УКРАЇНА
ЖИТОМИРСЬКИЙ КООПЕРАТИВНИЙ ФАХОВИЙ
КОЛЕДЖ БІЗНЕСУ І ПРАВА

ЗАТВЕРДЖУЮ
Директор Житомирського
кооперативного фахового коледжу
бізнесу і права

Олена СИЛАНТЬЄВА
«30» серпня 2024 р.

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Навчальна практика на умовно робочих місцях

Освітньо-професійний ступінь	*Фаховий молодший бакалавр*
Галузь знань	*12 «Інформаційні технології»*
Спеціальність	*121 «Інженерія програмного забезпечення»*
Освітньо-професійна програма	*«Інженерія програмного забезпечення»*

Житомир 2024 р.

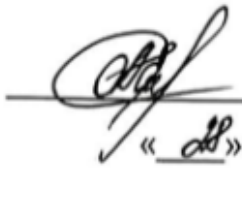
Навчальна практика на умовно робочих місцях. Програма обов’язкової навчальної дисципліни для підготовки фахових молодших бакалаврів у ЖКФКБП. Спеціальність: 121 Інженерія програмного забезпечення. Освітньо-професійна програма: Інженерія програмного забезпечення.

Укладач: Панасенко М.Д. — викладач

Розглянуто на засіданні циклової комісії
інженерії програмного забезпечення та математики
Протокол № 11 від 07 червня 2024 р.
Голова комісії _____ Катерина СЛІДЗЕВСЬКА

Розглянуто на засіданні Методичної ради
Протокол № 7 від «____» 27.06.2024 р.

УКРАЇНА
ЖИТОМИРСЬКИЙ КООПЕРАТИВНИЙ ФАХОВИЙ
КОЛЕДЖ БІЗНЕСУ І ПРАВА

ЗАТВЕРДЖУЮ
Директор Житомирського
кооперативного фахового коледжу
бізнесу і права

Олена СИЛАНТЬЄВА
«28» серпня 2025 р.

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Навчальна практика на умовно робочих місцях

Освітньо-професійний ступінь	*Фаховий молодший бакалавр*
Галузь знань	*F «Інформаційні технології»*
Спеціальність	*F2 «Інженерія програмного забезпечення»*
Освітньо-професійна програма	*«Інженерія програмного забезпечення»*

Житомир 2025 р.

Навчальна практика на умовно робочих місцях. Програма обов’язкової навчальної дисципліни для підготовки фахових молодших бакалаврів у ЖКФКБП. Спеціальність: F2 Інженерія програмного забезпечення. Освітньо-професійна програма: Інженерія програмного забезпечення.

Укладач: Панасенко М.Д. — викладач

Розглянуто на засіданні циклової комісії
інженерії програмного забезпечення та математики
Протокол № 11 від 07 червня 2025 р.
Голова комісії _____ Дмитро ОЛЕНЮК

Розглянуто на засіданні Методичної ради
Протокол № 7 від «____» 27.06.2025 р.

УКРАЇНА
ЖИТОМИРСЬКИЙ КООПЕРАТИВНИЙ ФАХОВИЙ
КОЛЕДЖ БІЗНЕСУ І ПРАВА

ЗАТВЕРДЖУЮ
Директор Житомирського
кооперативного фахового коледжу
бізнесу і права
Олена СИЛАНТЬЄВА
«29» серпня 2026 р.

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Навчальна практика на умовно робочих місцях

Освітньо-професійний ступінь	*Фаховий молодший бакалавр*
Галузь знань	*F «Інформаційні технології»*
Спеціальність	*F2 «Інженерія програмного забезпечення»*
Освітньо-професійна програма	*«Інженерія програмного забезпечення»*

Житомир 2026 р.

Навчальна практика на умовно робочих місцях. Програма обов’язкової навчальної дисципліни для підготовки фахових молодших бакалаврів у ЖКФКБП. Спеціальність: F2 Інженерія програмного забезпечення. Освітньо-професійна програма: Інженерія програмного забезпечення.

Укладач: Степанов Дмитро Олександрович — спеціаліст

Розглянуто на засіданні циклової комісії
інженерії програмного забезпечення та математики
Протокол № 11 від 07 червня 2026 р.
Голова комісії _____ Наталія ГОНГАЛО

Розглянуто на засіданні Методичної ради
Протокол № 7 від «____» 27.06.2026 р.

ЗМІСТ

1. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ 8

2. МЕТА І ЗАВДАННЯ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ 9

3. ПЕРЕЛІК ЗАГАЛЬНИХ І СПЕЦІАЛЬНИХ КОМПЕТЕНТНОСТЕЙ 11

4. НОРМАТИВНИЙ ЗМІСТ ПІДГОТОВКИ ЗДОБУВАЧІВ ФАХОВОЇ ПЕРЕДВИЩОЇ ОСВІТИ, СФОРМУЛЬОВАНИЙ У ТЕРМІНАХ РЕЗУЛЬТАТІВ НАВЧАННЯ 13

5. ІНФОРМАЦІЙНИЙ ОБСЯГ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ 15

6. ЗМІСТ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ 16

7. ТЕМИ ПРАКТИЧНИХ ЗАНЯТЬ 22

8. САМОСТІЙНА РОБОТА 24

9. ІНДИВІДУАЛЬНЕ ЗАВДАННЯ 26

10. МЕТОДИ НАВЧАННЯ, КОНТРОЛЮ ТА ЗАСОБИ ОЦІНЮВАННЯ 28

11. ПОРЯДОК ТА КРИТЕРІЇ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ 30

12. СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ 33

1. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

<table>
<tr><td rowspan="2">Характеристика навчальної дисципліни</td><td rowspan="2">Галузь знань, спеціальність, освітньо-професійна програма, освітньо-професійний ступінь</td><td colspan="8">Розподіл годин</td></tr>
<tr><td colspan="8">Семестри</td></tr>
<tr><td rowspan="11">Обов’язкова

