МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

Робоча програма
навчальної дисципліни

"РОЗРОБКА *WEB*-ДОДАТКІВ"

для студентів напряму підготовки
6.051501 "Видавничо-поліграфічна справа"
всіх форм навчання

Харків
ХНЕУ ім. С. Кузнеця
2016

УДК 004.42(07.034)
Р58

Укладач В. М. Гіковатий

Затверджено на засіданні кафедри комп'ютерних систем та технологій. Протокол № 1 від 31.08.2015 р.

Самостійне електронне текстове мережеве видання

Робоча програма навчальної дисципліни "Розробка *Web*-
Р58 додатків" для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" всіх форм навчання [Електронний ресурс] / уклад. В. М. Гіковатий. – Харків : ХНЕУ ім. С. Кузнеця, 2016. – 42 с.

Подано тематичний план навчальної дисципліни та її зміст за модулями й темами. Вміщено плани лекцій та лабораторних занять, матеріали для закріплення знань (завдання для самостійної роботи, контрольні запитання), критерії оцінювання знань студентів, професійні компетентності, якими має володіти студент після вивчення дисципліни.

Рекомендовано для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" всіх форм навчання.

УДК 004.42(07.034)

Вступ

Важливість мережі Інтернет для розвитку суспільства складно переоцінити. Все більша кількість додатків мають за обов'язкову вимогу реалізацію у мережі Інтернет. З розвитком технологій створення *Web*-додатків у команді розробників сформувалися певні спеціалізації, одною з яких є *frontend developer*. Ці фахівці відповідають за розроблення клієнтської частини *Web*-додатка, яка забезпечує взаємодію користувача з основним функціоналом системи у браузері. В обов'язки *frontend developer* входить проектування та реалізація користувальницького інтерфейсу *Web*-додатка, забезпечення взаємодії клієнтської та серверної частин *Web*-додатка, а також проектування та реалізація взаємодії користувача та системи тощо.

Вирішення більшості із цих завдань входить до області професійних компетентностей студентів напряму підготовки "Видавничо-поліграфічна справа". Саме тому навчальну дисципліну побудовано таким чином, щоб сформувати у студентів систему знань, вмінь та навичок, що дозволить йому займати посаду *frontend developer*.

Навчальна дисципліна "Розробка *Web*-додатків" є вибірковою навчальною дисципліною та вивчається згідно з навчальним планом підготовки фахівців освітнього ступеня "бакалавр" напряму підготовки "Видавничо-поліграфічна справа" для всіх форм навчання.

1. Опис навчальної дисципліни

<table>
<tr><td rowspan="2">Найменування показників</td><td rowspan="2">Галузь знань, напрям підготовки, освітній ступінь</td><td colspan="2">Характеристика навчальної дисципліни</td></tr>
<tr><td>денна форма навчання</td><td>заочна форма навчання</td></tr>
<tr><td>Кількість кредитів – 4</td><td>Галузь знань 0515 "Видавничо-поліграфічна справа"</td><td colspan="2">За вибором</td></tr>
<tr><td rowspan="2">Змістових модулів – 2</td><td rowspan="4">Напрям підготовки 6.051501 "Видавничо-поліграфічна справа"</td><td colspan="2">Рік підготовки</td></tr>
<tr><td>4-й</td><td>4-й</td></tr>
<tr><td rowspan="2">Загальна кількість годин – 144</td><td colspan="2">Семестр</td></tr>
<tr><td>7-й</td><td>7-й</td></tr>
<tr><td rowspan="10">Тижневих годин для денної форми навчання:
аудиторних – 3,5;
самостійної роботи студента – 4,9</td><td rowspan="10">Освітній ступінь: бакалавр</td><td colspan="2">Лекції</td></tr>
<tr><td>26 год</td><td>13 год</td></tr>
<tr><td colspan="2">Лабораторні</td></tr>
<tr><td>34 год</td><td>17 год</td></tr>
<tr><td colspan="2">Самостійна робота</td></tr>
<tr><td>84 год</td><td>42 год</td></tr>
<tr><td colspan="2">Вид контролю</td></tr>
<tr><td colspan="2">екзамен</td></tr>
</table>

Співвідношення кількості годин аудиторних занять до самостійної і індивідуальної роботи становить для денної форми навчання – 71 %.

2. Мета та завдання навчальної дисципліни

Метою викладання даної навчальної дисципліни є формування теоретичних знань, практичних вмінь та навичок щодо розробки *Web*-додатків.

