



Серверні веб-технології

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	Перший (бакалаврський)
Галузь знань	17 Електроніка, автоматизація та електронні комунікації
Спеціальність	174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Освітня програма	Автоматизація та комп'ютерно-інтегровані технології кібер-енергетичних систем
Статус дисципліни	Вибіркова
Форма навчання	Очна (денна)
Рік підготовки, семестр	3 курс, весняний семестр
Обсяг дисципліни	4 кредити ECTS (120 годин): 18 год лекцій, 36 год комп'ютерних практикумів, 66 год самостійної роботи
Семестровий контроль/ контрольні заходи	Залік
Розклад занять	https://schedule.kpi.ua
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: к.т.н., доц. Бунке Олександр Сергійович, bunke.oleksander@iit.kpi.ua Практичні / Семінарські: к.т.н., доц. Бунке Олександр Сергійович
Розміщення курсу	Google classroom

Програма навчальної дисципліни

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

Практично всі сучасні веб-сайти й мобільні застосунки звідкись отримують і кудись зберігають дані — ця непомітна для користувача компонента називається серверною, або «бек-ендом». Розробка серверних продуктів для веб базується на знаннях клієнт-серверної архітектури, баз даних, контейнеризації та сучасних мов і фреймворків бекенду. Якісно спроектована серверна частина визначає продуктивність, безпеку та масштабованість усього застосунку.

У цьому курсі будуть вивчатися: принципи роботи серверних веб-застосунків, клієнт-серверна архітектура та протокол HTTP, проектування програмних інтерфейсів (REST-API, OpenAPI), реляційні бази даних, сучасний синтаксис мови PHP 8.x та фреймворк Symfony, серверний JavaScript на Node.js та фреймворк NestJS, контейнеризація застосунків з Docker та підходи до неперервної інтеграції (CI/CD).

Предмет навчальної дисципліни — теоретичні та практичні основи розробки сучасних серверних WEB-додатків із використанням PHP, Symfony, NestJS, реляційної СУБД MySQL та контейнеризації засобами Docker.

Метою дисципліни є формування та розвиток у студентів таких компетентностей:

- ЗК01 – Здатність застосовувати знання у практичних ситуаціях;
- ЗК04 – Навички використання інформаційних і комунікаційних технологій;
- ЗК05 – Здатність до пошуку, опрацювання та аналізу інформації з різних джерел;

- ЗК08 – Здатність працювати в команді;
- ФК06 – Здатність використовувати для вирішення професійних завдань новітні технології у галузі автоматизації та комп'ютерно-інтегрованих технологій, зокрема, проектування багаторівневих систем керування, збору даних та їх архівування для формування бази даних параметрів процесу та їх візуалізації за допомогою засобів людино-машинного інтерфейсу;
- ФК09 – Здатність вільно користуватись сучасними комп'ютерними та інформаційними технологіями для вирішення професійних завдань, програмувати та використовувати прикладні та спеціалізовані комп'ютерно-інтегровані середовища для вирішення задач автоматизації;
- ФК14 – Здатність використовувати програмне забезпечення для задач автоматизації технологічних процесів на мовах програмування високого рівня з використанням сучасних технологій програмування.

У результаті вивчення навчальної дисципліни студенти набудуть таких загальних програмних результатів навчання:

- ПРН03 – Вміти застосовувати сучасні інформаційні технології та мати навички розробляти алгоритми та комп'ютерні програми з використанням мов високого рівня та технологій об'єктно-орієнтованого програмування, створювати бази даних та використовувати інтернет-ресурси;
- ПРН09 – Вміти проектувати багаторівневі системи керування і збору даних для формування бази параметрів процесу та їх візуалізації за допомогою засобів людино-машинного інтерфейсу, використовуючи новітні комп'ютерно-інтегровані технології;
- ПРН12 – Вміти використовувати різноманітне спеціалізоване програмне забезпечення для розв'язування типових інженерних задач у галузі автоматизації, зокрема, математичного моделювання, автоматизованого проектування, керування базами даних, методів комп'ютерної графіки.