Загальний обсяг:
Годин — 90
Кредитів — 3

Тижневих годин:
Аудиторні — 36
(3 тижні по 12 годин)

Індивідуальне завдання —
передбачено навчальним планом</td><td rowspan="11">Галузь знань:
F «Інформаційні технології»

Спеціальність:
F2 «Інженерія програмного забезпечення»

Освітньо-професійна програма:
«Інженерія програмного забезпечення»

Освітньо-професійний ступінь: фаховий молодший бакалавр</td><td>1-й</td><td>2-й</td><td>3-й</td><td>4-й</td><td>5-й</td><td>6-й</td><td>7-й</td><td>8-й</td></tr>
<tr><td colspan="8">Лекції</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td colspan="8">Практичні заняття</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>36</td></tr>
<tr><td colspan="8">Семінарські заняття</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td colspan="8">Лабораторні заняття</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td colspan="8">Самостійна робота</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>54</td></tr>
<tr><td colspan="8">Індивідуальні завдання</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>+</td></tr>
<tr><td colspan="8">Форма підсумкового контролю</td></tr>
<tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>ДЗ</td></tr>
</table>

Форма підсумкового контролю — диференційований залік.

2. МЕТА І ЗАВДАННЯ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Мета навчальної дисципліни: формування у здобувачів освіти практичних навичок професійної діяльності фахівця з інженерії програмного забезпечення в умовах, наближених до реальних умов роботи ІТ-компанії: виконання професійних обов'язків на умовно робочих місцях у складі команди розробників, проходження основних етапів життєвого циклу програмного забезпечення — від аналізу вимог до тестування, документування та презентації завершеного програмного продукту — із застосуванням сучасних інструментів командної роботи та гнучких методологій розроблення.

Завдання навчальної дисципліни:

– ознайомити здобувачів освіти з організаційною структурою ІТ-компанії, основними професійними ролями та посадовими обов'язками фахівців у сфері розроблення програмного забезпечення;
– сформувати навички організації умовно робочого місця фахівця з інженерії програмного забезпечення з дотриманням вимог охорони праці та техніки безпеки;
– опанувати практичне застосування гнучких методологій розроблення (Scrum, Kanban) та планування роботи команди;
– закріпити навички роботи з системою контролю версій Git, платформою GitHub та трекерами задач у процесі командного розроблення;
– сформувати вміння збирати, аналізувати та документувати вимоги до програмного забезпечення, складати технічне завдання;
– відпрацювати базові прийоми проєктування програмного продукту: побудову UML-діаграм, проєктування бази даних та макетів інтерфейсу користувача;
– сформувати навички командного розроблення програмного продукту з дотриманням стандартів кодування та проведенням перевірки коду (code review);

– опанувати основи тестування програмного забезпечення: складання тест-кейсів, оформлення звітів про дефекти, модульне тестування;
– сформувати вміння документувати програмний продукт, готувати його до розгортання та передавання замовнику;
– розвинути навички ділової комунікації, роботи в команді, самоорганізації та презентації результатів власної роботи.

Предметом вивчення навчальної дисципліни є професійна діяльність фахівця з інженерії програмного забезпечення в умовах, що моделюють роботу ІТ-компанії: організація робочого місця, командна взаємодія, аналіз вимог, проєктування, розроблення, тестування, документування та супровід програмного продукту із застосуванням сучасних інструментальних засобів і гнучких методологій.

Міждисциплінарні зв'язки: «Основи програмування», «Об'єктно-орієнтоване програмування», «Алгоритми та структури даних», «Бази даних», «Операційні системи», «Організація комп'ютерних мереж», «Компоненти програмної інженерії», «Програмування веб-застосувань», «Групова динаміка та комунікації», «Екологія, безпека життєдіяльності та охорона праці», «Проєктний практикум».

3. ПЕРЕЛІК ЗАГАЛЬНИХ І СПЕЦІАЛЬНИХ КОМПЕТЕНТНОСТЕЙ

Перелік загальних (ЗК) і спеціальних (СК) компетентностей випускника відповідно до Освітньо-професійної програми підготовки фахового молодшого бакалавра зі спеціальності F2 «Інженерія програмного забезпечення»:

- ЗК 01. Здатність реалізувати свої права і обов'язки як члена суспільства, усвідомлювати цінності громадянського (вільного демократичного) суспільства та необхідність його сталого розвитку, верховенства права, прав і свобод людини і громадянина в Україні.
- ЗК 02. Здатність зберігати та примножувати моральні, культурні, наукові цінності і досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про природу і суспільство та у розвитку суспільства, техніки і технологій, використовувати різні види та форми рухової активності для активного відпочинку та ведення здорового способу життя.
- ЗК 03. Здатність спілкуватися державною мовою як усно, так і письмово.
- ЗК 04. Здатність спілкуватися іноземною мовою.
- ЗК 05. Знання та розуміння предметної області та розуміння професійної діяльності.
- ЗК 06. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
- ЗК 07. Здатність застосовувати знання у практичних ситуаціях.
- СК 01. Здатність алгоритмічно та логічно мислити.
- СК 02. Здатність вдосконалювати знання і навички в галузі інформаційних технологій та усвідомлення важливості навчання протягом усього життя.
- СК 03. Здатність застосовувати теоретичні та емпіричні знання для розроблення, тестування, впровадження та супроводу програмного забезпечення.