Для досягнення мети поставлені такі основні **завдання:**

засвоєння основних принципів побудови *Web*-додатків, етапів їх розробки, а також інструментальних засобів процесу розроблення *Web*-додатків;

оволодіння практичними навичками вирішення типових завдань, які виникають у процесі розробки *Web*-додатків.

"Розробка *Web*-додатків" – навчальна дисципліна, що вивчає підходи та засоби як до реалізації типової функціональності *Web*-додатків,

так і до автоматизації окремих технологічних операції власне процесу розробки.

Об'єктом навчальної дисципліни є процес розробки *Web*-додатків.

Предметом навчальної дисципліни є інструментальні засоби розробки *Web*-додатків та засоби автоматизації процесу їх розробки.

Вивчення даної навчальної дисципліни студент розпочинає, прослухавши більшість навчальних дисциплін професійного циклів. Теоретико-методологічною базою вивчення цієї дисципліни є такі навчальні дисципліни, як: "Основи програмування та алгоритмізації", "Об'єктно-орієнтоване програмування", "*Web*-дизайн", "Скриптові мови". Зі свого боку знання із цієї дисципліни забезпечують успішне виконання тренінгів, міждисциплінарних комплексних курсових робіт, бакалаврських та магістерських дипломних робіт.

У процесі навчання студенти отримують необхідні знання під час лекційних занять. Практичні навички, якими мають оволодіти студенти винесені на лабораторні заняття. Також велике значення в процесі вивчення та закріплення знань має самостійна робота студентів. Усі види занять розроблені відповідно до кредитно-трансферної системи організації навчального процесу.

У результаті вивчення навчальної дисципліни студент повинен:

знати:

понятійний та категорійний апарат щодо процесу розробки *Web*-додатків;

основні етапи розробки *Web*-додатків;

основні принципи та підходи щодо побудови *Web*-додатків;

типові завдання, які мають вирішувати *Web*-додатки;

типові завдання, які виникають у процесі розробки *Web*-додатків;

вміти:

здійснювати комунікацію з замовником у процесі формування вимог до *Web*-додатка, що розробляється;

здійснювати комунікацію з іншими членами команди розробників та використовувати засоби спільної роботи над проектом;

верстати ключові сценарії *Web*-додатка засобам мови розмітки *HTML*;

реалізовувати представлення клієнтської частини *Web*-додатка;

проектувати та реалізовувати взаємодію користувача та *Web*-додатка: опрацювання дій користувача;

реалізовувати API серверної частини *Web*-додатка за принципами *REST*;

реалізовувати взаємодію клієнтської та серверної частин *Web*-додатка засобами асинхронних запитів.

У процесі викладання навчальної дисципліни основна увага приділяється оволодінню студентами професійними компетентностями, що наведені в табл. 2.1.

Таблиця 2.1

Професійні компетентності, які отримують студенти після вивчення навчальної дисципліни

Код компетентності	Назва компетентності	Складові компетентності
2a8a4020-3e7d-40a6-b2ca-d34a36e40e8e	Формування екосистеми розробки *Web*-додатків	Знати основні етапи розробки *Web*-додатків та завдання, які вирішуються на кожному з них
		Використовувати сучасними засобами екосистеми розробки *Web*-додатків
		Налаштовувати екосистему розробки *Web*-додатків на автоматизацію останнього
eb983cb3-3227-4c25-bcb3-c6d8a5180a86	Розробляти клієнтську частину *Web*-додатків	Верстати подання *Web*-додатка засобами *HTML*/CSS
		Здійснювати відображення та форматування даних
		Реалізовувати взаємодію між користувачем та *Web*-додатком
		Проектувати модель даних та подання *Web*-додатка і здійснювати опрацювання даних засобами *JavaScript*
		Реалізовувати взаємодію між клієнтської та серверною частинами *Web*-додатка
e5412890-496a-4124-a461-126bc1c35bec	Розробляти *Web API*	Проектувати *Web API* серверної частини *Web*-додатка
		Реалізовувати основні методи *Web API*
		Налаштовувати маршрути *Web API*

Структуру складових професійних компетентностей та їх формування відповідно до Національної рамки кваліфікацій України наведено в додатку А.

3. Програма навчальної дисципліни

Змістовий модуль 1
Розробка клієнтської частини *Web*-додатка

Тема 1. Вступ до розробки *Web*-додатків

1.1. Поняття: *Web*-додаток, клієнтська частина *(front-end)*, серверна частина (*back-end*).