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

Для успішного опанування дисципліни студент має володіти базовими знаннями з алгоритмів, прослухати дисципліни «сучасні технології програмування» та «проектування та розробка баз даних».

Результати навчання використовуються при дипломному проектуванні, у вибіркових дисциплінах подібного спрямування та у професійній діяльності.

3. Зміст навчальної дисципліни

Тема 1. Клієнт-серверна архітектура та протокол HTTP. Розглядаються принципи взаємодії веб- і мобільних клієнтів із серверною частиною, життєвий цикл «запит-відповідь», семантика методів HTTP/HTTPS, коди стану, заголовки, керування сесіями та автентифікаційні токени. Формується уявлення про програмні інтерфейси (API), формати обміну даними (JSON, XML), архітектурні принципи REST та підходи до документування інтерфейсів (OpenAPI/Swagger). Оглядом вводиться інструментарій бекенд-розробника: PhpStorm/VS Code, Postman, Docker Desktop, система контролю версій Git та платформа GitHub.

Тема 2. Типи веб-сайтів. Розглядається еволюція моделей побудови веб-застосунків: статичні сайти на зорі Інтернету, технологія AJAX, односторінкові застосунки (SPA, Single Page Application), повернення до серверного рендерингу (Server Side Rendering, SSR) та статичних генераторів сайтів (Static Site Generator, SSG). Порівнюються підходи з огляду на продуктивність, індексованість

пошуковими системами, складність розробки та користувацький досвід. Формується розуміння критеріїв вибору моделі під прикладну задачу.

Тема 3. Огляд мов для серверних застосунків. Розглядаються типи мов програмування, базові технології розмітки та стилізації (HTML, CSS) та найпоширеніші мови серверної розробки: Python, JavaScript, PHP, Java, C#. Порівнюються особливості моделей виконання (синхронне vs подієво-орієнтоване), типізація, екосистема пакетних менеджерів, типові області застосування. Формується розуміння критеріїв вибору мови й фреймворку для задачі: вимоги до пропускну здатності, наявність бібліотек предметної області, кадровий ринок, вартість супроводу.

Тема 4. Сервісна архітектура. Розглядаються основні принципи мікросервісної архітектури та її переваги порівняно з монолітним підходом, типові сценарії переходу від моноліту до мікросервісів. Висвітлюється контейнеризація як інструмент сервісної архітектури та брокери повідомлень як засіб асинхронної комунікації між сервісами. Формуються критерії вибору архітектури під прикладну задачу: продуктивність, складність експлуатації, командна структура.

Тема 5. Основи мови PHP. Висвітлюються базові конструкції PHP: змінні, константи, функції, посилання, масиви, генератори. Розглядається сучасний синтаксис PHP 8.x — типізовані параметри й значення, перерахування (enum), readonly-властивості, конструкція match, іменовані аргументи, атрибути — та об'єктно-орієнтована модель: класи, інтерфейси, абстрактні класи, трейти, видимість, успадкування. Окрему увагу приділено принципам чистого коду (найменування, функції, форматування, класи, обробка помилок, межі), практичним веб-прикладам і стандартам PSR.

Тема 6. Пакетний менеджер Composer. Розглядається інсталяція Composer на Linux/Unix/macOS та Windows, синтаксис та основні опції, структура файлів composer.json і composer.lock. Висвітлюється семантичне позначення версій (SemVer), оновлення та видалення пакетів, скидання автозавантаження, автозавантаження класів за стандартом PSR-4. Формується практика підключення сторонніх бібліотек до власного проєкту з репозиторію Packagist.

Тема 7. Фреймворк Symfony. Розглядаються основні компоненти Symfony, структура веб-додатку та процес обробки запиту, розгортання проєкту. Висвітлюються контролери та маршрутизація, об'єктно-реляційне відображення Doctrine ORM (сутності, репозиторії, міграції, Query Builder, DQL), сервіси та контейнер впровадження залежностей (Dependency Injection). Окрему увагу приділено валідації та серіалізації даних, а також тестуванню Symfony-застосунків.

