни — моделі життєвого циклу, процеси, методи, практики та інструментальні засоби інженерії програмного забезпечення від формування вимог до супроводу й випуску продукту.

В результаті вивчення дисципліни у студентів повинні сформуватися наступні компетентності:

загальні:

- ЗК 01 — здатність до абстрактного мислення, аналізу та синтезу.
- ЗК 02 — здатність застосовувати знання у практичних ситуаціях.
- ЗК 06 — здатність вчитися і оволодівати сучасними знаннями.
- ЗК 09 — здатність працювати в команді.
- ЗК 11 — здатність приймати обґрунтовані рішення.
- ЗК 12 — здатність оцінювати та забезпечувати якість виконуваних робіт.

фахові компетентності:

- ФК 08 — здатність проєктувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування з відповідними моделями, методами, структурами даних і механізмами управління.
- ФК 10 — здатність застосовувати методології, технології та інструментальні засоби для управління процесами життєвого циклу інформаційних і програмних систем, продуктів і сервісів відповідно до вимог замовника.

Очікувані програмні результати навчання:

- ПРН 09 — Розробляти програмні моделі предметних середовищ, вибирати парадигму програмування з позицій зручності та якості застосування для реалізації методів та алгоритмів розв'язання задач в галузі комп'ютерних наук.
- ПРН 11 — Володіти навичками управління життєвим циклом програмного забезпечення, продуктів і сервісів інформаційних технологій відповідно до вимог і обмежень замовника, вміти розробляти проектну документацію (техніко-економічне обґрунтування, технічне завдання, бізнес-план, угоду, договір, контракт).
- ПРН 15 — Застосовувати знання методології та CASE-засобів проектування складних систем, методів структурного аналізу систем, об'єктно-орієнтованої методології проектування при розробці і дослідженні функціональних моделей організаційно-економічних і виробничо-технічних систем.

Після засвоєння навчальної дисципліни студенти мають продемонструвати такі знання та вміння:

знання:

- моделей і процесів життєвого циклу програмного забезпечення, принципів їх вибору та адаптації;
- принципів планово-керованої та гнучкої розробки програмного забезпечення за Waterfall, Scrum і Kanban;
- методів формування, аналізу, документування та керування змінами вимог;
- принципів командної розробки програмного забезпечення та організації спільної роботи з кодом;
- CASE-технологій, UML та підходу «документація як код»;
- основ об'єктно-орієнтованого проєктування, архітектурних рішень, шаблонів проєктування та рефакторингу;
- сучасних мов програмування, технологій та інструментальних засобів розробки програмних систем;
- методів тестування, автоматизованої перевірки та безперервної інтеграції програмного забезпечення;
- методів керування версіями, формування випусків, розгортання та відновлення програмних систем;
- методів документування програмного забезпечення та забезпечення простежуваності проєктних рішень;
- характеристик якості програмного забезпечення відповідно до ISO/IEC 25010 та методів їх оцінювання.

вміння:

- обирати й обґрунтовувати модель життєвого циклу та організовувати процес розробки програмного забезпечення;
- планувати й контролювати виконання робіт із застосуванням засобів керування проєктами;
- працювати в команді, використовувати Git, гілки, Pull Request і Code Review;
- формувати, аналізувати, документувати й трасувати вимоги до програмного забезпечення;
- складати технічне завдання або специфікацію вимог до програмного забезпечення;

- створювати функціональні та об’єктно-орієнтовані моделі із застосуванням CASE-засобів і UML;
- створювати та підтримувати проєктну документацію у графічній і текстовій формах;
- обґрунтовувати вибір парадигми програмування, архітектурного рішення та шаблонів проєктування;
- реалізовувати програмне забезпечення із застосуванням сучасних мов, технологій та інструментальних засобів;
- застосовувати рефакторинг і керувати технічним боргом;
- розробляти автоматизовані тести, аналізувати результати перевірок і керувати дефектами;
- налаштовувати безперервну інтеграцію, автоматизовану перевірку та формування випусків програмного забезпечення;
- оцінювати характеристики якості програмного забезпечення відповідно до ISO/IEC 25010;
- виконувати розгортання, перевірку та відновлення програмного забезпечення після випуску;
- аналізувати результати застосування різних процесів розробки та обґрунтовувати пропозиції щодо їх удосконалення.

