

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка

Методичні рекомендації
до виконання практикуму з навчальної дисципліни
«Проект з розробки системи опрацювання даних»
для студентів спеціальності 122 Комп'ютерні науки,
які навчаються за освітньою програмою «Комп'ютерні науки»

Львів 2025

Затверджено
кафедрою оптоелектроніки
та інформаційних технологій
Протокол №6 від 24.06.2025 р.

Уклали:
Василь Васюта
Богдан Горон
Олег Кушнір

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
до виконання практикуму з навчальної дисципліни
«Проект з розробки системи опрацювання даних» для студентів
спеціальності 122 Комп'ютерні науки,
які навчаються за освітньою програмою «Комп'ютерні науки»

Формат 60x84/16. Умовн. друк. арк. 2,5. Тираж 100 прим. Зам.

Видавець та виготовлювач:
Львівський національний університет імені Івана Франка,
вул. Університетська, 1, м. Львів, 79000
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру видавців, виготівників
і розповсюджувачів видавничої продукції
Серія ДК №3059 від 13.12.2007 р.

ЗМІСТ

1. Загальні положення	4
2. Практикум.....	6
2.1. Вибір задачі для проекту.....	6
2.2. Схема практикуму	7
2.3. Вибір спеціалізації студента	8
2.4. Організація навчального процесу.....	10
2.5. Академічна доброчесність.....	11
3. Навчальні матеріали	12
4. Командна робота.....	13
Література.....	15
ДОДАТОК А	16
ДОДАТОК Б.....	19
ДОДАТОК В.....	20

1. Загальні положення

Вибіркова навчальна дисципліна «Проект з розробки системи опрацювання даних» спрямована на набуття студентами практичних навичок, необхідних для розробки систем опрацювання даних. Реальні комп'ютерні системи, задачею яких є робота з даними, є великими за розмірами і вимагають залучення ІТ-спеціалістів різних спеціалізацій (розробник ПЗ, інженер контролю якості, UI/UX дизайнер, тощо). Тому акцент у виконанні даного практикуму робиться на реалізацій кожним студентом тієї частини системи, яка відповідає його спеціалізації. Фактично, практикум є симуляцією командної роботи ІТ-спеціалістів над конкретним проектом, а самі практичні роботи більше схожі на послідовні етапи розробки єдиної великої комп'ютерної програми, ніж на традиційні практичні заняття.

Метою виконання практикуму з цієї дисципліни є ознайомлення студентів з практичними основами командної роботи над проектом із розробки різних систем опрацювання даних, починаючи з початкового етапу дослідження вимог до продукту, розробку базової версії та ітеративної розробки за методологією Scrum.

Обсяг навчальної дисципліни складає 48 год. практичних занять, що включають також семінари та лабораторні завдання, і 72 год. самостійної роботи, сумарно – 120 год.

Бали, які можна отримати в рамках цього курсу, розподілені так: залежно від обраної ролі, студент виконує 3 практичні роботи з максимально можливою оцінкою 15 балів за кожну роботу (45 балів усього); 15 балів відводиться для оцінювання представлення своєї роботи на занятті, з презентацією фінального продукту; ще

40 балів нараховують на два тестування (20 балів максимально за кожен тест).

Спеціальні компетентності, які здобуваються під час вивчення цієї навчальної дисципліни, такі:

СК 8. Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.

СК 9. Здатність реалізувати багаторівневу обчислювальну модель на основі архітектури клієнт-сервер, включаючи бази даних, знань і сховища даних, виконувати розподілену обробку великих наборів даних на кластерах стандартних серверів для забезпечення обчислювальних потреб користувачів, у тому числі на хмарних сервісах.

СК 10. Здатність застосовувати методології, технології та інструментальні засоби для управління процесами життєвого циклу інформаційних і програмних систем, продуктів і сервісів інформаційних технологій відповідно до вимог замовника.