Тема 8. Серверний JavaScript: Node.js. Розглядається проблема блокуючих введень/виведень і C10K, середовище виконання Node.js та модель «циклу подій» (event loop), обробка CPU-ємних завдань через воркери, директиви Import/Export, керування cookie та сесіями, пакетний менеджер npm і структура package.json. Висвітлюються основи TypeScript для серверної розробки. Вивчається фреймворк NestJS та Prisma ORM: створення проєкту та структура NestJS, побудова REST-API (контролери, сервіси), робота з базою даних через Prisma, валідація та DTO, гварди, перехоплювачі та JWT-аутентифікація.

Тема 9. Розгортання веб-додатків у мережі. Розглядається розгортання простих PHP-сайтів через FTP як вступний сценарій публікації. Висвітлюється контейнеризація з Docker: поняття образу та контейнера, синтаксис Dockerfile, оптимізація розміру образу (multi-stage builds, .dockerignore), мережі й томи, оркестрація багатоконтейнерного застосунку засобами Docker Compose. Вивчаються основи неперервної інтеграції та доставки (CI/CD) на прикладі GitLab CI та GitHub Actions: запуск лінійних і тестів, складання образів, публікація до реєстру.

Структура комп'ютерних практикумів. Дисципліна включає три комп'ютерні практикуми, кожен з яких є самостійним серверним застосунком на окремому стеку. Тема кожного застосунку обирається студентом самостійно з-поміж типових прикладних задач: каталог продукції з кошиком; бібліотека ресурсів зі списком уподобань і фільтрацією; панель моніторингу з агрегацією показників; сервіс обліку завдань тощо. Комп'ютерний практикум № 1 передбачає реалізацію REST-

API на фреймворку Symfony 7 з використанням Doctrine ORM, реляційної СУБД MySQL та документуванням інтерфейсу за стандартом OpenAPI. Комп'ютерний практикум № 2 — реалізацію REST-API мовою TypeScript на фреймворку NestJS 10+ з використанням Prisma, DTO-валідації та токен-аутентифікації; теми застосунків різні з метою порівняння стеків. Комп'ютерний практикум № 3 — контейнеризацію обох сервісів засобами Docker, оркестрацію через Docker Compose та налаштування CI-конвеєра у GitHub Actions (лінтери, тести, складання образів). Захист кожного практикуму проводиться у форматі індивідуальної співбесіди студента з викладачем, де демонструється робочий результат і Git-історія етапу.

4. Навчальні матеріали та ресурси

Базова література

1. Мартін Р. С. Чиста архітектура: пер. з англ. Київ : ТОВ «Фабула», 2019. 368 с.
2. Фрімен Е., Фрімен Е. Head First. Патерни проєктування. Київ : ТОВ «Фабула», 2020. 672 с.
3. Symfony Documentation : офіційний посібник з фреймворку. SensioLabs. URL: <https://symfony.com/doc/current/> (дата звернення: 01.05.2026).
4. NestJS Documentation : офіційний посібник з фреймворку. URL: <https://docs.nestjs.com/> (дата звернення: 01.05.2026).
5. PHP Manual : офіційна документація мови PHP. The PHP Group. URL: <https://www.php.net/manual/> (дата звернення: 01.05.2026).
6. Doctrine ORM Documentation : офіційна документація об'єктно-реляційного відображення. URL: <https://www.doctrine-project.org/projects/orm.html> (дата звернення: 01.05.2026).