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити дисципліни. Вивчення дисципліни спирається на знання, отримані в дисциплінах з попередніх семестрів, зокрема:

«Алгоритмізація та програмування» (1, 2 семестри),

«Об’єктно-орієнтоване програмування» (3 семестр),

«Системи баз даних» (3 семестр),

«Веб-технології та веб-дизайн» (3 семестр).

Постреквізити дисципліни. Матеріал дисципліни може бути використаний при вивченні дисциплін: «Вступ до інтелектуального аналізу даних», «Моделювання систем в енергетиці», «Технології паралельних обчислень в енергетичних комплексах», «Проектування інформаційних систем» та інших, що подаються в наступних семестрах, а також при дипломному проектуванні

3. Зміст навчальної дисципліни

Тема 1. Моделі життєвого циклу та процеси розроблення програмного забезпечення

Тема 2. Інфраструктура командного розроблення

Тема 3. Інженерія вимог і планово-керована розробка

Тема 4. Гнучка розробка програмного забезпечення

Тема 5. Проєктування, конструювання та верифікація програмного забезпечення

Тема 6. Якість, супровід і керування випусками програмного забезпечення

4. Навчальні матеріали та ресурси

Основна література

1. IEEE Computer Society. Guide to the Software Engineering Body of Knowledge (SWEBOK Guide), Version 4.0. — IEEE Computer Society, 2024.

2. Sommerville I. Engineering Software Products: An Introduction to Modern Software Engineering. — Pearson, 2021.

3. Wiegers K., Beatty J. Software Requirements. — 3rd ed. — Microsoft Press, 2013. — 672 p.

4. Мартін Р. Чиста архітектура: мистецтво розроблення програмного забезпечення / пер. з англ. І. Бондар-Терещенко. — Харків : Фабула, 2019. — 368 с.

5. Fowler M. Patterns of Enterprise Application Architecture. — Addison-Wesley Professional, 2002. — 560 p.

Додаткова література

1. Fowler M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. — 3rd ed. — Addison-Wesley, 2003. — 208 p.
2. Farley D. Modern Software Engineering: Doing What Works to Build Better Software Faster. — Addison-Wesley Professional, 2022. — 256 p.
3. Schwaber K., Sutherland J. The Scrum Guide. — 2020. — Режим доступу: https://scrumguides.org/download.html .
4. Chacon S., Straub B. Pro Git. — 2nd ed. — Apress, 2014. — Режим доступу: https://git-scm.com/book/uk/v2 .
5. GitHub Actions Documentation. — Режим доступу: https://docs.github.com/actions .
6. Seemann M., van Deursen S. Dependency Injection Principles, Practices, and Patterns. — Manning, 2019. — 552 p.
7. Meszaros G. xUnit Test Patterns: Refactoring Test Code. — Addison-Wesley Professional, 2007. — 833 p.
8. Лавріщева К.М. Програмна інженерія. — К. — 2008. — 319 с.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Тема 1. Моделі життєвого циклу та організація процесу розроблення

Лекція 1. Моделі життєвого циклу та вибір процесу

Послідовна, інкрементна й еволюційна стратегії розроблення. Каскадна модель, V-модель, інкрементна і спіральна моделі, прототипування, швидка та компонентно-орієнтована розробка. Планово-керовані, гнучкі й гібридні процеси. Вибір і адаптація процесу з урахуванням стабільності вимог, ризиків, критичності, розміру проєкту та характеру постачання.

Лекція 2. Командна інфраструктура та безперервна інтеграція

Розподілене керування версіями із застосуванням Git. Гілки, інтеграція змін, запити на злиття, перевірка коду та підтвердження індивідуального внеску. Структура репозиторію і створення каркаса продукту з навчального шаблону. Базовий конвеєр безперервної інтеграції: відновлення залежностей, збирання та запуск тестів.