1.2. Сучасні вимоги до розробників *front-end*. Ніша фахівців напряму підготовки "Видавничо-поліграфічна справа" на ринку *IT*. Огляд вимог до розробників клієнтської частини *Web*-додатка.

Переваги та недоліки *Web*-додатків. Проблеми розробки. Визначення успішних проектів та їх статистика. Чинники неуспіху проектів: неконтрольовані зміни, недостатня надійність, недостатнє керування.

1.3. Етапи процесу розробки *Web*-додатка. Аналіз вимог. Проектування програмного забезпечення. Кодування. Налагодження та тестування. Документування. Упровадження.

1.4 Моделі процесу розробки *Web*-додатка. Каскадна модель. Екстремальне програмування.

Тема 2. Формування екосистеми розробки *Web*-додатків

2.1. Екосистема процесу розробки *Web*-додатків.

Поняття "екосистема процесу розробки *Web*-додатків". *Node.js*: призначення, інсталяція. *Git Bash*: призначення, інсталяція, основи роботи. Менеджер пакетів *npm*: інсталяція сторонніх пакетів.

2.2. Середовище розроблення клієнтської частини *Web*-додатків.

Інсталяція *Sublime Text 2*. Інсталяція плагінів.

2.3. Автоматизація розгортання *Web*-додатка.

Сутність процесу розгортання та її основні операції. Інструменти розгортання *Web*-додатка. Інсталяція *Grunt JS*. Налаштування *Grunt JS*. Перевірка автоматизованого процесу розгортання засобами *Grunt JS* на тестовому проекті.

2.4. Пакетний менеджер *Bower*.

Інсталяція. Пошук, інсталяція, оновлення та вилучення бібліотек засобами *Bower*. Налаштування конфігураційного файлу. Версії бібліотек. Інтеграція *Bower* та *Grunt JS*.

2.5. Контроль версій.

Призначення систем контролю версій. Основні поняття систем контролю версій. Система контролю версій *Git*. Схема життєвого циклу змін. Основні команди система контролю версій *Git*. Інтеграція *Bower* та *Grunt JS*.

2.6. Налагодження коду.

Процес налагодження коду. Інструментальні засоби браузерів для дослідження *Web*-сторінок та налагодження коду. *Batarang*: спеціалізований засіб налагодження *Web*-додатків із використанням *AngularJS*.

Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS*

3.1. Загальні відомості про додаток-приклад.

Призначення додатка. Функціональність та вимоги.

3.2. Архітектура *Web*-додатка.

Принципи REST. Шаблон "модель – подання – контролер" (*model-view-controller, MVC*). Стек технологій *Web*-додатка.

3.3. Ознайомлення з *AngularJS*.

Загальні відомості про фреймворк *AngularJS*. Спільнота та навчальні матеріали у мережі Інтернет. Реалізація MVC у *AngularJS*. Моделі. Подання. Контролери. Модулі та впровадження залежностей.

3.4. Організація файлів і каталогів.

Каталоги *Web*-додатка. Каталоги з вихідним кодом. Угоди по іменуванню файлів. Модулі и файли *AngularJS*.

3.5. Приклад першого додатка. Ознайомлення з директивами *Angular JS*. Угоди щодо назв директив у *AngularJS*. Директиви *ngApp*, *ngInit*, {{}}, *ngInclude*, *ngClick*. Відображення результатів обчислення виразів. Відображення значень за допомогою *ngBind*.

Тема 4. Відображення та форматування даних

4.1. Відображення за умовою.

Директиви *ngShow*, *ngHide*, *ngSwitch*, *ngIf*. Включення блоків вмісту за умовою.

4.2. Відображення колекцій.

Директива *ngRepeat*. Спеціальні змінні директиви *ngRepeat*. Прийоми використання директиви *ngRepeat*. Форматування рядків таблиці. Директива *ngClass*. Застосування директиви *ngRepeat* до безлічі елементів *DOM*.

4.3. Фільтри.

Застосування вбудованих фільтрів. Фільтри форматування. Фільтри перетворення масивів. Фільтрація за допомогою *filter*. Підрахунок елементів після фільтрації. Сортування за допомогою фільтра *orderBy*. Створення власних фільтрів – реалізація посторінкового виведення.

Тема 5. Взаємодія між користувачем та *Web*-додатком

5.1. Обробники подій.

Поняття події. Вбудовані директиви для оброблення подій *AngularJS*: *ngClick*, *ngDblClick*, *ngMousedown*, *ngMouseup*, *ngMouseenter*, *ngMouseleave*, *ngMousemove*, *ngMouseover*, *ngKeydown*, *ngKeyup*, *ngKeypress*. Директива *ngChange*. Передача параметрів до обробник події.