– СК 04. Здатність дотримуватися стандартів при розробці програмного забезпечення.

– СК 05. Здатність брати участь у визначенні та формулюванні вимог до програмного забезпечення.

– СК 06. Здатність брати участь у проектуванні програмного забезпечення.

– СК 07. Здатність розробляти модулі і компоненти програмного забезпечення за допомогою типових алгоритмів та інструментів.

– СК 08. Здатність забезпечувати інформаційну та функціональну безпеку програмного забезпечення.

– СК 09. Здатність вибирати та використовувати ефективні інструментальні засоби розробки програмного продукту.

– СК 10. Здатність реалізовувати всі етапи життєвого циклу програмного забезпечення.

4. НОРМАТИВНИЙ ЗМІСТ ПІДГОТОВКИ ЗДОБУВАЧІВ ФАХОВОЇ ПЕРЕДВИЩОЇ ОСВІТИ, СФОРМУЛЬОВАНИЙ У ТЕРМІНАХ РЕЗУЛЬТАТІВ НАВЧАННЯ

– РН 01. Застосовувати основні принципи професійної етики у галузі програмної інженерії, усвідомлювати їх соціальну значимість та культурні аспекти в професійній діяльності.

– РН 02. Систематизувати та узагальнювати інформацію про підходи, методи та засоби розробки супроводу програмного забезпечення.

– РН 03. Застосовувати спеціалізовані емпіричні та теоретичні знання у сфері інженерії програмного забезпечення.

– РН 04. Використовувати знання математичних методів на рівні, необхідному для розв'язання типових задач програмної інженерії.

– РН 05. Розробляти та супроводжувати програмне забезпечення.

– РН 06. Використовувати основні методології та підходи до організації життєвого циклу програмного забезпечення.

– РН 07. Застосовувати стандарти, специфікації в процесах життєвого циклу програмного забезпечення.

– РН 08. Аналізувати вимоги до програмного забезпечення.

– РН 09. Розуміти основні принципи командної роботи при розробці програмного забезпечення.

– РН 10. Обирати та застосовувати ефективні методи оптимізації алгоритмів.

– РН 11. Обирати інструментальні засоби, ефективні методи та здійснювати тестування програмних систем.

– РН 12. Впроваджувати і супроводжувати програмні продукти.

– РН 13. Спілкуватися українською та іноземною мовою усно і письмово з питань інженерії програмного забезпечення.

– РН 14. Розуміти предметну область, застосовувати знання у професійній діяльності.

– РН 15. Аналізувати та узагальнювати необхідну інформацію з різних джерел та ресурсів для розв'язання професійних задач з урахуванням сучасних досягнень інформаційних технологій.

5. ІНФОРМАЦІЙНИЙ ОБСЯГ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

№ з/п	Назва теми	Усього годин	З них	
			практичні заняття	Самостійна робота
1	Вступ до навчальної практики. Організація умовно робочого місця фахівця з інженерії програмного забезпечення	6	2	4
2	Структура та процеси IT-компанії. Гнучкі методології розроблення програмного забезпечення	10	4	6
3	Інструменти командної роботи: система контролю версій та трекери задач	10	4	6
4	Аналіз і специфікація вимог до програмного забезпечення	10	4	6
5	Проєктування програмного продукту: архітектура, база даних та інтерфейс користувача	10	4	6
6	Командне розроблення програмного продукту	22	10	12
7	Тестування та забезпечення якості програмного забезпечення	10	4	6
8	Документування, розгортання та захист командного проєкту	12	4	8
Разом		**90**	**36**	**54**

Форма підсумкового контролю — диференційований залік.

6. ЗМІСТ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Тема 1. Вступ до навчальної практики. Організація умовно робочого місця фахівця з інженерії програмного забезпечення

Мета, завдання та зміст навчальної практики на умовно робочих місцях. Порядок проходження практики, вимоги до виконання завдань, ведення щоденника та оформлення звіту з практики. Інструктаж з охорони праці, техніки безпеки та пожежної безпеки під час роботи з персональним комп'ютером. Ергономіка та санітарно-гігієнічні вимоги до робочого місця. Поняття умовно робочого місця. Модель навчальної ІТ-компанії: команди, ролі, правила внутрішнього розпорядку. Налаштування робочого місця розробника: операційна система, середовище розробки, необхідне програмне забезпечення, облікові записи на платформах командної роботи.

Студенти повинні:

знати: мету, завдання та порядок проходження практики; правила охорони праці й техніки безпеки під час роботи з комп'ютером; вимоги до організації робочого місця; структуру навчальної ІТ-компанії та свою роль у команді;

уміти: організовувати робоче місце з дотриманням вимог охорони праці та ергономіки; встановлювати й налаштовувати середовище розробки та необхідне програмне забезпечення; створювати облікові записи на платформах командної роботи.