Тема 2. Планово-керована розробка програмного забезпечення

Лекція 3. Інженерія вимог у планово-керованому процесі

Виявлення, аналіз, специфікація та перевірка вимог. Функціональні та нефункціональні вимоги. Огляд характеристик якості за ISO/IEC 25010 як основа формування нефункціональних вимог. Формулювання вимірюваних критеріїв приймання. Технічне завдання або SRS, базова лінія вимог, матриця простежуваності, контроль і оцінювання змін.

Лекція 4. Проєктування програмного забезпечення та CASE-засоби

Перехід від вимог до проєктного рішення. Функціональні моделі та UML-моделі структури й поведінки. Застосування CASE-засобів. Планування послідовних стадій та визначення критеріїв їх завершення.

Лекція 5. Реалізація та інтеграція програмного забезпечення

Організація реалізації відповідно до затвердженої специфікації. Послідовна розробка та інтеграція API, предметної логіки, доступу до даних і клієнтського застосунку. Контроль відхилень від проєктного рішення.

Лекція 6. Перевірка та випуск програмного забезпечення

Верифікація відповідності специфікації, приймальне тестування і робота з дефектами. Версіонування, підготовка приміток до випуску, формування випускного артефакту та демонстраційне розгортання.

Тема 3. Гнучка розробка програмного забезпечення за Scrum

Лекція 7. Перехід до Scrum і організація спринту

Керування змінами вимог після випуску першої версії. Команда, події й артефакти Scrum. Журнал вимог, користувацькі історії, критерії приймання, мета і план спринту, критерії готовності, огляд і ретроспектива.

Лекція 8. Еволюційна архітектура та документування рішень

Архітектурно значущі вимоги, архітектурні стилі, компоненти та їх взаємодія. Порівняння альтернатив, записи архітектурних рішень, еволюція моделей і керування технічним боргом.

Тема 4. Предметна логіка, проєктування та автоматизоване тестування

Лекція 9. Предметна логіка та доступ до даних

Процедурний, об'єктно-орієнтований, функціональний і поєднаний способи організації предметної логіки. Сценарій транзакції, модель предметної області, репозиторій, перетворювач даних, одиниця роботи та впровадження залежностей.

Лекція 10. Автоматизоване тестування предметної логіки

Модульне та інтеграційне тестування. Побудова тесту за схемою «підготовка — дія — перевірка», параметризовані тести, тестові замінники, перевірка предметних правил і доступу до даних. Цикл розроблення через тестування.

Лекція 11. Шаблони проєктування

Породжувальні, структурні й поведінкові шаблони GoF. Виявлення проблеми проєктування, порівняння альтернатив і вибір шаблону відповідно до характеру мінливості програмного коду.

Лекція 12. Безпечний рефакторинг і завершення спринту

Захисні тести, виконання рефакторингу малими кроками та синхронізація моделей із програмним кодом. Перевірка інкременту, завершення спринту та формування випуску версії 2.0.

Тема 5. Керування потоком робіт та автоматизація перевірок

Лекція 13. Керування потоком робіт за методом Kanban

Візуалізація процесу, витягувальна система, явні правила, обмеження незавершених завдань, класи обслуговування та опрацювання блокувань. Тривалість виконання, повний час проходження і пропускна здатність.

Лекція 14. Стратегія тестування та автоматизація перевірок

Формування стратегії тестування відповідно до ризиків програмного продукту. Піраміда тестів. Поєднання модульних, інтеграційних, контрактних і наскрізних тестів. Регресійне тестування. Відтворення дефекту автоматизованим тестом перед його виправленням. Автоматичний запуск перевірок у конвеєрі безперервної інтеграції, аналіз результатів і критерії проходження перевірок.

Тема 6. Забезпечення якості, стабілізація та випуск програмного забезпечення

Лекція 15. Якість програмного забезпечення та статичний аналіз

Характеристики якості за ISO/IEC 25010. Статичний аналіз коду, виявлення дефектів і технічного боргу. Порогові критерії якості, визначення пріоритетів з урахуванням ризиків і керування потоком виправлень.