СК 11. Здатність до інтелектуального аналізу даних на основі методів обчислювального інтелекту включно з великими та погано структурованими даними, їхньої оперативної обробки та візуалізації результатів аналізу в процесі розв'язування прикладних задач.

СК 12. Здатність забезпечити організацію обчислювальних процесів в інформаційних системах різного призначення з урахуванням архітектури, конфігурування, показників результативності функціонування операційних систем і системного програмного забезпечення.

2. Практикум

2.1. Вибір задачі для проєкту

Більшість комп'ютерних систем так чи інакше пов'язані з опрацюванням даних, однак в контексті даного курсу акцент ставиться на опрацюванні даних як основній функціональній особливості системи. Однак навіть із таким обмеженням, спектр систем все ще залишається надзвичайно різноманітним. Це процеси ETL (extract-transform-load), процесинг Big Data, аналітичні системи для дашбордів реального часу, аналіз даних засобами машинного навчання тощо. Тому для збереження загальності задачі для практичних робіт сформульовані агностично до самої системи: наприклад, розробник будь-якої системи повинен покрити базову логіку програмними тестами, а девопс має десь реалізовану систему захостити.

Це в свою чергу дозволяє студентам самим обрати задачу, яку вони будуть реалізовувати впродовж курсу. **Вибір задачі для проєкту здійснюється до 2-го заняття** даного курсу.

Системами для реалізації можуть бути, зокрема:

- *рекомендаційна система* для
 - підбору фільму (на основі відкритого IMDB-дату-сету);
 - підбору книги за жанром/автором тощо;
- *веб-скрапери* для збору даних про
 - LinkedIn-анкети;
 - ціни на товари в різних веб-магазинах;
 - ціни на оренду житла в різних районах;
 - афіші подій у місті;
 - музичних виконавців;
- *предикативні системи* для
 - прогнозу результатів спортивних (кіберспортивних) ігор;

- прогнозу цін на об'єкти нерухомості.

Студенти можуть також пропонувати свої теми для розробки. Однак єдиним обмеженням на тему є потреба в **опрацюванні даних з потенційним застосуванням методів машинного навчання**.

Зауважимо, що задачі такого типу є задачами багатьох реальних комерційних проєктів, які розробляються роками, тому вони не можуть бути повністю виконані в рамках університетського курсу. Однак, як показує практика, часто студенти виконують проєкт на рівні фактичного MVP (Minimal Viable Product).

2.2. Схема практикуму

Курс поділено на **3 фази тривалістю 5 навчальних тижнів** кожна (див. ДОДАТОК В. СХЕМА КУРСУ):

- **Фаза дискавері** (тижні 1–5) – підготовка до розробки логіки, узгодження вимог до програмного забезпечення, розбиття проєкту на підзадачі, створення репозиторіїв і кістяків проєктів тощо.
- **Фаза MVP** (тижні 6–10) – розробка базової версії функціоналу. На момент закінчення фази 2 передбачається наявність в команді працюючої системи з базовим функціоналом.
- **Фаза ітеративної розробки** (тижні 11–15) – робота команди за методологією скраму для ітеративної розробки нового функціоналу.

Кожна фаза закінчується **захисним заняттям** (тижні 5, 10 і 15), до якого студент повинен підготувати належний звіт про виконану роботу. **Робота, яка не завантажена в середовище Moodle за встановлений до захисного заняття час, НЕ приймається.**

2.3. Вибір спеціалізації (ролі) студента

Як зауважено в першому розділі, для ефективної роботи над проектом вводять ІТ-спеціалізації, представники яких виконують відведену їм функцію. В даному курсі **кожному студенту пропонують обрати одну з семи ролей:**

- розробник програмного забезпечення;
- інженер контролю якості;
- проектний менеджер;
- DevOps;
- архітектор програмного забезпечення;
- бізнес-аналітик;
- дизайнер UI/UX.

