Чернівецький національний університет імені Юрія Федьковича

Факультет математики та інформатики

Кафедра прикладної математики та інформаційних технологій

«ЗАТВЕРДЖУЮ»
Декан факультету

_____ Ольга МАРТИНЮК
«____» _____ 20___ року

РОБОЧА ПРОГРАМА «FRONTEND-РОЗРОБКА WEB-ДОДАТКІВ»

вибіркова

Освітньо-професійна програма

Технології програмування та комп'ютерне моделювання

Спеціальність

F1 – Прикладна математика

Галузь знань

F – Інформаційні технології

Рівень вищої освіти перший (бакалаврський)

Факультет математики та інформатики

Мова навчання українська

Чернівці 2026 рік

Робоча програма навчальної дисципліни
«FRONTEND-РОЗРОБКА WEB-ДОДАТКІВ» **складена відповідно до** освітньо-професійної програми «Технології програмування та комп'ютерне моделювання».

Розробник:

Ковдриш Володимир Володимирович, асистент, к.ф.-м.н.

Викладач (чі), що забезпечує читання даної навчальної дисципліни:

Ковдриш Володимир Володимирович, асистент, к.ф.-м.н.

Затверджено на засіданні кафедри прикладної математики та інформаційних технологій

Протокол № ___ від «___» _____ 20__ року

Завідувач кафедри _____ **Ярослав БІГУН**

Схвалено методичною радою факультету

Протокол № ___ від «___» _____ 202__ року

Голова методичної ради _____ **Тоня ФРАТАВЧАН**

Мета навчальної дисципліни: Сформувати у студентів практичні навички розробки односторінкових веб-додатків з використанням React, забезпечити розуміння архітектурних підходів до організації інтерфейсу користувача, ефективного управління станом, маршрутизації, взаємодії з API та публікації застосунку.

Опис навчальної дисципліни

Загальна інформація

Форма навчання	Рік підготовки	Семестр	**Кількість**		**Кількість годин**						Вид підсумкового контролю
			кредитів	годин	лекції	практичні	семінарські	лабораторні	самостійна робота	індивідуальні завдання	
Денна	4	7	3	90	30	-	-	14	46	-	залік
Заочна	-	-	-	-	-	-	-	-	-	-	-

Структура змісту навчальної дисципліни

Назви змістових модулів і тем навчальних занять	**Кількість годин**											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		лекц.	**практ.**	**лаб.**	**інд.**	**срс**		**лекц.**	**практ.**	**лаб.**	**інд.**	**срс**
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. (КОМПОНЕНТИ, СТАН І ФОРМИ У REACT)												
Тема 1. Основи React і Vite. JSX, функціональні компоненти та props.	10	4		2		4						
Тема 2. Композиція компонентів, списки, ключі та оформлення інтерфейсу.	7	2		1		4						
Тема 3. Локальний стан, події та операції зі списком.	9	2		1		6						
Тема 4. Керовані форми, валідація, додавання та редагування відгуків.	12	4		2		6						
Тема 5. Спільний стан: Context API, Provider та useContext.	7	2		1		4						
Разом за ЗМ 1	**45**	**14**		**7**		**24**						

Продовження таблиці «Структура змісту навчальної дисципліни»

Назви змістових модулів і тем навчальних занять	Кількість годин											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		лекц.	**практ.**	**лаб.**	**інд.**	**срс**		**лекц.**	**практ.**	**лаб.**	**інд.**	**срс**
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 2. (МАРШРУТИЗАЦІЯ, СЕРВЕРНІ ДАНІ ТА ПУБЛІКАЦІЯ)												
Тема 1. React Router: вкладені маршрути, параметри та обробка помилок.	8	3		1		4						
Тема 2. HTTP, useEffect і взаємодія з json-server.	9	3		2		4						
Тема 3. Google Sheets та Google Apps Script: читання і зміна даних.	12	4		2		6						
Тема 4. Tailwind CSS та daisyUI: компоненти, теми й адаптивне оформлення.	9	4		1		4						
Тема 5. Стан входу, приватні маршрути та публікація застосунку.	7	2		1		4						
Разом за ЗМ 2	**45**	**16**		**7**		**22**						
Усього годин	**90**	**30**		**14**		**46**						