Лекція 16. Нефункціональне тестування та стабілізація

Перевірка продуктивності, надійності, безпеки та інших пріоритетних характеристик якості. Регресійне тестування, стабілізація і формування кандидата на випуск.

Лекція 17. Керування випусками та порівняння процесів

Визначення готовності до випуску, рішення про випуск або його відкладення, семантичне версіонування, примітки до випуску, незмінний артефакт, розгортання, базова перевірка працездатності та повернення до попередньої версії. Порівняння каскадного процесу, Scrum, Kanban і гібридних процесів.

Модульна контрольна робота (2 год.)

6. Лабораторні заняття

Лабораторний практикум охоплює 36 аудиторних годин і складається з дев'яти лабораторних робіт по 4 години. Один продукт послідовно проходить планово-керований етап Waterfall, гнучкий розвиток за Scrum і стабілізацію потоку робіт за Kanban. Така організація дає змогу порівняти процеси на власному досвіді, не створюючи трьох незалежних навчальних проєктів.

До складу продукту входять API, серверна реалізація, база даних і простий клієнтський застосунок. Клієнт може бути консольним, настільним, простим вебклієнтом або іншим відомим команді застосунком. Він забезпечує демонстрацію завершених користувацьких сценаріїв і не створює окремої лінії вивчення складного клієнтського фреймворку.

№	Лабораторна робота	Передумови	Основний результат	Год.
1	Вибір процесу та підготовка інфраструктури програмного продукту	Лекції 1–4	Порівняння процесів, обґрунтований вибір процесу, сформована команда, репозиторій, каркас програмного інтерфейсу (API) і клієнтського застосунку, налаштовані Git та безперервна інтеграція (CI).	4
2	Каскадна модель (Waterfall): специфікація вимог і проєктування першої версії	Лекції 5–6	Специфікація вимог до програмного забезпечення (SRS), базова лінія вимог, матриця простежуваності, функціональна та UML-моделі, послідовний план розроблення і критерії завершення стадій.	4
3	Каскадна модель (Waterfall): реалізація, перевірка і випуск версії 1.0	Лекція 7	Реалізація відповідно до зафіксованої специфікації, перевірка програмного забезпечення, працездатний клієнтський сценарій, позначка версії в репозиторії, примітки до випуску та випуск версії 1.0.	4
4	Scrum, спринт 1: розвиток архітектури	Лекції 8–10	Перехід до журналу вимог до продукту (Product Backlog), визначення мети спринту (Sprint Goal), архітектурна зміна, запис архітектурного рішення (ADR), завершений інкремент, огляд результатів і ретроспектива спринту.	4
5	Scrum, спринт 2: розвиток предметної логіки та доступу до даних	Лекція 11	Реалізована користувацька історія (User Story), що охоплює предметні правила, доступ до даних, автоматизовані тести та операцію клієнтського застосунку.	4
6	Scrum, спринт 3: шаблони проєктування, рефакторинг і випуск версії 2.0	Лекція 12	Обґрунтовано застосований шаблон проєктування GoF, створені захисні тести, виконаний рефакторинг, актуалізована модель програмного забезпечення та сформований випуск версії 2.0.	4
7	Kanban: організація потоку робіт та автоматизація тестування	Лекції 13–14	Дошка Kanban, обмеження кількості незавершених завдань (WIP), класи обслуговування, показники потоку, автоматизовані тести, відтворення дефекту регресійним тестом та його подальше виправлення.	4
8	Kanban: аналіз якості та стабілізація програмного забезпечення	Лекції 15–16	Організований потік усунення дефектів і технічного боргу, виконані статичний аналіз та нефункціональне тестування, установлені порогові критерії якості (quality gates), сформований кандидат на випуск (release candidate).	4
9	Фінальний випуск і порівняння процесів розроблення	Лекція 17	Ухвалено рішення про випуск або його відкладення (Go/No-Go), визначено номер версії, сформовано незмінний випускний артефакт, виконано розгортання, базову перевірку працездатності (smoke test) і повернення до попередньої версії (rollback); порівняно каскадний процес, Scrum і Kanban.	4

7. Самостійна робота студента

Тема 1. Моделі життєвого циклу та вибір процесу розроблення програмного забезпечення.