Про відповідальності кожної з ролей можна прочитати при підготовці до відповідного заняття ([Moodle | Тема 1.](#)). **Кількість спеціалістів кожної ролей різна** залежно від обсягу роботи над конкретним проектом, проте для більшості проектів зберігаються деякі неформальні пропорції – один інженер контролю якості припадає на 2–3 розробників, 1–2 проектних менеджери (залежить від розміру команди), 1 дизайнер UI/UX тощо. **Рекомендована кількість кожної зі спеціалізацій подана в таблиці нижче за текстом.**

Залежно від обраної ролі, студент повинен виконати **3 практичні роботи лабораторного плану:**

Роль (рекомендована кількість студентів на роль)	Фаза 1 (тижні 1–5)	Фаза 2 (тижні 6–10)	Фаза 3 (тижні 11–15)
<i>Розробник ПЗ (3–...)</i>	Створення кістяків проектів. Налаштування кросс-катінгу в	Написання логічної частини аплікації	Написання автоматизованих тестів

	аспектно-орієнтованому підході		
QA-інженер (2)	Розробка тест-плану, написання тест-кейсів до функціональних вимог	Написання тест-кейсів і проведення нефункціонального тестування	Створення звіту про результати тестування (Test Summary Report)
Проектний менеджер (2)	Старт проекту. Створення базової проектної документації	Налаштування системи проектного трекінгу	Створення проектної документації і організація роботи команди
DevOps (1-2)	Налаштування гітхаб-політик для гілок. Створення docker-файлу і налаштування локального запуску	Налаштування хостингу	Створення CI/CD-пайплайнів
Архітектор ІЗ (1-2)	Побудова архітектурної діаграми. Вибір і обґрунтування обраних технологій	Створення WBS	Створення Run Book
Бізнес-аналітик (1-2)	Виділення функціональних особливостей. Розбиття задачі по функціональних частинах	Написання звіту про аналіз конкурентів	Створення SRS
UI/UX дизайнер (1)	Створення палітри кольорів, схеми розмірів і типографії	Створення дизайн-системи окремих компонент	Створення мокапів базових сценаріїв

Якщо кількість студентів у групі більша за 15, то групу ділять на дві команди, кожна з яких працює незалежно.

Кожна практична робота, яку нижче ми називаємо лабораторною, має відповідну сторінку в середовищі Moodle із описом завдань до роботи. Якщо одну лабораторну роботу

виконують кілька студентів, то звіт до цієї роботи повинен містити тільки індивідуальний внесок автора. Якщо ті самі результати містяться у двох і більше звітах, вони не будуть зараховані.

2.4. Організація навчального процесу

Заняття до курсу триває півтори пари посліпль. Більшість практичних занять (окрім тижнів 1, 5, 7, 10, 14, 15 і 16) зазвичай проводять так: на першій півпарі розглядають теоретичний матеріал даного заняття (див. ДОДАТОК В. СХЕМА КУРСУ). Повну пару опісля студенти працюють над своїми лабораторними роботами.

Під час виконання лабораторної роботи на парі студент отримує консультації від викладача.

Кожна лабораторна робота має набір менших завдань із відповідною максимальною кількістю балів за завдання. Студенти кожної команди, які обрали ту саму спеціалізацію, повинні узгодити задачі між собою. Крім того, в кожній групі повинна бути виконана обов'язкова складова задач (позначена зірочкою) для даної лабораторної роботи. Приклад задачі до лабораторної роботи №1.1 показано на рисунку:

Задачі для виконання у ЛР

* обов'язкові для виконання у кожній команді

(2б.)* Вибір і обґрунтування стратегії управління кодом: окремі репозиторії чи монорепозиторії.

(3б.)* Вибір і обґрунтування робочого процесу: Git Flow чи trunk-based.

(5б. за кожну складову)* Для кожної зі складових системи (клієнтська частина фронтенд, серверна частина бекенд, рекомендаційна система) розробити спосіб розбиття файлів по папках з врахуванням сучасних підходів (чиста архітектура, N-рівнева архітектура, DDD, менеджер станів (state manager)).