Тематика лекційних занять з переліком питань

№	Назва теми з основними питаннями
1	Основи React і Vite. JSX, функціональні компоненти та props. — 1) SPA, React і Vite; структура проєкту, залежності npm і запуск сервера розробки; 2) JSX: вирази JavaScript, фрагменти та умовне відображення; 3) Функціональні компоненти, імпорт і експорт модулів
2	Основи React і Vite. JSX, функціональні компоненти та props. — 1) Передавання чисел, рядків, об’єктів і масивів через props; 2) Композиція компонентів і використання children; 3) Відображення масивів через map та призначення ключів key
3	Композиція компонентів, списки, ключі та оформлення інтерфейсу. — 1) Компоненти списку відгуків, картки, заголовка та статистики; 2) Повторне використання Card, props і children; 3) CSS, обчислення кількості та середньої оцінки; початок роботи з useState
4	Локальний стан, події та операції зі списком. — 1) Події та передавання обробників через props; 2) Незмінне оновлення масивів: filter і функціональна форма setState; 3) Підтвердження видалення, очищення та відновлення списку
5	Керовані форми, валідація, додавання та редагування відгуків. — 1) Керована форма, onChange, onSubmit і preventDefault; 2) Перевірка довжини тексту, повідомлення та disabled; 3) Додавання відгуку, бібліотека uuid і вибір числової оцінки
6	Керовані форми, валідація, додавання та редагування відгуків. — 1) Круглий RatingSelect на основі радіокнопок; 2) Вибір відгуку за id, заповнення форми та скасування редагування; 3) Оновлення через map, сталість id та роль key у стані форми

№	Назва теми з основними питаннями
7	Спільний стан: Context API, Provider та useContext. — 1) Передавання даних через props і потреба у спільному контексті; 2) createContext, Provider, useContext та найближчий провайдер; 3) Перенесення стану й дій у FeedBackProvider; локальна чернетка форми
8	React Router: вкладені маршрути, параметри та обробка помилок. — 1) createBrowserRouter та RouterProvider; кореневий і вкладені маршрути; 2) Спільний макет, Outlet, Link і NavLink; 3) Розміщення Provider над маршрутизатором і збереження стану при навігації
9	React Router: вкладені маршрути, параметри та обробка помилок. / HTTP, useEffect і взаємодія з json-server. — 1) Параметри адреси, useParams, errorElement та useRouteError; 2) json-server, db.json, HTTP, Promise, async/await і JSON; 3) GET у useEffect, залежності ефекту, завантаження, помилки й повторне читання
10	HTTP, useEffect і взаємодія з json-server. — 1) POST із uuid; очікування відповіді та збереження чернетки при помилці; 2) PUT зі сталим id; DELETE і підтвердження користувача; 3) Серверні очищення та відновлення, часткове виконання й перечитування
11	Google Sheets та Google Apps Script: читання і зміна даних. — 1) Google Sheets: аркуш feedbacks, заголовки та типи даних; 2) Apps Script: SpreadsheetApp, getValues, doGet, ContentService і JSON; 3) Розгортання /exec, змінні VITE_gasURL та читання data.feedbacks у React
12	Google Sheets та Google Apps Script: читання і зміна даних. — 1) Додавання рядка через appendRow; uuid і кодування query-параметрів; 2) Пошук за id, getRange/setValues та deleteRow; 3) Параметр method і HTTP-метод; оновлення версії розгортання, серверний список і масові дії
13	Tailwind CSS та daisyUI: компоненти, теми й адаптивне оформлення. — 1) Підключення Tailwind CSS до Vite та утилітарний підхід; 2) Відступи, типографіка, flex/grid і адаптивні модифікатори; 3) Поетапне перенесення оформлення й узгодження з наявним CSS
14	Tailwind CSS та daisyUI: компоненти, теми й адаптивне оформлення. — 1) Підключення daisyUI, компоненти інтерфейсу та теми; 2) Оформлення навігації, карток, форм і сторінок; 3) Стани елементів, контрастність, адаптивність та узгодженість оформлення
15	Стан входу, приватні маршрути та публікація застосунку. — 1) Навчальні форми входу й реєстрації; спільний стан користувача; 2) Приватні маршрути, вихід і межі клієнтської перевірки доступу; 3) Production-збірка, конфігурація середовища та перевірка опублікованого SPA