Тема 2. Засоби керування версіями, командна робота та безперервна інтеграція.

Тема 3. Інженерія вимог, CASE-засоби, UML-моделювання та проєктна документація.

Тема 4. Організація гнучкої розробки за Scrum та еволюція архітектури програмного забезпечення.

Тема 5. Шаблони проєктування, рефакторинг та автоматизоване тестування.

Тема 6. Керування потоком робіт за Kanban, забезпечення якості та формування випусків.

Політика та контроль

8. Політика навчальної дисципліни (освітнього компонента)

Лабораторні роботи виконуються командами у складі 3–4 студентів у межах одного наскрізного програмного проєкту. У ЛР №1–3 застосовується каскадний процес (Waterfall), у ЛР №4–6 — Scrum, у ЛР №7–8 — Kanban. ЛР №9 завершує розроблення програмного продукту та передбачає порівняння застосованих процесів.

Після ЛР №1 у кожній наступній роботі застосовуються система керування версіями Git, запити на злиття змін, перевірка коду та безперервна інтеграція і постачання (CI/CD).

Командний результат не передбачає автоматичного зарахування роботи всім учасникам. Кожен студент повинен мати визначені завдання, підтверджений внесок у спільний результат і бути здатним пояснити прийняті рішення. Упродовж семестру студенти виконують і захищають дев’ять лабораторних робіт.

Кожен функціональний інкремент повинен містити працездатний користувацький сценарій. Для його демонстрації може використовуватися консольний, настільний, простий вебклієнт або інший відомий команді клієнтський застосунок. Оцінюються взаємодія з програмним інтерфейсом (API), правильність оброблення даних і помилок та завершеність сценарію. Візуальне оформлення клієнтського застосунку окремо не оцінюється.

Шаблони проєктування застосовуються для розв’язання виявлених проблем у програмному коді. Додавання шаблонів, архітектурних шарів, інтерфейсів або діаграм без належного обґрунтування не підвищує оцінку.

Критерії готовності результату уточнюються відповідно до опрацьованих тем. Робота, подана після встановленого строку без поважної причини, може бути оцінена не більше ніж у 80 % від максимальної кількості балів.

Копіювання результатів між командами та подання чужого внеску як власного є порушенням академічної доброчесності. Використання генеративного штучного інтелекту не звільняє студента від перевірки отриманих результатів, розуміння поданих матеріалів і пояснення прийнятих рішень.

Політика та принципи академічної доброчесності визначені Кодексом честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»: https://kpi.ua/code .

Норми етичної поведінки студентів і працівників визначені Кодексом честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»: https://kpi.ua/code .

Передбачена можливість зарахування результатів навчання за окремими темами дисципліни за результатами успішного проходження онлайн-курсів. Визнання таких результатів здійснюється відповідно до «Положення про визнання в КПІ ім. Ігоря Сікорського результатів навчання, набутих у неформальній/інформальній освіті» (https://osvita.kpi.ua/node/179).

9. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Система рейтингових (вагових) балів та критерії оцінювання

1) Лабораторні роботи

За виконання та захист дев’яти лабораторних робіт студент може отримати до 45 балів.