Тема 2. Структура та процеси ІТ-компанії. Гнучкі методології розроблення програмного забезпечення

Типи ІТ-компаній: продуктові, аутсорсингові, аутстафінгові. Організаційна структура ІТ-компанії та основні підрозділи. Професійні ролі в команді розроблення: керівник проєкту, бізнес-аналітик, розробник, тестувальник, дизайнер; їхні посадові обов'язки та взаємодія. Рівні кваліфікації фахівців (Trainee, Junior, Middle, Senior). Моделі життєвого

циклу програмного забезпечення: каскадна, ітеративна, інкрементна. Маніфест Agile. Методологія Scrum: ролі, події (спринт, планування спринту, щоденна зустріч, огляд спринту, ретроспектива), артефакти (беклог продукту, беклог спринту, інкремент). Метод Kanban: дошка задач, обмеження незавершеної роботи. Ділова комунікація в команді, корпоративна етика, корпоративні месенджери.

Студенти повинні:

знати: типи IT-компаній та їхню організаційну структуру; професійні ролі в команді розроблення та їхні обов'язки; моделі життєвого циклу програмного забезпечення; принципи Agile, ролі, події та артефакти Scrum, основи Kanban;

уміти: визначати свої посадові обов'язки відповідно до ролі в команді; брати участь у подіях Scrum; формувати беклог продукту; вести Kanban-дошку; дотримуватися правил ділової комунікації.

Тема 3. Інструменти командної роботи: система контролю версій та трекери задач

Призначення систем контролю версій. Основні команди Git: init, clone, add, commit, status, log, diff. Робота з гілками: branch, checkout (switch), merge. Віддалені репозиторії, платформа GitHub: push, pull, fetch. Модель розгалуження Git Flow / GitHub Flow (ознайомлення). Запит на злиття (pull request), перевірка коду (code review), коментування змін. Конфлікти злиття та способи їх розв'язання. Правила оформлення комітів, файл .gitignore. Трекери задач (GitHub Projects, Trello, Jira): створення задач, призначення виконавців, статуси задач, мітки, терміни виконання.

Студенти повинні:

знати: призначення та основні команди системи контролю версій Git; принципи роботи з гілками та віддаленими репозиторіями; порядок

створення запиту на злиття та проведення перевірки коду; можливості трекерів задач;

уміти: створювати та клонувати репозиторії; працювати з гілками; створювати запити на злиття та розв'язувати конфлікти; виконувати перевірку коду колег; створювати та вести задачі у трекері.

Тема 4. Аналіз і специфікація вимог до програмного забезпечення

Роль замовника та бізнес-аналітика у проєкті. Методи збирання вимог: інтерв'ю, анкетування, аналіз аналогів. Функціональні та нефункціональні вимоги. Користувацькі історії (user stories) та критерії приймання. Пріоритизація вимог. Технічне завдання на розроблення програмного забезпечення: структура та вимоги до оформлення. Оцінювання трудомісткості задач (story points, експертна оцінка), декомпозиція задач. Планування спринту.

Студенти повинні:

знати: види вимог до програмного забезпечення; методи збирання вимог; правила формулювання користувацьких історій та критеріїв приймання; структуру технічного завдання; способи оцінювання трудомісткості задач;

уміти: збирати та аналізувати вимоги замовника; формулювати функціональні та нефункціональні вимоги й користувацькі історії; складати технічне завдання; декомпозувати й оцінювати задачі; планувати роботу на спринт.

Тема 5. Проєктування програмного продукту: архітектура, база даних та інтерфейс користувача

Мета та етапи проєктування програмного забезпечення. Вибір технологій та засобів розроблення. Поняття архітектури програмного забезпечення, клієнт-серверна та багаторівнева архітектура (ознайомлення).

Мова моделювання UML: діаграма варіантів використання, діаграма класів, діаграма діяльності. Проєктування бази даних: сутності, атрибути, зв’язки, ER-діаграма. Проєктування інтерфейсу користувача: каркасні макети (wireframes), прототипування, основні принципи зручності використання (UI/UX). Інструменти проєктування: draw.io, Figma.

Студенти повинні:

знати: етапи проєктування програмного забезпечення; основні види UML-діаграм та їх призначення; принципи проєктування баз даних; основні принципи проєктування інтерфейсу користувача;

уміти: обирати технології для реалізації проєкту; будувати UML-діаграми варіантів використання та класів; розробляти ER-діаграму бази даних; створювати макети інтерфейсу за допомогою інструментів проєктування.

Тема 6. Командне розроблення програмного продукту

Організація структури програмного проєкту. Стандарти та стиль кодування, угоди про іменування. Розподіл задач між членами команди відповідно до ролей. Ітеративне розроблення програмного продукту протягом спринтів. Реалізація модулів програмного продукту: інтерфейс користувача, бізнес-логіка, робота з базою даних. Інтеграція модулів. Щоденні зустрічі команди, відстеження виконання задач у трекері. Перевірка коду (code review) та рефакторинг. Огляд результатів спринту та ретроспектива. Ведення щоденника практики та фіксування індивідуального внеску в командний проєкт.