Тематика лабораторних занять з переліком питань

№	Назва теми (завдання)
1	Компоненти та список відгуків (Створити компоненти відгуків, передати props і children, відобразити список зі сталими key та обчислити статистику.)
2	Оформлення та операції зі списком (Оформити картки й панель дій; реалізувати видалення, очищення та відновлення списку без мутації стану.)
3	Додавання й редагування відгуків (Реалізувати керовану форму, валідацію, uuid, круглий вибір оцінки, редагування за id та скасування.)
4	Контекст і маршрутизація (Перенести спільний стан у Provider; створити вкладені сторінки, навігацію, параметри адреси та обробку невідомого маршруту.)
5	Взаємодія з локальним API (Підключити json-server, реалізувати GET/POST/PUT/DELETE, стани завантаження і помилки та перевірити збереження після перезавантаження.)
6	Google Sheets як джерело даних (Створити навчальну таблицю і вебзастосунок Apps Script; підключити читання, додавання, редагування та видалення з React.)
7	Оформлення, стан входу й публікація (Оформити застосунок за допомогою Tailwind CSS і daisyUI; додати навчальний сценарій входу, зібрати та опублікувати SPA.)

Індивідуальні науково-дослідні завдання (ІНДЗ)

№	Завдання до тем
1	Інтерфейс керування відгуками — Створити компонентний застосунок із формою, валідацією, статистикою, редагуванням і спільним станом. Подати вихідний код та пояснити потік даних.
2	Застосунок із серверними даними — Підключити маршрутизацію та API, реалізувати операції з відгуками у локальному сервері й Google Sheets; продемонструвати обробку помилок.
3	Оформлення та публікація React-проєкту — Оформити інтерфейс із Tailwind CSS і daisyUI, реалізувати навчальний стан входу та опублікувати застосунок. Пояснити обмеження клієнтського захисту.

Завдання для самостійної роботи студентів

№	Назва теми	Завдання для самостійної роботи	К-сть год.
1	Основи React і Vite. JSX, функціональні компоненти та props.	Створити приклади JSX, компонентів, props і children; пояснити структуру проєкту Vite.	4
2	Композиція компонентів, списки, ключі та оформлення інтерфейсу.	Розділити інтерфейс на компоненти, оформити картки та дослідити сталі ключі списку.	4
3	Локальний стан, події та операції зі списком.	Реалізувати обробники подій і порівняти пряме та функціональне оновлення стану.	6
4	Керовані форми, валідація, додавання та редагування відгуків.	Розробити форму додавання й редагування з перевіркою даних; дослідити вплив key на локальну чернетку.	6
5	Спільний стан: Context API, Provider та useContext.	Побудувати приклади спільного стану через Context та вкладених провайдерів.	4
6	React Router: вкладені маршрути, параметри та обробка помилок.	Реалізувати вкладені маршрути, параметри URL, активну навігацію та сторінку помилки.	4
7	HTTP, useEffect і взаємодія з json-server.	Перевірити HTTP-запити, обробку помилок і часткове виконання масових операцій.	4
8	Google Sheets та Google Apps Script: читання і зміна даних.	Підготувати Apps Script для навчальної таблиці; перевірити типи даних, спецсимволи в тексті й оновлення розгортання.	6
9	Tailwind CSS та daisyUI: компоненти, теми й адаптивне оформлення.	Оформити компоненти за допомогою Tailwind CSS і daisyUI; перевірити теми й адаптивність.	4
10	Стан входу, приватні маршрути та публікація застосунку.	Реалізувати стан входу та приватну сторінку; перевірити production-збірку, вкладені адреси й налаштування API.	4

Разом за наведеними завданнями: 46 год.