№	Лабораторна робота	Результат, що оцінюється	Командний результат, до 3,5 бала	Індивідуальний внесок і захист, до 1,5 бала	Усього
1	Вибір процесу та підготовка інфраструктури програмного продукту	Обґрунтований вибір процесу; створені репозиторій, структура проєкту, програмний інтерфейс і клієнтський застосунок; налаштована безперервна інтеграція	Обґрунтованість вибору процесу — 1,0; працездатність початкової структури — 1,5; налаштування репозиторію та автоматизованого збирання — 1,0	Підтверджений внесок — 0,5; пояснення прийнятих рішень — 1,0	5
2	Каскадна модель: специфікація вимог і проєктування першої версії	Специфікація вимог, базова лінія, матриця простежуваності, функціональна та UML-моделі, план послідовних стадій	Повнота й узгодженість вимог — 1,0; якість моделей — 1,0; простежуваність і планування — 1,0; застосування CASE-засобу — 0,5	Підтверджений внесок — 0,5; пояснення вимог і моделей — 1,0	5
3	Каскадна модель: реалізація, перевірка і випуск версії 1.0	Реалізована версія відповідно до специфікації, результати перевірки, клієнтський сценарій і сформований випуск	Відповідність специфікації — 1,0; працездатність реалізації — 1,0; перевірка вимог — 0,75; оформлення випуску — 0,75	Підтверджений внесок — 0,5; демонстрація і пояснення реалізації — 1,0	5
4	Scrum, спринт 1: розвиток архітектури	Журнал вимог до продукту, мета спринту, архітектурна зміна, запис архітектурного рішення, завершений інкремент, огляд і ретроспектива	Планування спринту — 0,75; обґрунтованість архітектурної зміни — 1,0; завершеність інкременту — 1,0; аналіз результатів спринту — 0,75	Підтверджений внесок — 0,5; пояснення архітектурного рішення — 1,0	5
5	Scrum, спринт 2: розвиток предметної логіки та доступу до даних	Завершена користувацька історія, предметні правила, доступ до даних, автоматизовані тести та клієнтська операція	Правильність предметної логіки — 1,0; реалізація доступу до даних — 0,75; працездатність сценарію — 1,0; якість тестів — 0,75	Підтверджений внесок — 0,5; пояснення програмного рішення — 1,0	5
6	Scrum, спринт 3: шаблони проєктування,	Обґрунтований шаблон проєктування, захисні тести,	Доцільність шаблону — 1,0; безпечність рефакторингу — 1,0; узгодженість моделі	Підтверджений внесок — 0,5; пояснення проблеми та	5

	рефакторинг і випуск версії 2.0	рефакторинг, актуалізована модель і сформований випуск	з кодом — 0,75; оформлення випуску — 0,75	способу її розв’язання — 1,0	
7	Kanban: організація потоку робіт та автоматизація тестування	Налаштована дошка, обмеження незавершених завдань, показники потоку, автоматизовані тести та виправлений дефект	Організація потоку робіт — 1,0; обґрунтованість обмежень — 0,75; збирання й аналіз показників — 0,75; автоматизоване виправлення дефекту — 1,0	Підтверджений внесок — 0,5; пояснення організації потоку і тестів — 1,0	5
8	Kanban: аналіз якості та стабілізація програмного забезпечення	Проаналізовані дефекти й технічний борг, виконані статичний аналіз і нефункціональне тестування, оцінена якість за ISO/IEC 25010, сформований кандидат на випуск	Організація потоку виправлень — 0,75; результати аналізу і тестування — 1,0; оцінювання характеристик якості — 1,0; обґрунтованість кандидата на випуск — 0,75	Підтверджений внесок — 0,5; пояснення результатів оцінювання якості — 1,0	5
9	Фінальний випуск і порівняння процесів розроблення	Ухвалене рішення про випуск, сформований артефакт, виконані розгортання і перевірка, підготовлений план відновлення, порівняні застосовані процеси	Готовність і відтворюваність випуску — 1,0; розгортання та перевірка — 0,75; план відновлення — 0,75; доказовість порівняння процесів — 1,0	Підтверджений внесок — 0,5; демонстрація результату і захист висновків — 1,0	5
	Разом		**31,5**	**13,5**	**45**

Командний результат становить 70 % оцінки за лабораторну роботу, індивідуальний внесок і захист — 30 %. Бали за окремими критеріями можуть виставлятися з точністю до 0,25 бала. Непредставлений або непрацездатний результат оцінюється в 0 балів до повторного захисту. Якщо студент не підтвердив власний внесок або не може пояснити прийняті рішення, за індивідуальну складову виставляється 0 балів.

Для кожної роботи результат першого етапу може бути зафіксований як проміжна контрольна точка, але окрема лабораторна оцінка виставляється за завершений результат. Непредставлений або непрацездатний результат оцінюється у 0 балів до повторного захисту.