(10б. за кожну складову)* Реалізувати кістяк проекту з кросс-катінгом (логування, кешування та обробку помилок) в аспектно-орієнтованому підході на бекенді та фронтенді.

(5б.) Реалізувати функціонал надсилання та валідації форм на фронтенді.

(5б.) Налаштувати систему стилів для фронтенду для обраної бібліотеки стилів (Tailwind CSS, Bootstrap, тощо).

Заняттями, на яких не виконуються лабораторні роботи, є:

- Заняття 1: Вступне заняття.
- Заняття 5: Захисне заняття фази 1.
- Заняття 7: Тестування 1 та проведення студентами демо-результатів своєї роботи.
- Заняття 10: Захисне заняття фази 2.
- Заняття 14: Реліз продукту. Представлення фінальних результатів.
- Заняття 15: Захисне заняття фази 3.
- Заняття 16: Заключне заняття. Тестування 2. Анкетування.

Звіт про виконання лабораторної роботи слід завантажити в середовище Moodle за два дні до початку захисного заняття.

Згідно з ухвалою Вченої ради факультету електроніки на комп'ютерних технологій, після завершенням семестру виконання, прийом і захист лабораторних робіт *припиняється*. Під час сесії та на талонах студент має змогу здавати лише теоретичний матеріал (модулі).

Зверніть увагу! Оскільки вивчення дисципліни завершується заліком, то студенти, які на момент завершення семестру набрали *10 балів або менше* сумарно за практичні завдання (лабораторні роботи) та представлення фінальних результатів, втрачають навіть теоретичну можливість здати дисципліну за талонами 2 і «К», незалежно від їхніх результатів на контрольних замірах знань і Perezдачі модулів за талонами.

Зверніть увагу! Бали можна здобути лише за виконання практичних завдань (лабораторних робіт), усне представлення фінальних результатів роботи і теоретичні модулі. У плані дисципліни немає балів за написання рефератів або виконання інших робіт, не передбачених силабусом.

2.5. Академічна доброчесність

Очікується, що практичні роботи та дані контрольних замірів знань студентів будуть їхніми оригінальними дослідженнями або міркуваннями. Здавання роботи, виконаної не власноруч, а іншою людиною, відсутність посилань на використані джерела, фабрикування джерел, списування, втручання в роботу інших студентів становлять, але не обмежують, приклади можливої академічної недоброчесності. Виявлення ознак академічної недоброчесності в роботі студента є підставою для її незарахування викладачем, незалежно від масштабів плагіату або спроб обману.

Отже, якщо одну практичну роботу виконують кілька студентів, то звіт до неї повинен містити тільки індивідуальний внесок автора. Якщо ті самі результати містяться у двох і більше звітах, то їх не зараховують.

3. Навчальні матеріали

Усі матеріали, потрібні для вивчення дисципліни «Проект з розробки системи опрацювання даних», містяться в середовищі Moodle: <https://e-learning.lnu.edu.ua/course/view.php?id=6957>.

Наведений там матеріал є двох типів:

- **Інваріантна складова.** Це теми, обов'язкові для вивчення кожним студентом курсу (незалежно від обраної спеціалізації), представлені у додатках А і В. В середовищі Moodle для кожного семінарського заняття можна знайти відповідний документ із теоретичним матеріалом. **Контроль знань інваріантної складової відбувається шляхом тестувань 1 і 2.**
- **Варіативна складова.** Це теми, специфічні для обраної спеціалізації. Теоретичний матеріал, посилання на зовнішні навчальні ресурси та список контрольних тем розміщено у відповідному файлі в середовищі Moodle. Контроль знань варіативної складової відбувається на захисних заняттях.