Студенти повинні:

знати: порядок організації командного розроблення; стандарти та стиль кодування; правила проведення перевірки коду; зміст подій спринту; способи інтеграції модулів програмного продукту;

уміти: реалізовувати модулі програмного продукту відповідно до поставлених задач; дотримуватися стандартів кодування; працювати з кодом у спільному репозиторії; проводити перевірку коду та рефакторинг; звітувати про виконану роботу на щоденних зустрічах і огляді спринту.

Тема 7. Тестування та забезпечення якості програмного забезпечення

Поняття якості програмного забезпечення. Мета та принципи тестування. Рівні тестування: модульне, інтеграційне, системне, приймальне. Види тестування: функціональне, нефункціональне, регресійне. Методи «чорної» та «білої» скриньки. Тест-план, чек-лист, тест-кейс: структура та правила складання. Життєвий цикл дефекту, оформлення звіту про дефект (баг-репорту), пріоритет та серйозність дефекту. Модульне тестування з використанням тестових фреймворків (ознайомлення). Виправлення дефектів та повторне тестування.

Студенти повинні:

знати: поняття якості програмного забезпечення; рівні та види тестування; структуру тест-кейсу та баг-репорту; життєвий цикл дефекту; призначення модульного тестування;

уміти: складати чек-листи та тест-кейси; проводити ручне тестування програмного продукту; оформлювати звіти про дефекти; розробляти прості модульні тести; виправляти виявлені дефекти.

Тема 8. Документування, розгортання та захист командного проєкту

Види документації програмного забезпечення: технічна, користувацька, проєктна. Коментування коду, файл README. Інструкція користувача. Підготовка релізу, версіонування програмного продукту. Розгортання програмного продукту (локально, на хостингу, через GitHub Pages — за

вибором). Передавання продукту замовнику. Підготовка презентації та демонстрація програмного продукту. Оформлення звіту з навчальної практики. Захист командного проєкту та звіту з практики.

Студенти повинні:

знати: види та склад документації програмного забезпечення; правила оформлення файлу README та інструкції користувача; порядок підготовки релізу та розгортання програмного продукту; вимоги до оформлення та захисту звіту з практики;

уміти: документувати програмний продукт; готувати реліз та розгортати програмний продукт; презентувати результати командної роботи; аргументовано пояснювати власний внесок у проєкт; оформлювати та захищати звіт з практики.

7. ТЕМИ ПРАКТИЧНИХ ЗАНЯТЬ

№ з/п	№ заняття	Тема заняття	Кількість годин
1	1	Тема 1. Вступ до навчальної практики. Організація умовно робочого місця фахівця з інженерії програмного забезпечення Практичне заняття 1. Інструктаж з охорони праці та техніки безпеки. Ознайомлення з моделлю навчальної ІТ-компанії, формування команд і розподіл ролей. Налаштування умовно робочого місця розробника.	2
2	2	Тема 2. Структура та процеси ІТ-компанії. Гнучкі методології розроблення програмного забезпечення Практичне заняття 2. Організаційна структура ІТ-компанії. Професійні ролі та посадові обов’язки членів команди. Правила ділової комунікації.	2
3	3	Тема 2. Структура та процеси ІТ-компанії. Гнучкі методології розроблення програмного забезпечення Практичне заняття 3. Методологія Scrum: ролі, події, артефакти. Kanban-дошка. Формування беклогу продукту командного проєкту.	2
4	4	Тема 3. Інструменти командної роботи: система контролю версій та трекери задач Практичне заняття 4. Система контролю версій Git. Створення командного репозиторію на GitHub. Робота з гілками, комітами та віддаленим репозиторієм.	2
5	5	Тема 3. Інструменти командної роботи: система контролю версій та трекери задач Практичне заняття 5. Запити на злиття (pull request) та перевірка коду. Розв’язання конфліктів злиття. Налаштування трекера задач.	2
6	6	Тема 4. Аналіз і специфікація вимог до програмного забезпечення Практичне заняття 6. Збирання вимог: інтерв’ю із «замовником». Функціональні та нефункціональні вимоги. Користувацькі історії та критерії приймання.	2
7	7	Тема 4. Аналіз і специфікація вимог до програмного забезпечення Практичне заняття 7. Складання технічного завдання на розроблення програмного продукту. Декомпозиція та оцінювання задач, планування спринту.	2
8	8	Тема 5. Проєктування програмного продукту: архітектура, база даних та інтерфейс користувача Практичне заняття 8. Вибір технологій та архітектури програмного продукту. Побудова UML-діаграм варіантів використання та класів.	2
9	9	Тема 5. Проєктування програмного продукту: архітектура, база даних та інтерфейс користувача Практичне заняття 9. Проєктування бази даних (ER-діаграма). Розроблення макетів інтерфейсу користувача.	2
10	10	Тема 6. Командне розроблення програмного продукту Практичне заняття 10. Спринт 1. Створення структури проєкту, узгодження стандартів кодування. Реалізація базових модулів програмного продукту.	2
11	11	Тема 6. Командне розроблення програмного продукту Практичне заняття 11. Спринт 1. Реалізація задач беклогу	2

