Чернівецький національний університет імені Юрія Федьковича
(повне найменування закладу вищої освіти)

Факультет математики та інформатики
(назва факультету/навчально-наукового інституту)

Кафедра математичного моделювання
(назва кафедри, що забезпечує викладання)

"ЗАТВЕРДЖУЮ"
Декан факультету
математики та інформатики
_____ **Ольга МАРТИНЮК**
"28" серпня **2026 року**

РОБОЧА ПРОГРАМА
навчальної дисципліни
Сучасні клієнтські веб-технології
(назва навчальної дисципліни)

вибіркова
(вказати: обов'язкова)

Освітньо-професійна програма Комп'ютерні науки та проєктування програмних систем
(назва програми)

Спеціальність F3 Комп'ютерні науки
(вказати: код, назва)

Галузь знань F Інформаційні технології
(вказати: шифр, назва)

Рівень вищої освіти другий (магістерський)
(вказати: перший (бакалаврський) / другий (магістерський) / третій (освітньо-науковий)

Факультет математики та інформатики
(назва факультету/ навчально-наукового інституту, на якому здійснюється підготовка фахівців за вказаною освітньою програмою)

Мова навчання українська
(вказати: на якій мові читається дисципліна)

Чернівці 2026 рік

Робоча програма навчальної дисципліни «*Сучасні клієнтські веб-технології*» складена відповідно до освітньо-професійної програми «Комп'ютерні науки та проєктування програмних систем»

Розробник:
Мельник Василь Сергійович, асистент кафедри математичного моделювання, кандидат фіз.-мат. наук

Викладач, що забезпечує читання даної навчальної дисципліни:
Мельник Василь Сергійович, асистент кафедри математичного моделювання, кандидат фіз.-мат. наук

Затверджено на засіданні кафедри математичного моделювання
Протокол № 1 від «27» серпня 2026 року

Завідувач кафедри _____ **Ігор ЧЕРЕВКО**

Схвалено методичною радою факультету математики та інформатики
Протокол № 1 від «28» серпня 2026 року

Голова методичної ради _____ **Тоня ФРАТАВЧАН**

Пояснювальна записка

Мета навчальної дисципліни: Головною метою курсу “Сучасні клієнтські веб-технології” є вивчення основних прийомів і методів фулл-стак розробки, конструювання веб-аплікацій, освоєння важливих патернів розробки і ознайомлення з сучасними бібліотеками для розробки.

Цей курс розроблений з метою надання студентам глибокого розуміння та практичних навичок у сфері клієнтського веб-програмування. Студенти зможуть ознайомитися з останніми тенденціями у веб-розробці, включаючи HTML5, CSS3, та JavaScript ES6+. Студенти будуть навчатися створювати веб-додатки та інтерактивні інтерфейси з використанням сучасних клієнтських технологій.

Курс також охоплює адаптивний та мобільний дизайн, що є важливим аспектом сучасних веб-технологій. Курс надає можливість студентам реалізовувати власні проекти та вдосконалювати їхні навички веб-розробника.

Результати навчання:

Загальні компетенції

ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.

ЗК3. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.

ЗК5. Здатність розробляти проєкти та управляти ними.

Фахові компетенції

СК2. Здатність проєктувати архітектуру інформаційних систем.

СК7. Здатність управляти робочими процесами у сфері інформаційних технологій, які є складними, непередбачуваними та потребують нових стратегічних підходів.

СК8. Здатність розробляти і реалізовувати наукові та прикладні проєкти в галузі інформаційних технологій та дотичні до неї міждисциплінарні проєкти.

СК10. Здатність до самоосвіти та професійного розвитку.

Програмні результати навчання:

РН1. Спеціалізовані концептуальні знання, що включають сучасні наукові здобутки у сфері системного аналізу та інформаційних технологій і є основою для оригінального мислення та проведення досліджень.

РН2. Будувати та досліджувати моделі складних систем і процесів застосовуючи методи системного аналізу, математичного, комп’ютерного та інформаційного моделювання.