Кожна з 3x7 лабораторних робіт містить вказівки на головну та альтернативну ролі, термін виконання, необхідне програмне забезпечення, максимальну кількість балів, задачі для виконання (з яких студент може обрати цікаві для нього за умови погодження з іншими студентами, які обрали дану роль), теоретичний матеріал і вказівки з посиланням на зовнішні навчальні ресурси та список контрольних тем. Нижче наведено приклад Лаб. №1.6.

Лабораторна робота №1.6. Виділення функціональних особливостей. Розбиття задачі по функціональних складових.

Головна виконуюча роль	Бізнес-аналітик
Альтернативна виконуюча роль	Проектний менеджер
Термін виконання	Фаза 1: 1-5 тижні семестру
Програмне забезпечення	Пакет MS Office 365, draw.io.
Максимальна кількість балів	15 балів
Задачі для виконання у ЛР * обов'язкові для виконання у кожній команді	(5 б.) Сформулювати опис системи, функціональну задачу системи та опис ключових користувачів та їх дії. (5 б.)* Виділити функціональні особливості системи (великі функціональні блоки системи, епіки). (10 б.)* На основі функціональних блоків провести подальше розбиття на функціональні складові (фічі та юзер сторі) з вказанням відносної складності та пріоритетів для кожної складової. (5 б. за сценарій) Провести візуалізацію користувацької взаємодії за допомогою інструментів UML-діаграм чи їх аналогів.

4. Командна робота

Проект, який виконують студенти впродовж даного курсу, є достатньо великим, а тому його не можна виконати індивідуально

за відведений час. Ось чому вкрай важливою є командна робота і колективна відповідальність усіх за успіх фінального продукту.

Як було вказано вище, кожна лабораторна робота містить обов'язкові до виконання задачі, які прямо пов'язані з успіхом продукту. Якщо студенти, що обрали певну роль, **не виконали всіх обов'язкових складових (або, що те саме, не виконали їх до захисного заняття)**, то **максимальна оцінка за представлення фінального продукту (15 балів) буде пропорційно зменшена для всієї команди.**

Щоби знизити ці ризики, рекомендується обирати спеціалізації з деяким «підстрахуванням» – аби в разі, **якщо певний студент з якимось причини не виконав своє обов'язкове завдання, – хтось інший міг це завдання виконати.** Якщо виконання задач «за когось» відбувається в рамках однієї ролі, то узгодження з викладачем не потрібне. Якщо ж на певну спеціалізацію зареєструвався один студент, який не виконав свою задачу, то цю задачу може виконати представник іншої ролі (для більшості лабораторних вказано альтернативну роль). **Однак в такому разі необхідне узгодження з викладачем. Незалежно від кількості виконаної роботи, студент за одну фазу може одержати максимум 15 балів.**

Попри всі намагання формалізувати різні ролі по рівних фазах і балах за лабораторну, які прямо впливають із об'єму поставлених задач, розробка будь-якого проєкту є важко прогнозованою і часто нелінійною задачею. Тому для успіху проєкту як цілого вкрай важливий швидкий старт, який в основному залежить від задач таких ролей як бізнес-аналітик, архітектор програмного забезпечення та проєктний менеджер. Майже всі задачі першої фази для цих ролей потрібно виконати в перші 2–3 тижні. Переконливо рекомендуємо обирати для цих ролей найсильніших студентів.