		спринту. Щоденні зустрічі команди, відстеження задач у трекері.	
12	12	Тема 6. Командне розроблення програмного продукту Практичне заняття 12. Спринт 1. Перевірка коду, огляд результатів спринту, ретроспектива. Планування спринту 2.	2
13	13	Тема 6. Командне розроблення програмного продукту Практичне заняття 13. Спринт 2. Реалізація роботи з базою даних та інтеграція модулів програмного продукту.	2
14	14	Тема 6. Командне розроблення програмного продукту Практичне заняття 14. Спринт 2. Доопрацювання функціональності, рефакторинг, огляд результатів спринту та ретроспектива.	2
15	15	Тема 7. Тестування та забезпечення якості програмного забезпечення Практичне заняття 15. Складання чек-листів і тест-кейсів. Ручне тестування програмного продукту, оформлення баг-репортів.	2
16	16	Тема 7. Тестування та забезпечення якості програмного забезпечення Практичне заняття 16. Модульне тестування. Виправлення дефектів та регресійне тестування.	2
17	17	Тема 8. Документування, розгортання та захист командного проєкту Практичне заняття 17. Документування програмного продукту: файл README, інструкція користувача. Підготовка релізу та розгортання продукту.	2
18	18	Тема 8. Документування, розгортання та захист командного проєкту Практичне заняття 18. Демонстрація програмного продукту «замовнику». Захист командного проєкту та звіту з навчальної практики.	2
Усього			**36**

8. САМОСТІЙНА РОБОТА

Самостійна робота студентів є складовою навчальної практики на умовно робочих місцях і спрямована на закріплення професійних навичок, отриманих під час практичних занять, та на виконання власної частини командного проєкту. Обсяг самостійної роботи становить 54 години. Самостійна робота виконується в позааудиторний час відповідно до ролі студента в команді та задач, призначених йому в трекері задач.

№ з/п	Назва теми та зміст самостійної роботи	Кількість годин
1	**Тема 1. Вступ до навчальної практики. Організація умовно робочого місця фахівця з інженерії програмного забезпечення** Вивчення правил охорони праці та вимог ергономіки під час роботи з персональним комп'ютером. Встановлення та налаштування середовища розробки, системи контролю версій Git і необхідного програмного забезпечення на власному комп'ютері. Створення облікових записів на платформах командної роботи. Початок ведення щоденника практики.	4
2	**Тема 2. Структура та процеси IT-компанії. Гнучкі методології розроблення програмного забезпечення** Опрацювання офіційного посібника зі Scrum (The Scrum Guide) та Agile-маніфесту. Аналіз вакансій IT-компаній щодо вимог до професійних ролей і рівнів кваліфікації. Підготовка опису власної ролі та посадових обов'язків у команді. Доповнення беклогу продукту командного проєкту.	6
3	**Тема 3. Інструменти командної роботи: система контролю версій та трекери задач** Самостійне опрацювання розділів книги Pro Git (робота з гілками, віддалені репозиторії). Виконання вправ з Git у навчальному репозиторії. Оформлення файлу .gitignore і правил іменування гілок та комітів для командного репозиторію. Заповнення трекера задач командного проєкту.	6
4	**Тема 4. Аналіз і специфікація вимог до програмного забезпечення** Вивчення предметної області командного проєкту, аналіз програмних продуктів-аналогів. Доопрацювання функціональних і нефункціональних вимог, користувацьких історій та критеріїв приймання. Оформлення технічного завдання.	6
5	**Тема 5. Проєктування програмного продукту: архітектура, база даних та інтерфейс користувача** Вивчення документації обраних технологій розроблення. Доопрацювання UML-діаграм та ER-діаграми бази даних. Розроблення макетів інтерфейсу користувача у Figma або draw.io.	6
6	**Тема 6. Командне розроблення програмного продукту** Реалізація індивідуальних задач беклогу спринту. Самостійне опрацювання документації бібліотек і фреймворків, що використовуються в проєкті. Перевірка коду колег (code review) та рефакторинг власного коду. Ведення щоденника практики та	12

	фіксування індивідуального внеску в командний проєкт.	
7	**Тема 7. Тестування та забезпечення якості програмного забезпечення** Опрацювання глосарію термінів тестування ISTQB. Складання чек-листів і тест-кейсів для власних модулів. Розроблення модульних тестів, виправлення виявлених дефектів.	6
8	**Тема 8. Документування, розгортання та захист командного проєкту** Підготовка документації програмного продукту (файл README, інструкція користувача). Оформлення щоденника та звіту з навчальної практики. Підготовка презентації та доповіді до захисту командного проєкту.	8
Разом		**54**

Форми контролю самостійної роботи: перевірка задач у трекері та історії змін у спільному репозиторії; оцінювання артефактів командного проєкту (технічного завдання, діаграм, макетів, програмного коду, тест-кейсів, документації); перевірка записів у щоденнику практики; усне опитування під час практичних занять і щоденних зустрічей команди. Результати самостійної роботи враховуються під час виставлення оцінок поточного контролю та під час захисту командного проєкту і звіту з навчальної практики.

9. ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Індивідуальне завдання передбачено навчальним планом і виконується в межах годин самостійної роботи. Воно полягає у виконанні студентом власної частини командного проєкту відповідно до його ролі в команді та в оформленні звіту з навчальної практики. Індивідуальне завдання дає змогу оцінити особистий внесок кожного студента в командний результат.

Зміст індивідуального завдання:

– розроблення закріпленого за студентом модуля (функціональної частини) програмного продукту з проходженням основних етапів: уточнення вимог до модуля, проєктування, реалізація, тестування та документування;

– виконання задач, призначених студентові у трекері задач, з фіксуванням результатів у спільному репозиторії (коміти, запити на злиття);

– ведення щоденника практики;

– оформлення та захист звіту з навчальної практики.