Додаткова література

1. Prisma Documentation : сучасний type-safe ORM для Node.js та TypeScript. URL: <https://www.prisma.io/docs/> (дата звернення: 01.05.2026).
2. MySQL 8.0 Reference Manual : офіційна документація СУБД. Oracle Corporation. URL: <https://dev.mysql.com/doc/refman/8.0/en/> (дата звернення: 01.05.2026).
3. Docker Documentation : офіційна документація платформи контейнеризації. Docker Inc. URL: <https://docs.docker.com/> (дата звернення: 01.05.2026).
4. GitHub Actions Documentation : офіційна документація платформи неперервної інтеграції. GitHub Inc. URL: <https://docs.github.com/en/actions> (дата звернення: 01.05.2026).
5. PHP Standards Recommendations (PSR) : стандарти спільноти PHP-FIG. URL: <https://www.php-fig.org/psr/> (дата звернення: 01.05.2026).
6. TypeScript Handbook : офіційний посібник з мови. Microsoft. URL: <https://www.typescriptlang.org/docs/handbook/> (дата звернення: 01.05.2026).
7. OWASP API Security Top 10 : найпоширеніші вразливості програмних інтерфейсів. The OWASP Foundation. URL: <https://owasp.org/API-Security/editions/2023/en/0x00-header/> (дата звернення: 01.05.2026).

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Методи навчання включають:

1. Лекції: мультимедійні презентації з ілюстрацією прикладів на дошці;

2. Комп'ютерні практикуми: групові та індивідуальні виконання завдань в навчальній лабораторії з обладнанням або симуляторами, консультації стосовно проблемних питань
3. Проблемне навчання: вивчення, опис та представлення релевантної тематиці курсу проблеми.

Перелік тем лекційних занять:

№ заняття	Тема лекції
1	Тема 1. Клієнт-серверна архітектура та протокол HTTP
2	Тема 2. Типи веб-сайтів
3	Тема 3. Огляд мов програмування для розробки серверних застосунків
4	Тема 4. Сервісна архітектура веб-застосунків
5	Тема 5. Основи мови програмування PHP
6	Тема 6. Пакетний менеджер Composer
7	Тема 7. Фреймворк Symfony
8	Тема 8. Серверний JavaScript: Node.js
9	Тема 9. Розгортання веб-додатків у мережі Інтернет

Комп'ютерні практикуми використовуються для отримання та закріплення знань і практичних навичок, висвітлених в лекційній частині курсу.

Перелік тем і задач комп'ютерних практикумів:

№ заняття	Тема заняття
1-2	Інструктаж з охорони праці. Ознайомлення з PhpStorm/VS Code, Postman, Docker Desktop, системою контролю версій Git та платформою GitHub. Постановка завдань трьох комп'ютерних практикумів, ініціалізація репозиторіїв
3	Сучасний синтаксис PHP 8.x: типи, enum, readonly, match, атрибути
4	Composer: керування залежностями, автозавантаження PSR-4, сторонні бібліотеки
5-6	Symfony 7: маршрутизація, контролери, Doctrine ORM, міграції; реалізація CRUD-ендпойнтів та документування за OpenAPI. Захист комп'ютерного практикуму № 1
7-8	TypeScript для бекенду: типи, інтерфейси, декоратори, налаштування проєкту
9	NestJS 10: модулі, контролери, провайдери, впровадження залежностей
10	NestJS: Prisma, DTO та валідація через class-validator, обробка помилок
11	Гварди, перехоплювачі, JWT-аутентифікація на NestJS
12-13	Завершення REST-API на NestJS: міграції, валідація, документація. Захист комп'ютерного практикуму № 2
14-15	Контейнеризація: Dockerfile для PHP та Node.js застосунків, оптимізація образів (multi-stage builds)
16-17	Docker Compose для багатоконтейнерного середовища (Symfony+Nginx+MySQL, NestJS+MySQL); CI-конвеєр у GitHub Actions: лінери, тести, складання образів. Захист комп'ютерного практикуму № 3

Послідовність і тематика занять можуть змінюватися; про зміни студенти повідомляються заздалегідь через Google Classroom курсу.

6. Самостійна робота студента/аспіранта

Перед кожним лекційним заняттям, окрім першого, студент переглядає надані матеріали та рекомендовані джерела, що стосуються теми лекції.