Література

1. <https://e-learning.lnu.edu.ua/course/view.php?id=6957> та посилання у відповідних матеріалах.
2. Томас Д. Програміст-прагматик: друге видання / Томас Девід, Хант Ендрю. – 2024. – 368 с.
3. Мартін Р. Чиста архітектура. Мистецтво розробки програмного забезпечення / Мартін Роберт. – Фабула, Серія #PROSystem, 2019. – 368 с.
4. Graham D. Foundations of Software Testing. ISTQB Certification / Graham Dorothy, Black Rex, van Veenendaal Erik. – CENGAGE, 2018. – 290 p.
5. Піхлер Р. Agile продукт-менеджмент за допомогою Scrum / Піхлер Роман. – Фабула, Серія #PROSystem, 2019. – 128 с.
6. Wiegers K. Software Requirements (Developer Best Practices) 3rd Edition / Wiegers Karl, Beatty Joy. – Print2print, 2013. – 672 p.
7. Готельф Дж. Learn UX. Створення класних продуктів з командами Agile / Джеф Готельф, Джош Сейден. – ArtHuss, 2024. – 206 с.
8. Кім Дж. DevOps. Посібник / Кім Джин, Хамбл Джек, Дебуа Патрік, Вілліс Джон. – Фабула, Серія #PROSystem, 2023. – 384 с.
9. Kleppmann M. Designing Data-Intensive Applications / Kleppmann Martin. – O`Reilly Media, 2021. – 614 p.
10. Фрімен Е. Head First. Паттерни проектування / Е. Фрімен, Е. Робсон, К. Сієрра, Б. Бейтс. – Фабула, Серія #PROSystem, 2020. – 672 с.
11. Петтон Дж. Мапа історій користувача. Відкрий правдиву історію, створи саме той продукт / Петтон Джеф, Ікономі Пітер. – ArtHuss, Серія PRJCTR-teka, 2022. – 276 с.
12. Рейнвотер Г. Як пасти котів. Посібник для програмістів, які мають керувати іншими програмістами / Рейнвотер Дж. Генк. – Фабула, Серія #PROSystem, 2020. – 320 с.

ДОДАТОК А

Перелік теоретичних питань для підготовки до модульного контролю

1. Відповідальності розробника ПЗ.
2. Відповідальності інженера контролю якості (тестувальника).
3. Відповідальності проєктного менеджера.
4. Відповідальності DevOps.
5. Відповідальності архітектора ПЗ.
6. Відповідальності бізнес-аналітика.
7. Відповідальності UI/UX дизайнера.
8. Рефакторинг.
9. Юніт-тести.
10. Інтеграційні тести.
11. Тести продуктивності.
12. Smoke-тести.
13. Регресійне тестування.
14. Тест-стратегія.
15. Тест-план.
16. Тест-сценарій.
17. Пенетрейшн-тестування.
18. Менеджмент ризиків.
19. CI/CD-пайплайни.
20. Матриця відповідальностей RACI.
21. Діаграма Ганта.
22. Дорожня карта проєкту.
23. Задача фази пресейлу.
24. Складові компоненти контракту розробки ПЗ.
25. Типи контрактів: фіксована ціна, час-матеріали, окрема команда.
26. Задача фази дискавері.
27. Цілі та обсяг проєкту (Project Scope).
28. Поняття про PoC (Proof of Concept).
29. Складові і ціль документу BRD (Business Requirements Definition).
30. Складові і ціль документу FRD (Functional Requirements Definition).
31. Функціональні та нефункціональні вимоги.
32. Різновиди нефункціональних вимог.
33. Користувацький сценарій (Use Case).
34. Ієрархія задач: епіки, фічі та юзер-сторі.