Орієнтовні напрями індивідуальних завдань (визначаються командою під час планування з урахуванням теми проєкту та ролі студента):

– модуль реєстрації та автентифікації користувачів;

– модуль роботи з базою даних (створення, перегляд, редагування та видалення записів);

– модуль пошуку, фільтрації та сортування даних;

– модуль формування звітів або статистики;

– інтерфейс користувача або адміністративна панель;

– набір тест-кейсів, модульних тестів і звітів про дефекти для програмного продукту;

– технічна та користувацька документація, підготовка релізу та розгортання програмного продукту.

Структура звіту з навчальної практики:

– титульний аркуш;

– зміст;

– вступ (мета та завдання практики);

– характеристика навчальної ІТ-компанії, складу команди та ролі студента;

– опис командного проєкту: вимоги, технічне завдання, обрані технології;

– опис індивідуального внеску: виконані задачі з трекера, реалізований модуль, посилання на коміти та запити на злиття;

– результати тестування;

– висновки;

– список використаних джерел;

– додатки (діаграми, макети інтерфейсу, фрагменти програмного коду, знімки екрана, щоденник практики).

Порядок подання та захисту: звіт оформлюється державною мовою в електронному та друкованому вигляді відповідно до вимог коледжу до оформлення письмових робіт і подається керівникові практики до початку захисту. Захист індивідуального завдання та звіту відбувається на останньому практичному занятті під час диференційованого заліку: студент демонструє реалізовану частину програмного продукту та відповідає на запитання щодо свого внеску в проєкт.

Оцінювання: індивідуальне завдання оцінюється за 5-бальною шкалою з урахуванням якості та повноти реалізованого модуля, дотримання стандартів кодування, регулярності роботи в репозиторії й трекері, якості оформлення звіту та відповідей під час захисту. Оцінка за індивідуальне завдання враховується під час визначення підсумкової оцінки з навчальної дисципліни.

10. МЕТОДИ НАВЧАННЯ, КОНТРОЛЮ ТА ЗАСОБИ ОЦІНЮВАННЯ

Методи навчання

– словесні (інструктаж, пояснення, бесіда, обговорення, консультація);
– наочні (демонстрація роботи інструментів командної роботи, показ зразків документації та програмного коду, презентації, відеоматеріали);
– практичні (виконання професійних завдань на умовно робочих місцях, розроблення, тестування та документування програмного продукту);
– імітаційні (моделювання діяльності ІТ-компанії, рольова взаємодія «замовник — команда розробників», ділові ігри);
– метод проєктів (виконання командного проєкту як завершеного програмного продукту);
– проблемно-пошукові (самостійний пошук рішень поставлених задач, вибір технологій та засобів розроблення);
– робота з технічною документацією, довідковими та мережевими джерелами інформації;
– взаємонавчання та взаємоперевірка під час перевірки коду (code review);
– індивідуальні та групові консультації.

Методи контролю

Усне опитування (фронтальне, індивідуальне); спостереження за виконанням професійних обов’язків на умовно робочих місцях; перевірка виконання задач у трекері та історії змін у репозиторії; оцінювання артефактів кожного етапу проєкту (технічного завдання, діаграм, програмного коду, тест-кейсів, документації); оцінювання участі в подіях Scrum; перевірка щоденника та звіту з практики; захист командного проєкту; самоконтроль та взаємооцінювання.

Засоби оцінювання

– технічне завдання та проєктна документація (UML-діаграми, ER-діаграма, макети інтерфейсу);

– програмний код у спільному репозиторії та історія змін;

– задачі в трекері та беклог командного проєкту;

– тест-кейси та звіти про дефекти;

– щоденник та звіт з навчальної практики;

– результати виконання завдань самостійної роботи;

– усне фронтальне та індивідуальне опитування;

– презентація та захист командного проєкту.

11. ПОРЯДОК ТА КРИТЕРІЇ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Результати навчання студента, що відображають досягнутий ним рівень компетентностей відносно очікуваних, ідентифікують і вимірюють у процесі контрольних заходів за допомогою критеріїв. Оцінювання студентів охоплює низку письмових, усних та практичних контрольних процедур, які використовують відповідно до компетентнісних характеристик (знання, вміння/навички, комунікація, відповідальність і автономія).

Оцінювання результатів навчання студентів проводиться під час поточного і підсумкового контролю. Поточний контроль здійснюється на кожному практичному занятті у формі усного опитування, спостереження за виконанням професійних обов’язків на умовно робочих місцях, перевірки виконаних задач у трекері та репозиторії, оцінювання артефактів відповідного етапу командного проєкту. Підсумковий контроль проводиться у формі диференційованого заліку, що передбачає захист командного проєкту та звіту з навчальної практики.

Оцінювання здійснюється за 5-бальною шкалою. Позитивними вважаються оцінки від 2 до 5 балів, що відповідають чотирьом рівням навчальних досягнень: початковому (2 бали), середньому (3 бали), достатньому (4 бали) та високому (5 балів); оцінка 1 бал є незадовільною. Оцінка за кожне практичне заняття виставляється у журнал. Підсумкова оцінка за навчальну дисципліну визначається як середнє арифметичне оцінок поточного контролю з урахуванням оцінок за індивідуальний внесок у командний проєкт, захист проєкту та звіту з практики й округлюється до цілого числа.