РН7. Розробляти інтелектуальні системи в умовах слабо структурованих даних різної природи.

РН10. Зрозуміло і недвозначно доносити власні знання, висновки та аргументацію до фахівців і нефахівців, зокрема до осіб, які навчаються.

РН11. Вільно презентувати та обговорювати усно і письмово результати досліджень та інновацій, інші питання професійної діяльності державною та англійською мовами.

Опис змісту робочої програми 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	1	2	3	90	14			16	60		залік
Заочна											

Структура змісту навчальної дисципліни

Назви змістових модулів і тем	Кількість годин											
	денна форма						Заочна форма					
	усього	у тому числі					усього	у тому числі				
		л	п	лаб.	інд.	с.р.		л	п	лаб	інд	с.р.
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. (Frontend and Backend development)												
Тема 1. React та сучасний CSS: JSX, компоненти, props, state, Hooks, керування станом, Sass та Less.	12	2		2		8						
Тема 2. ASP.NET Core та Blazor: компоненти, життєвий цикл, маршрутизація, навігація, data binding, обробка подій.	12	2		2		8						
Тема 3. Blazor: Dependency Injection, форми, валідація, керування станом, взаємодія компонентів та JavaScript Interop.	12	2		2		8						
Тема 4. Взаємодія клієнтської та серверної частин: Fetch, Axios, Web API, робота з даними та обробка помилок.	12	2		2		8						
Разом за ЗМ1	48	8		8		32						
Змістовий модуль 2. (Client-Server technologies and best practices)												
Тема 1. REST та GraphQL: проєктування API, queries, mutations, subscriptions.	12	2		2		8						
Тема 2. Автентифікація та авторизація: JWT, OAuth 2.0, захист клієнт-серверної взаємодії.	12	2		2		8						

Тема 3. WebSockets, WebHooks та безпека вебзастосунків: XSS, CSRF, CORS, OWASP Top 10.	9	1		2		8						
Тема 4. Оптимізація та тестування: продуктивність, lazy loading, unit та integration testing.	9	1		2		4						
Разом за ЗМ 2	42	6		8		28						
Усього годин	90	14		16		60						

Тематика лекційних занять з переліком питань

№	Назва теми з основними питаннями
1	**React: основи.** Архітектура React. JSX. Функціональні компоненти. Props. State. Рендеринг компонентів.
2	**React Hooks та керування станом.** Життєвий цикл компонентів. useState. useEffect. Local та global state. Lifting state. useReducer.
3	**CSS-препроцесори Sass та Less.** Змінні. Вкладеність. Mixins. Модульний CSS. Організація стилів у вебзастосунках.
4	**ASP.NET Core та Blazor.** .NET 8. ASP.NET Core. Blazor. Архітектура Blazor-застосунків. Компоненти Blazor.
5	**Blazor: маршрутизація та взаємодія з користувачем.** Routing. Navigation. Data binding. Event handling. Dependency Injection.
6	**Blazor: форми та валідація.** Створення форм. Model binding. Валідація даних. Валідаційні атрибути. Обробка результатів форм.
7	**Blazor: керування станом та взаємодія компонентів.** State management. Передача даних між компонентами. Життєвий цикл. JavaScript Interop.
8	**Взаємодія frontend та backend.** Fetch API. Axios. HTTP-запити. JSON. Web API. Підключення до бази даних. Обробка помилок.
9	**REST та проєктування API.** Принципи REST. HTTP-методи. URI. HTTP status codes. CRUD. Проєктування REST API.
10	**GraphQL.** Архітектура GraphQL. Schema. Types. Queries. Mutations. Subscriptions. Порівняння REST та GraphQL.
11	**Authentication та Authorization.** Автентифікація. Авторизація. JWT. Access та Refresh Tokens. OAuth 2.0. Захист API.
12	**WebSockets та WebHooks.** Принципи роботи WebSockets. Двонапрямлена комунікація. Інтеграція з WebSocket-серверами. WebHooks та сценарії їх використання.
13	**Безпека вебзастосунків.** XSS. CSRF. CORS. Основні загрози вебзастосункам. OWASP Top 10. Засоби захисту.
14	**Оптимізація продуктивності вебзастосунків.** Lazy loading. Code splitting. Кешування. Оптимізація JavaScript та CSS. Аналіз продуктивності.