Завдання навчальної дисципліни

Опанувати JSX, функціональні компоненти, props, children і композицію інтерфейсу.

Навчитися керувати станом, обробляти події, створювати форми та перевіряти введені дані.

Організовувати спільні дані через Context API та навігацію через React Router.

Виконувати HTTP-запити, обробляти JSON, завантаження й помилки; реалізовувати операції з даними на сервері.

Підключати Google Sheets через Google Apps Script і налаштовувати змінні середовища Vite.

Застосовувати Tailwind CSS і daisyUI; реалізовувати стан входу та готувати SPA до публікації.

Очікувані результати навчання

Створювати React-проєкт із Vite та організовувати компоненти, сторінки й допоміжні модулі.

Передавати дані через props і children, відображати списки зі сталими ключами.

Оновлювати стан без мутацій, додавати, редагувати й видаляти елементи масиву.

Створювати керовані форми, перевіряти введення і зберігати чернетку при помилці запиту.

Використовувати Context, Provider та useContext для спільного стану.

Налаштовувати вкладені маршрути, параметри адреси, навігацію та сторінку помилки.

Виконувати GET, POST, PUT і DELETE до json-server та узгоджувати інтерфейс із відповіддю сервера.

Читати й змінювати відгуки у Google Sheets через вебзастосунок Apps Script.

Оформлювати інтерфейс засобами CSS, Tailwind CSS та daisyUI.

Реалізовувати навчальний сценарій входу, збирати й публікувати застосунок та пояснювати його роботу.

Методи навчання

Лекції з поетапною демонстрацією коду і роботи застосунку, лабораторні роботи, аналіз помилок, контрольні питання, самостійні вправи та розробка індивідуального React-проєкту.

Система контролю та оцінювання

Методи контролю

1. перевірка та аналіз програмного коду
2. тестування функціональності програмного забезпечення
3. демонстрація роботи застосунків
4. усне опитування
5. оцінювання індивідуального науково-дослідного завдання

Форми контролю

1. поточний контроль (лабораторні роботи, тестові та індивідуальні завдання)
2. підсумковий контроль — залік (захист інтерактивного застосунку)

Розподіл балів, які отримують студенти

Поточне оцінювання (аудиторна та самостійна робота)										Кількість балів (залікова/екзаменаційна робота)	Сумарна к-ть балів
Змістовий модуль №1					**Змістовий модуль № 2**						
T1	T2	T3	T4	T5	T1	T2	T3	T4	T5	30	100
7	7	7	7	7	7	7	7	7	7		

Шкала оцінювання: національна та ЄКТС

Оцінка за національною шкалою	**Оцінка за шкалою ЄКТС**	
	Оцінка (бали)	**Пояснення за розширеною шкалою**
зараховано	A (90-100)	відмінно
	B (80-89)	дуже добре
	C (70-79)	добре
	D (60-69)	задовільно
	E (50-59)	достатньо
не зараховано	FX (35-49)	(незадовільно) з можливістю повторного складання
	F (1-34)	(незадовільно) з обов'язковим самостійним опрацюванням освітнього компоненту до перескладання

Перелік питань для самоконтролю та підсумкового контролю

Питання для підсумкового контролю

1. Чим односторінковий застосунок відрізняється від багатосторінкового сайту?
2. Яку роль виконують React, Vite, npm і package.json?
3. Як JSX поєднує розмітку й вирази JavaScript?
4. Як створюють, експортують та імпортують функціональний компонент?
5. Як передавати дані через props і чому їх не змінюють усередині компонента?
6. Для чого використовують children та композицію компонентів?
7. Як відобразити список через map і чому ключ key має бути сталим?
8. Як розподілити відповідальність між списком, карткою, формою та статистикою?
9. Чим стан відрізняється від props і звичайної локальної змінної?
10. Як працюють useState і функціональне оновлення стану?
11. Як передавати обробник події дочірньому компоненту?
12. Як filter і map допомагають оновлювати масив без мутацій?
13. Що означає керована форма і як працюють value та onChange?
14. Навіщо викликати preventDefault під час надсилання форми?
15. Як перевіряти введений текст і керувати disabled?
16. Навіщо використовують uuid та чому id не змінюється при редагуванні?