37. Складові і ціль документу WBS (Work Breakdown Structure).
38. Універсальна мова UML.
39. Класова діаграма UML.
40. Діаграма послідовностей UML.
41. Діаграма сценаріїв UML.
42. Архітектурна діаграма.
43. Принципи побудови складу команди для проєкту.
44. Життєвий цикл програмного забезпечення (SDLC).
45. Каскадна методологія розробки (Waterfall).
46. Маніфест Agile. 12 принципів маніфесту Agile.
47. Поняття про скрам і канбан. Скрамбан.
48. Дошка канбану та її складові компоненти.
49. Тест-керована розробка (TDD).
50. Парне та екстремальне програмування.
51. Масштабування методології розробки: скрам скрамів і модель Spotify.
52. Мінімальний життєздатний продукт (MVP).
53. Задачі і функціонал систем проєктного трекінгу: Jira, Azure DevOps, Trello.
54. Метрики продуктивності команди.
55. Документ аналізу конкурентів.
56. SWOT аналіз.
57. Складові і ціль документу SRS (Service Requirement Specification).
58. Складові і ціль посібника користувача (User Guide).
59. Поняття про Test Item, Test Feature, Test Case та відмінності між ними.
60. Складові і ціль документу Test Summary Report.
61. Поняття про дизайн-систему та її складові.
62. Мокапи, ваєрфрейми та прототипи дизайну.
63. Шкала Лікерта.
64. Шкала NPS (Net Promoter Score).
65. Менеджмент ризиків та супутні документи. Матриця ризиків. Ймовірність та імпакт.
66. Технічна документація: OpenAPI, StoryBook та інші.
67. Складові і ціль документу Run Book.
68. Складові і ціль деплоймент- (ролбек-) планів.
69. Дорожня карта міграції.
70. Зустрічі в скрамі: щоденні стендапи, планування спринта, грумінг беклогу, рев'ю спринта, ретроспектива.
71. Беклог спринта і беклог продукту.

72. Метод аналогій оцінки тривалості проекту.
73. Метод оцінки тривалості проекту способом розбиття на частини.
74. Нисхідний метод оцінки тривалості проекту.
75. Метод експертної оцінки. Модель Кука.
76. Дельфійський метод.
77. Пленнінг-покер.
78. Три- і двоточкова оцінки.
79. Оцінки в годинах і сторі-пойнтах.

ДОДАТОК Б
Шаблон титульної сторінки звіту

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій

Кафедра оптоелектроніки
та інформаційних технологій

ЗВІТ
з лабораторної роботи № ____
«[НАЗВА РОБОТИ]»
з курсу «Проект з розробки системи опрацювання даних»

Виконав:
[ПРИЗВИЩЕ Ім'я студента], [група]
Перевірив:
[посада викладача] [ПРИЗВИЩЕ Ім'я викладача]

Львів [РІК]

ДОДАТОК В

Схема курсу

Тиж.	Тема, план, короткі тези	Форма діяльності (заняття)	Література, ресурси в Інтернеті	Завдання (год.)	Термін виконання
1	<p>Вступне заняття.</p> <p>Академічна доброчесність. Необхідність опрацювання великої кількості даних при експоненційному зростанні збереженої інформації.</p> <p>Виклики і проблеми обробки даних терабайтних масштабів. Дані і машинне навчання. Спеціалізації IT-спеціалістів.</p> <p>Відповідальності кожної зі сторін. RACI-матриця.</p> <p>Результати розробки.</p> <p>Діаграма Ганта та дорожня карта проекту.</p>	Практ.	[1], [5]	3 год.	Кінець поточного тижня
2	<p>Фаза пресеїлу та фаза дискавері. Типи контрактів.</p> <p>Функціональні і нефункціональні вимоги.</p> <p>BRD vs FRD. WBS.</p>	Практ.	[1], [6]	1 год.	Кінець поточного тижня
3	<p>Старт розробки.</p> <p>Архітектурна діаграма системи. Універсальна мова моделювання UML.</p> <p>Правильний вибір технологій та складу команди.</p>	Практ.	[1], [2], [3]	1 год.	Кінець поточного тижня
4	<p>Життєвий цикл ПЗ SDLC.</p> <p>Методології розробки ПЗ: каскадна модель, Agile та інші. Мінімальний життєвий продукт MVP.</p>	Практ.	[1], [5]	1 год.	Кінець поточного тижня