15	**Тестування вебзастосунків.** Unit testing. Integration testing. Тестування компонентів. Mocking. Інструменти автоматизованого тестування.

Тематика лабораторних занять з переліком питань

№	Назва теми (завдання)
1	**Розроблення клієнтського застосунку на React.** Реалізувати компоненти з використанням JSX, props, state та React Hooks. Реалізувати керування станом та взаємодію компонентів.
2	**Розроблення клієнтського застосунку на Blazor.** Створити компоненти Blazor із маршрутизацією, data binding, обробкою подій, Dependency Injection, формами та валідацією.
3	**Інтеграція frontend та backend.** Розробити ASP.NET Core Web API та підключити до нього React або Blazor-застосунок із використанням Fetch/Axios. Реалізувати роботу з даними та обробку помилок.
4	**Розроблення захищеного та протестованого вебзастосунку.** Реалізувати JWT-аутентифікацію, налаштувати CORS, дослідити захист від XSS та CSRF, реалізувати WebSocket-взаємодію та написати unit/integration тести.

Завдання для самостійної роботи студентів

№	Назва теми	Завдання для самостійної роботи	К-сть год.
1	React та CSS-препроцесори	Опрацювати документацію React щодо компонентів, JSX, Hooks та керування станом. Дослідити Sass і Less та створити приклад модульної організації стилів.	15
2	Blazor та ASP.NET Core	Опрацювати документацію Blazor щодо компонентів, маршрутизації, data binding, Dependency Injection, форм та валідації. Реалізувати додатковий компонент із валідацією даних.	15
3	Client-Server технології	Дослідити REST, GraphQL, JWT, OAuth 2.0, WebSockets та WebHooks. Порівняти підходи до взаємодії клієнтської та серверної частин.	15
4	Безпека, продуктивність і тестування	Опрацювати OWASP Top 10, XSS, CSRF та CORS. Дослідити lazy loading та інші способи оптимізації. Реалізувати unit та integration тести для вебзастосунку.	15

Методи навчання

Лекції, лабораторні заняття, демонстрація прикладів програмного коду, проблемно-орієнтоване навчання, проєктне навчання, практичне виконання програмних завдань, аналіз типових помилок та проблемних ситуацій, самостійна

робота з технічною документацією, виконання індивідуальних завдань, тестування та аналіз результатів роботи вебзастосунків.

Система контролю та оцінювання

- **Методи контролю:** усне опитування, тестування, перевірка лабораторних робіт, перевірка самостійних та індивідуальних завдань, контрольні роботи, захист лабораторних робіт, оцінювання практичного проєкту.
- **Форми контролю:** поточний контроль, модульний контроль, підсумковий контроль у формі заліку.

Критерії оцінювання та підсумкового контролю

90–100 балів (відмінно): повне та системне засвоєння матеріалу, правильне виконання практичних і лабораторних завдань, здатність самостійно застосовувати сучасні вебтехнології та обґрунтовувати прийняті рішення.

82–89 балів (добре): достатнє засвоєння матеріалу, правильне виконання основних завдань, незначні помилки, здатність застосовувати набуті знання на практиці.

74–81 балів (добре): загалом правильне виконання завдань, розуміння основних понять і технологій, наявність окремих неточностей та потреба в незначній допомозі.

64–73 балів (задовільно): базове засвоєння матеріалу, виконання типових завдань із помилками, неповне розуміння окремих технологій.

60–63 балів (задовільно): мінімально достатній рівень знань, виконання основних завдань із суттєвими недоліками.

35–59 балів (незадовільно): недостатнє засвоєння матеріалу, значні помилки під час виконання практичних завдань.