Перед кожним комп'ютерним практикумом студент ознайомлюється з темою та завданнями, запланованими на заняття, повторює теоретичний матеріал.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Правила відвідування занять

Очікується, що студент відвідуватиме кожне аудиторне заняття, робитиме це вчасно, без запізень. У разі особливих обставин студент, за узгодженням з викладачем, може бути відсутнім на занятті, але це не може бути систематично.

Правила поведінки на заняттях

На заняттях передбачається активність студентів, пов'язана з навчальним процесом, включення в інтерактивні форми та методи навчання.

Під час комп'ютерних практикумів, окрім наявного в лабораторії обладнання, студент може користуватися власним ноутбуком.

Правила виконання і захисту комп'ютерних практикумів

Навчальна лабораторія обладнана комп'ютерною технікою для програмування веб-додатків. До виконання робіт допускаються лише студенти, які пройшли відповідний інструктаж з техніки безпеки та правил використання комп'ютерного обладнання.

Комп'ютерні практикуми виконуються бригадами з кількох студентів, сформованими на всю тривалість навчання дисципліни.

До захисту комп'ютерного практикуму допускаються студенти, які правильно виконали комп'ютерний практикум та мають підтвердження у вигляді завіреного викладачем протоколу. Захист проходить у форматі індивідуального спілкування студента з викладачем по тематиці роботи та виконання практичного завдання.

Правила призначення заохочувальних та штрафних балів

За узгодженням з викладачем студент може зробити доповідь по темі, релевантній дисципліні, виконувати додаткові завдання чи брати участь у модернізації методичного забезпечення дисципліни. Заохоченням до подібної успішної роботи є додаткові рейтингові бали загальним обсягом до 10 балів згідно "Положення про систему оцінювання результатів навчання в КПІ ім. Ігоря Сікорського" https://osvita.kpi.ua/sites/default/files/downloads/Pologennia_RSO_2025.pdf.

Політика дедлайнів та перескладань

Студент, який з будь-яких причин не зміг вчасно виконати та захистити комп'ютерний практикум, може це зробити на наступних практичних заняттях за умови доступності обладнання та часу у викладачів. Під час виконання та захисту комп'ютерних практикумів пріоритет надається студентам, які виконують завдання згідно календарного плану. Виконання та/або захист комп'ютерного практикуму після відведеного на неї строку призводить до зниження максимального балу, який студент може отримати за цей вид роботи.

Допуск до перескладання заходів семестрового контролю та самі перескладання здійснюються згідно правил Університету у терміни, визначені Університетом.

Політика щодо академічної доброчесності

Політика та принципи академічної доброчесності визначені у розділі 3 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: <https://kpi.ua/code>.

У разі виявлення порушення академічної доброчесності результати роботи студента, які стосуються недоброчесності, анулюються.

Норми етичної поведінки

Норми етичної поведінки студентів і працівників визначені у розділі 2 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: <https://kpi.ua/code>.

8. Види контролю та рейтингова система оцінювання результатів навчання (PCO)

Шкала оцінювання успішності вивчення дисципліни містить 100 балів та складається з наступних видів робіт:

- Комп'ютерні практикуми (3 × 25 балів): 75% рейтингу;
- модульна контрольна робота: 25% рейтингу.
- залік

Поточний контроль: Комп'ютерні практикуми, МКР.

Оцінюються наступні складові **комп'ютерного практикуму**:

- Якість Git-репозиторію та технічної документації (структура коміт-історії, README, OpenAPI-документація, тести) — max 5 балів;

- Захист комп'ютерного практикуму — демонстрація робочого результату та відповіді на питання викладача (max 20 балів):

- a. «Відмінно», повна відповідь (не менше 90% потрібної інформації) – 18–20 балів;
- b. «Добре», достатньо повна відповідь (не менше 75% потрібної інформації) – 14–17 балів;
- c. «Задовільно», неповна відповідь (не менше 60% потрібної інформації) – 10–13 балів;
- d. «Незадовільно», незадовільна відповідь – 0 балів;

Максимальна кількість балів за всі Комп'ютерні практикуми дорівнює 3 к.п. × 25 балів = 75 балів.