17. Як реалізувати RatingSelect на основі радіокнопок?
18. Як організувати вибір запису, редагування та його скасування?
19. Як key впливає на створення компонента й початковий стан форми?
20. Коли передавання через props стає незручним і допомагає Context?
21. Чим відрізняються createContext, Provider та useContext?
22. Яке значення отримає компонент за наявності кількох провайдерів одного контексту?
23. Як незалежні контексти співіснують у дереві компонентів?
24. Які дані форми доцільно залишити локальними?
25. Як createBrowserRouter і RouterProvider організовують маршрути?
26. Для чого потрібні вкладені маршрути та Outlet?
27. Чим відрізняються Link і NavLink?
28. Як отримати параметри адреси через useParams?
29. Для чого потрібні errorElement і useRouteError?
30. Чому місце Provider відносно маршрутизатора впливає на збереження стану?
31. Що повертає fetch і як прочитати JSON-відповідь?
32. Як взаємодіють Promise, async та await?
33. Для чого перевіряти response.ok?
34. Чому завантаження даних виконують у useEffect і як обрати залежності?
35. Як відображати завантаження, помилку та повторне читання?
36. Як відрізняються GET, POST, PUT і DELETE у json-server?
37. Коли можна очистити форму після надсилання запиту?
38. Як узгодити React-стан із результатом редагування чи видалення на сервері?
39. Що відбудеться, якщо послідовність масових запитів виконається лише частково?
40. Як getDataRange і getValues представляють дані Google Sheets?
41. Навіщо пропускати заголовки таблиці та узгоджувати типи id і rating?
42. Де виконується doGet і звідки надходить request.parameter?
43. Як ContentService повертає JSON і чому клієнт читає data.feedbacks?
44. Чим ID таблиці відрізняється від адреси розгортання /exec?
45. Як Vite використовує .env і чому VITE-змінні доступні браузеру?
46. Чим query-параметр method відрізняється від HTTP-методу запиту?
47. Навіщо кодувати значення параметрів через encodeURIComponent?
48. Як appendRow, getRange/setValues і deleteRow змінюють таблицю?
49. Чому запис треба шукати за id, а не за поточним індексом картки?
50. Чому після зміни Apps Script потрібно оновити версію розгортання?
51. У чому полягає утилітарний підхід Tailwind CSS?
52. Як daisyUI доповнює Tailwind CSS і як застосовуються теми?
53. Як перевірити адаптивність і стани елементів інтерфейсу?
54. Як організувати стан входу, вихід і приватний маршрут у навчальному застосунку?
55. Чому клієнтська перевірка доступу не замінює серверної авторизації?

56. Як перевірити production-збірку, вкладені маршрути та API після публікації?

Зарахування результатів неформальної освіти

Зарахування результатів неформальної освіти здійснюється відповідно до чинного положення університету про визнання результатів навчання.

Здобувач може подати підтверджувальні документи: сертифікати, проєкти, портфоліо або посилання на GitHub-репозиторії.

Рекомендована література

Основна

1. React — офіційна документація

Додаткова

1. MDN Web Docs — JavaScript
2. React Router — документація
3. Google Apps Script — вебзастосунки

Інформаційні ресурси

1. Tailwind CSS — документація
2. DaisyUI — документація

Політика академічної доброчесності

Дотримання політики академічної доброчесності регламентовано чинними документами університету.

- Кодекс академічної доброчесності та етики ЧНУ
- Положення про виявлення та запобігання академічному плагіату ЧНУ
- Політика використання штучного інтелекту в ЧНУ

Використання здобувачами освіти інструментів штучного інтелекту (ШІ)

1. Дозволяється за умови дотримання академічної доброчесності, позначення факту та способу використання ШІ і відповідальності здобувача за результат.
2. Не допускається використання ШІ як заміни самостійного виконання завдань або для порушення академічної доброчесності.
3. Викладач може визначати додаткові вимоги до використання ШІ для конкретної дисципліни.