1–34 бали (незадовільно): відсутність необхідних знань і практичних навичок, нездатність виконувати основні завдання дисципліни.

Розподіл балів, які отримують студенти

Поточне оцінювання (*аудиторна та самостійна робота*)								Кількість балів (залікова робота)	Сумарнак-ть балів
Змістовий модуль №1				Змістовий модуль № 2					
Т1	Т2	Т3	Т4	Т6	Т7	Т8	Т9		
5	5	10	10	5	10	10	5	40	100

Шкала оцінювання: національна та ЄКТС

100-бальна шкала	Оцінка за національною шкалою		Оцінка за шкалою ЄКТС	
			Оцінка	Пояснення за розширеною шкалою
90-100	Зараховано	Відмінно	A	відмінно
80-89		Добре	B	дуже добре
70-79			C	добре
60-69		Задовільно	D	задовільно
50-59			E	достатньо

35-49	Незараховано	Незадовільно	FX	(незадовільно) з можливістю повторного складання
1-34			F	(незадовільно) з обов'язковим самостійним повторним опрацюванням освітнього компонента до перескладання

Перелік питань для самоконтролю та підсумкового контролю навчальних досягнень студентів

1. Що таке React та які основні принципи його роботи?
2. Що таке JSX?
3. Що таке компоненти React?
4. У чому відмінність між props та state?
5. Що таке React Hooks?
6. Для чого використовуються useState та useEffect?
7. Що таке lifting state?
8. Для чого використовується useReducer?
9. Які підходи до керування станом застосовуються у React?
10. Що таке Sass та Less?
11. Які переваги використання CSS-препроцесорів?
12. Що таке Blazor?
13. Які типи компонентів використовуються в Blazor?
14. Як працює маршрутизація в Blazor?
15. Що таке data binding у Blazor?
16. Як здійснюється обробка подій у Blazor?
17. Що таке Dependency Injection?
18. Як реалізуються форми та валідація в Blazor?
19. Які підходи до керування станом використовуються у Blazor?
20. Для чого використовується JavaScript Interop?
21. Як React або Blazor взаємодіє з ASP.NET Core Web API?
22. Що таке Fetch API?
23. Що таке Axios?
24. Як здійснюється обробка помилок під час HTTP-запитів?
25. Що таке REST?
26. Які основні принципи REST?
27. Які HTTP-методи використовуються в REST API?
28. Що таке CRUD?
29. Як проєктується REST API?
30. Що таке GraphQL?
31. Що таке Query, Mutation та Subscription у GraphQL?
32. У чому відмінність REST від GraphQL?

33. Що таке автентифікація та авторизація?
34. Що таке JWT?
35. З яких частин складається JWT?
36. Що таке OAuth 2.0?
37. Що таке WebSocket?
38. Чим WebSocket відрізняється від HTTP?
39. Що таке WebHook?
40. Що таке CORS?
41. Що таке XSS?
42. Що таке CSRF?
43. Які основні загрози вебзастосункам визначає OWASP Top 10?
44. Які методи захисту від XSS та CSRF застосовуються у вебзастосунках?
45. Що таке lazy loading?
46. Що таке code splitting?
47. Які основні методи оптимізації продуктивності вебзастосунків?
48. Що таке unit testing?
49. Що таке integration testing?
50. У чому відмінність між unit та integration testing?
51. Які інструменти використовуються для тестування вебзастосунків?
52. Які критерії використовуються для оцінювання якості вебзастосунку?