1-4	<p>Одна з лабораторних, залежно від ролі:</p> <p>Лаб. №1.1. Створення кістяків проектів. Налаштування кросс-катінгу в аспектно-орієнтованому підході.</p> <p>Лаб. №1.2. Розробка тест-плану, написання тест-кейсів до функціональних вимог.</p> <p>Лаб. №1.3. Старт проекту. Створення базової проектної документації.</p> <p>Лаб. №1.4. Налаштування гітхаб-політик для гілок. Створення docker-файлу і налаштування локального запуску.</p> <p>Лаб. №1.5. Побудова архітектурної діаграми. Вибір і обґрунтування обраних технологій</p> <p>Лаб. №1.6. Виділення функціональних особливостей. Розбиття задачі по функціональних частинах.</p> <p>Лаб. №1.7. Створення палітри кольорів, схеми розмірів та типографії.</p>	Практ. (Лаб.)	[1]	6 год.	Кінець тижня 5
5	Захисне заняття 1. Кінець фази дискавері.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня
6	Системи проектного трекінгу та їхні особливості. Трекінг прогресу задач. Канбан-дошка та її елементи. Дашборди з проектними метриками. Бурндаун-чарт і командна	Практ.	[1], [5]	1 год.	Кінець поточного тижня

	швидкість. Звітування робочого часу.				
7	Демо. Тестування №1.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня
8	Документація в розробці ПЗ I. Найкращі практики створення документації. Складові частини звіту про аналіз конкурентів. SWOT-аналіз. SRS. Test Items vs Test Features vs Test Cases. Test Summary Report. Дизайн-система. Васерфрейми, мокапи і прототипи.	Практ.	[1], [4], [5], [7]	1 год.	Кінець поточного тижня
9	Документація в розробці ПЗ II. Документація проєктного менеджера. Менеджмент ризиків. Технічна документація: OpenAPI, StoryBook, Run Book. Зовнішні звіти і звіти ассесменту. CI і CD.	Практ.	[1], [2], [3], [5], [8]	1 год.	Кінець поточного тижня
6–9	Одна з лабораторних, залежно від ролі: Лаб. №2.1. Написання логічної частини аплікації. Лаб. №2.2. Написання тест-кейсів і проведення нефункціонального тестування. Лаб. №2.3. Налаштування системи проєктного трекінгу. Лаб. №2.4. Налаштування хостингу. Лаб. №2.5. Створення WBS. Лаб. №2.6. Написання звіту про аналіз конкурентів.	Практ. (Лаб.)	[1]	6 год.	Кінець тижня 10

	Лаб. №2.7. Створення дизайн-системи окремих компонент.				
10	Захисте заняття 2. MVP.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня
11	Скрам. Спринт. Початок спринта. Спринт пленнінг. Щоденні стендапи. Грумінг беклогу. Рев'ю спринта. Ретроспектива. 1-1.	Практ.	[1], [4], [5]	1 год.	Кінець поточного тижня
12	Оцінка задачі. Оцінка в годинах і сторі-пойнтах. Методи оцінки. Метод аналогій. Метод розбиття на частини. Нисхідний метод. Метод експертної оцінки. Дельфійський метод. Триточкова оцінка.	Практ.	[1], [2], [4]	1 год.	Кінець поточного тижня
13	Робота з беклогом. Беклог спринта і беклог продукту. Пленнінг-покер. Аналіз стратегії розвитку продукту.	Практ.	[1], [5]	1 год.	Кінець поточного тижня
14	Реліз продукту. Представлення результатів.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня
11–14	Одна з лабораторних, залежно від ролі: Лаб. №3.1. Написання автоматизованих тестів. Лаб. №3.2. Створення звіту про результати тестування (Test Summary Report) Лаб. №3.3. Розробка проєктної документації і організація роботи команди. Лаб. №3.4. Створення	Практ. (Лаб.)	[1]	6 год.	Кінець тижня 15

	<p>CI/CD-пайплайнів.</p> <p>Лаб. №3.5. Створення Run Book.</p> <p>Лаб. №3.6. Розробка SRS.</p> <p>Лаб. №3.7. Створення мокапів базових сценаріїв.</p>				
15	Захисне заняття 3. Реліз продукту.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня
16	Заклучне заняття. Тестування №2. Анкетування.	Практ.	[1]-[12]	3 год.	Кінець поточного тижня