Критерії оцінювання навчальних досягнень студентів за 5-бальною шкалою

Рівень навчальних досягнень	Бали	Критерії оцінювання навчальних досягнень
Незадовільний	**1**	Студент не володіє базовими поняттями професійної діяльності (команда, задача, репозиторій, тестування), не виконує обов'язків на умовно робочому місці та задач командного проєкту; звіт з практики не підготовлений.
Початковий	**2**	Студент має фрагментарні уявлення про структуру ІТ-компанії та етапи розроблення програмного забезпечення; за зразком і з детальною допомогою викладача виконує найпростіші операції (налаштування робочого місця, створення облікового запису, клонування репозиторію), але не завжди розуміє їх призначення; участь у роботі команди мінімальна; звіт з практики містить суттєві недоліки.
Середній	**3**	Студент під керівництвом викладача виконує типові задачі командного проєкту за наведеним зразком; користується основними командами Git і трекером задач, але допускає помилки, які усуває з допомогою викладача або колег по команді; плутає етапи життєвого циклу програмного забезпечення та події Scrum, бере участь у них пасивно; свою роль у команді пояснює фрагментарно.
Достатній	**4**	Студент самостійно виконує задачі беклогу спринту, працює з гілками та запитами на злиття, веде задачі у трекері; бере участь у формулюванні вимог і проєктуванні, реалізує доручені модулі, складає тест-кейси, проводить перевірку коду колег; припускається несуттєвих помилок, які самостійно виявляє й усуває; аргументовано пояснює свою роботу та внесок у проєкт.
Високий	**5**	Студент самостійно та якісно виконує задачі на всіх етапах проєкту — від аналізу вимог до тестування й документування; вільно володіє інструментами командної роботи, обґрунтовано обирає технології та рішення, розв'язує конфлікти злиття, проводить рефакторинг; дотримується термінів, бере на себе координацію роботи команди, допомагає колегам; успішно демонструє завершений програмний продукт «замовнику», виявляє творчий підхід і відповідальність; вільно володіє державною мовою.

12. СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Основна:

1. Мартін Р. Чистий код. Створення і рефакторинг за допомогою Agile / Роберт Мартін ; пер. з англ. — Харків : Фабула, 2019. — 416 с.
2. Сазерленд Дж. Scrum. Навчись робити вдвічі більше за менший час / Джефф Сазерленд ; пер. з англ. Я. Лебеденка. — Харків : КСД, 2022. — 280 с.

Додаткова:

3. Лавріщева К. М. Програмна інженерія : підручник / К. М. Лавріщева. — Київ : Академперіодика, 2008. — 319 с.
4. Бабенко Л. П. Основи програмної інженерії : навчальний посібник / Л. П. Бабенко, К. М. Лавріщева. — Київ : Знання, 2001. — 269 с.
5. Чакон С. Pro Git / С. Чакон, Б. Штрауб. — 2-ге вид. — Apress, 2014. Український переклад. URL: https://git-scm.com/book/uk/v2
6. ДСТУ ISO/IEC/IEEE 12207:2018 (ISO/IEC/IEEE 12207:2017, IDT). Інженерія систем і програмних засобів. Процеси життєвого циклу програмних засобів.
7. ДСТУ ISO/IEC 25010:2025 (ISO/IEC 25010:2023, IDT). Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Модель якості продукту.

Інформаційні ресурси:

8. The Scrum Guide — офіційний посібник зі Scrum (доступний українською мовою). URL: https://scrumguides.org
9. Agile-маніфест розробки програмного забезпечення (офіційний переклад українською). URL: https://agilemanifesto.org/iso/uk/manifesto.html
10. GitHub Docs — офіційна документація платформи GitHub. URL: https://docs.github.com

11. GitHub Projects — документація щодо планування та відстеження задач. URL: https://docs.github.com/issues/planning-and-tracking-with-projects
12. Atlassian Agile Coach — матеріали про Scrum, Kanban та командну роботу. URL: https://www.atlassian.com/agile
13. Jira Software — система керування проєктами та задачами. URL: https://www.atlassian.com/software/jira
14. Trello — сервіс для керування задачами за методом Kanban. URL: https://trello.com
15. draw.io (diagrams.net) — онлайн-інструмент для побудови UML- та ER-діаграм. URL: https://app.diagrams.net
16. Figma — онлайн-інструмент для проєктування інтерфейсів користувача. URL: https://www.figma.com
17. ISTQB Glossary — глосарій термінів тестування програмного забезпечення. URL: https://glossary.istqb.org
18. MDN Web Docs — довідник з веб-технологій. URL: https://developer.mozilla.org
19. Microsoft Learn — навчальні матеріали та документація з технологій розроблення. URL: https://learn.microsoft.com/uk-ua/
20. Visual Studio Code — офіційна документація середовища розробки. URL: https://code.visualstudio.com/docs
21. roadmap.sh — дорожні карти розвитку для IT-фахівців різних спеціалізацій. URL: https://roadmap.sh
22. DOU — спільнота українських IT-фахівців: статті про структуру IT-компаній, професійні ролі та процеси. URL: https://dou.ua
23. Stack Overflow — платформа запитань і відповідей для пошуку рішень програмних задач. URL: https://stackoverflow.com
24. Prometheus — українська платформа безкоштовних онлайн-курсів, зокрема з програмування та управління проєктами. URL: https://prometheus.org.ua