Зарахування результатів неформальної освіти

Результати навчання, отримані здобувачем у процесі неформальної та/або інформальної освіти, можуть бути зараховані відповідно до нормативних документів закладу вищої освіти за умови відповідності отриманих компетентностей і результатів навчання змісту освітнього компонента. Зарахування результатів здійснюється на підставі документів, сертифікатів, результатів навчання або інших підтверджень, а за необхідності — за результатами співбесіди, тестування чи виконання індивідуального завдання. (https://www.chnu.edu.ua/media/4g5fzssb/poriadok-vyznannia-rezultativ-navchannia-zdobutykh-shliakhom-neformalnoi-ta-abo-informalnoi-osvity.pdf)

Рекомендована література

Основна

1. Freeman A. *Pro ASP.NET Core 8*. Apress.
2. Albahari J. *C# 12 in a Nutshell: The Definitive Reference*. O'Reilly Media.
3. Flanagan D. *JavaScript: The Definitive Guide*. O'Reilly Media.
4. Banks A., Porcello E. *Learning React: Modern Patterns for Developing React Apps*. O'Reilly Media.
5. React Documentation. *React Documentation*. React Documentation
6. Microsoft. *ASP.NET Core Documentation*. ASP.NET Core Documentation
7. Microsoft. *Blazor Documentation*. Blazor Documentation
8. Mozilla. *MDN Web Docs*. MDN Web Docs

Додаткова (допоміжна)

1. GraphQL Foundation. *GraphQL Documentation*. GraphQL Documentation

2. OWASP Foundation. *OWASP Top 10*. OWASP Top 10
3. OWASP Foundation. *Web Security Testing Guide*. OWASP Web Security Testing Guide
4. Microsoft. *ASP.NET Core Web API*. ASP.NET Core Web API
5. Microsoft. *ASP.NET Core SignalR*. ASP.NET Core SignalR
6. MDN. *Fetch API*. Fetch API
7. MDN. *WebSocket API*. WebSocket API
8. Testing Library. *Testing Library Documentation*. Testing Library Documentation
9. Vitest. *Vitest Documentation*. Vitest Documentation

Інформаційні ресурси

- React — офіційна документація
- React Learn — навчальні матеріали
- Microsoft Learn — ASP.NET Core
- Microsoft Learn — Blazor
- Microsoft Learn — ASP.NET Core Web API
- Microsoft Learn — SignalR
- MDN Web Docs
- MDN — HTTP
- MDN — Fetch API
- MDN — WebSocket API
- GraphQL — офіційна документація
- OWASP Top 10
- OWASP Web Security Testing Guide
- Testing Library
- Vitest
- Sass Documentation
- Less Documentation

Політика академічної доброчесності

Дотримання політики щодо академічної доброчесності учасниками освітнього процесу при вивченні освітнього компонента регламентовано чинними документами ЧНУ:

- «Етичний кодекс Чернівецького національного університету імені Юрія Федьковича» https://www.chnu.edu.ua/universytet/normatyvnidokumenty/etychnyi-kodeks-chernivetskoho-natsionalnoho-universytetuimeni-yuriia-fedkovycha/

- «Положенням про виявлення та запобігання академічного плагіату у Чернівецькому національному університеті імені Юрія Федьковича» https://www.chnu.edu.ua/universytet/normatyvni-dokumenty/polozhenniapro-vyiavlennia-ta-zapobihannia-akademichnomu-plahiatu-u-chernivetskomunatsionalnomu-universyteti-imeni-yuriia-fedkovycha/

- «Політика використання штучного інтелекту в Чернівецькому національному університету імені Юрія Федьковича» https://www.chnu.edu.ua/universytet/normatyvni-dokumenty/politykavykorystannia-

shtuchnoho-intelektu-v-chernivetskomu-natsionalnomuuniversyteti-imeni-yuriia-fedkovycha/

Використання здобувачами освіти інструментів штучного інтелекту (ШІ) під час виконання навчальних завдань:

- *допускається* за умови дотримання принципів академічної доброчесності (здобувач освіти несе відповідальність за зміст, достовірність і результати поданої роботи, а у разі використання ШІ зобов'язаний зазначити факт і спосіб його застосування);
- *не допускається* використання ШІ як заміни самостійного виконання завдань або з метою порушення вимог академічної доброчесності;
- *додаткові обмеження чи вимоги* щодо використання ШІ можуть визначатися викладачем в освітньому процесі для конкретних видів навчальної діяльності відповідно до особливостей дисципліни.