Модульна контрольна робота складається з двох частин тривалістю 1 ак. година кожна.

- "відмінно", повна відповідь (не менше 90% потрібної інформації) – 23–25 балів;
- "добре", достатньо повна відповідь (не менше 75% потрібної інформації), або повна відповідь з незначними неточностями – 18–22 балів;
- "задовільно", неповна відповідь (не менше 60% потрібної інформації) та незначні помилки – 10–17 балів;
- "незадовільно", незадовільна відповідь (не відповідає вимогам на 5 балів) – 0 балів.

Максимальна кількість балів за МКР дорівнює 25 балів.

Календарний контроль: проводиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу.

Для успішного проходження першого календарного контролю студент має успішно захистити 1 комп'ютерний практикум. Для успішного проходження другого календарного контролю студент має успішно захистити 2 Комп'ютерні практикуми.

Семестровий контроль: **залік**.

Умови допуску до семестрового контролю: зарахування усіх комп'ютерних практикумів, семестровий рейтинг більше 60 балів.

Студенти, які наприкінці семестру мають менше 60 балів, а також ті, хто бажає підвищити свою оцінку в системі ECTS, виконують залікову роботу у вигляді відповідей на теоретичні питання, виконання практичного завдання та співбесіди за результатами. Здобувач, який складав залік для підвищення своєї рейтингової оцінки за його результатами отримав меншу кількість балів за кількість балів, отриманих за результатами заходів поточного контролю, то у цьому випадку застосовується «м'який» підхід – зберігається вища оцінка. Залік проходить у формі співбесіди. Кожне завдання містить два теоретичних і одне практичне запитання. Перелік запитань надається студентам на консультації до заліку.

Автоматичний залік виставляється якщо протягом семестру студентом набрано ≥ 60 балів.

Кожне теоретичне запитання оцінюється в 10 балів, а практичне – 20 балів.

Система оцінювання теоретичних питань:

- "відмінно" – повна відповідь (не менше 90% потрібної інформації) – 9-10 балів;
- "добре" – достатньо повна відповідь (не менше 75% потрібної інформації) або відповідь з незначними неточностями – 7-8 балів;
- "задовільно" – неповна відповідь (не менше 60% потрібної інформації) та незначні помилки – 5-6 балів;
- "незадовільно" – незадовільна відповідь (не відповідає вимогам на "задовільно") – 0 балів.

Система оцінювання практичного завдання:

- "відмінно" – повне, безпомилкове розв'язування завдання – 18-20 балів;
- "добре" – повне розв'язування завдання з несуттєвими неточностями – 15-17 балів;
- "задовільно" – завдання виконане з певними недоліками – 12-14 балів;
- "незадовільно" – завдання невиконане або виконано невірно – 0 балів.

Максимальна кількість балів за залік дорівнює $10 \times 2 + 20 = 40$ балів.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

Кількість балів	Оцінка
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

9. Додаткова інформація з дисципліни (освітнього компонента)

Студенти будуть забезпечені необхідним програмним забезпеченням (демо-версії), яке працює у середовищі операційної системи Windows/Linux.

Є можливість зарахування сертифікатів проходження дистанційних чи онлайн курсів за відповідною тематикою як певна кількість балів рейтингу. Кількість балів визначається після аналізу програми, тривалості та результатів курсу, вказаного у сертифікаті.

Робочу програму навчальної дисципліни (силабус):

Складено к.т.н., доцентом Бунке Олександром Сергійовичем

Ухвалено кафедрою АЕП НН ІАТЕ (протокол № 21 від 15.06.2026)

Погоджено Методичною комісією НН ІАТЕ (протокол № ____ від __.__.2026)