2) Календарний контроль

Проводиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу.

Для першого календарного контролю студент повинен захистити ЛР №1–3, отримати позитивну оцінку з тесту та отримати не менше 50 % балів, запланованих календарем на цей момент.

Для другого календарного контролю студент повинен захистити ЛР №1–6 та мати поточний рейтинг не менше 25 балів.

Поточне тестування

Поточне тестування проводиться на сьомому-восьмому тижні навчання та охоплює матеріал, опрацьований на цей момент. Максимальна оцінка становить 5 балів і визначається пропорційно до частки правильних відповідей.

3) Модульна контрольна робота

Модульна контрольна робота (МКР) проводиться в кінці семестру з метою контролю оцінки отриманих знань.

Максимальна кількість балів за МКР дорівнює 10 балів.

Якість виконання роботи оцінюється за наступними критеріями:

- усі відповіді вірні та повні –10 балів,
- у відповідях допущені несуттєві неточності – 8 балів,
- половина відповідей вірна – 5 балів,
- відповіді з суттєвими неточностями, але без критичних помилок – 2 бали,
- менше половини відповідей вірна – 0 балів.

Можливе також проведення модульної контрольної роботи у формі тестування, в такому випадку студент отримує бал, який відповідає частці правильних відповідей тесту з розрахунку, що 10 балів становлять 100%.

4) Складання іспиту

Умови допуску до *іспиту*

Необхідною умовою допуску до іспиту є зарахування всіх шести лабораторних робіт, виконання модульної контрольної роботи та стартовий рейтинг (Rc) не менше 40 балів.

Максимальний ваговий бал $r_{ісп}$=40

На іспиті студент виконує письмову контрольну роботу, яка містить два теоретичних питання і одне практичне питання. Теоретичні питання оцінюються максимально по 10 балів, практичне – 20 балів.

Теоретична частина оцінюється таким чином:

- правильна чітко викладена, повна відповідь – (не менше 90% потрібної інформації) – 9-10 балів;
- достатньо повна відповідь (не менше 75% потрібної інформації) – 7.5-8.9 балів;
- неповна відповідь (не менше 60% потрібної інформації) – 6-7.4 балів;
- незадовільна відповідь – 0 балів.

Практичне завдання оцінюється таким чином:

- повне, безпомилкове розв'язання завдання — 18–20 балів;
- повне розв'язання завдання з несуттєвими невідповідностями — 15–17 балів;
- завдання виконано частково або з суттєвими недоліками — 12–14 балів;
- завдання не виконано або результат не відповідає умові — 0–11 балів.

5) Розрахунок шкали (R) рейтингу:

Сума вагових балів контрольних заходів протягом семестру (шкала рейтингу) складає:

$$R = r_{практ} + r_{тест} + r_{мод} + r_{ісп} = 45 + 5 + 10 + 40 = 100 \text{ балів.}$$

Максимальний стартовий рейтинг становить $R_c = r_{практ} + r_{тест} + r_{мод} = 60$ балів.

Рейтинг іспиту дорівнює 40 балів. Мінімальний рейтинг допуску до іспиту становить 40 балів.

Таким чином, рейтингова шкала з кредитного модуля складає

$$R = 60+40=100 \text{ балів.}$$

Для отримання студентом відповідних оцінок рейтингова оцінка студента переводиться згідно таблиці:

Бали	**Оцінка**
95 - 100	Відмінно
85 - 94	Дуже добре
75 - 84	Добре
65 - 74	Задовільно

60 - 64	Достатньо
Менше 60	Незадовільно
R < 40, є незараховані лабораторні роботи або не виконані інші умови допуску до екзамену	Не допущено

Робочу програму навчальної дисципліни (силабус):

Складено к.т.н., доцент каф. ЦТЕ, Тихоход Володимир Олександрович

Ухвалено кафедрою ЦТЕ (протокол № 22 від 17.06.26)

Погоджено науково-методичною комісією НН ІАТЕ КПІ ім. Ігоря Сікорського (протокол № 9 від 26.06.2026 р.)