5.2. Директива *ngModel*.

Поняття двонаправленого зв'язку між моделлю та елементами HTML. Використання директиви *ngModel*. Механізм зв'язування даних у *ngModel*.

5.3. Директива введення даних.

Директива перевірки обов'язкової наявності значення. Текстові елементи введення. Індикативні перемикачі (*checkbox*). Альтернативні перемикачі (*radio buttons*). Низпадальні списки (*select*, *drop-down list*).

5.4. Додавання динамічної поведінки.

Стеження за змінами даних. Виведення повідомлень про помилки. Зміна доступності елементів подання.

5.5. Типові подання *Web*-додатка.

Додавання нових даних. Редагування обраного запису. Вилучення обраного запису.

Змістовий модуль 2
Розробка серверної частини *Web*-додатка

Тема 6. Розробка *API* серверної частини *Web*-додатка

6.1. Вступ до *Web API*.

Процес створення *Web API MVC* проекту засобами *Visual Studio 2010*. Створення *API*-контролерів. Угоди щодо назв методів.

6.2. Маршрутизація в *Web API*.

Налаштування маршрутів.

6.3. Реалізація основних методів *AP*-контролера серверної частини *Web*-додатка та його тестування.

Створення моделей даних. Реалізація методів *GET, POST, PUT, DELETE*. Інструментальні засоби тестування *Web API*.

Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером

7.1. Асинхронні запити.

Мета використання та принцип роботи асинхронних запитів. Види асинхронних запитів (запити *GET, POST, PUT, DELETE, HEAD*). Формати даних під час взаємодії із сервером.

7.2. Служба *$http*.

Методи служби *$http.* Перетворення даних запиту. Опрацювання HTTP-відповідей. Перетворення даних відповіді.

7.3. *Promise API* иі служба *$q*.

Отримання відкладених результатів засобами служби *$q*. Об'єднання асинхронних запитів у ланцюжки. Об'єднання відкладених результатів. Використання *$q* в *AngularJS*.

7.4. Пример реалізації взаємодії із сервером засобами *AngularJS*.

Тема 8. Реалізація специфічних вимог до *Web*-додатків

8.1. Використання сторонніх бібліотек.

Ресурси в мережі Інтернет. Критерії вибору сторонніх бібліотек.

8.2. Навігація у *Web*-додатках.

Адреса *URL* в односторінкових *Web*-додатках. Служба *$route*. Визначення основних маршрутів. Відображення змісту маршруту. Гнучке складання маршрутів. Доступ до значень параметрів маршрутів. Запобігання зміні маршруту.

8.3. Створення інтернаціональних *Web*-додатків.

Використання національних наборів символів і налаштувань. Стратегії перекладів користувальницького інтерфейсу, їх аналіз та реалізація.

4. Структура навчальної дисципліни

Із самого початку вивчення навчальної дисципліни кожен студент має бути ознайомлений як з робочою програмою навчальної дисципліни і формами організації навчання, так і зі структурою, змістом та обсягом кожного з її навчальних модулів, а також з усіма видами контролю та методикою оцінювання сформованих професійних компетентностей.

Вивчення студентом навчальної дисципліни відбувається шляхом послідовного і ґрунтовного опрацювання навчальних модулів. Навчаль-

ний модуль – це окремий, відносно самостійний блок дисципліни, який логічно об'єднує кілька навчальних елементів дисципліни за змістом та взаємозв'язками. Тематичний план дисципліни складається з двох змістових модулів (табл. 4.1).

Таблиця 4.1

Структура залікового кредиту навчальної дисципліни

Назви змістових модулів і тем	Кількість годин				
	усього	у тому числі			
		лекційні	лабораторні	самостійна робота	
				проведення підсумкового контролю	підготовка до занять
Змістовий модуль 1 Розробка клієнтської частини *Web*-додатка					
Тема 1. Вступ до розробки *Web*-додатків	14	2	4	–	8
Тема 2. Формування екосистеми розробки *Web*-додатків	21	–	–	–	21
Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS*	18	4	6	–	8
Тема 4. Відображення та форматування даних	20	6	6	–	8
Тема 5. Взаємодія між користувачем та *Web*-додатком	20	6	6	–	8
Разом за змістовим модулем 1	**93**	**18**	**22**	**–**	**53**
Змістовий модуль 2 Розробка серверної частини *Web*-додатка					
Тема 6. Розробка API серверної частини *Web*-додатка	16	4	4	–	6
Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером	16	4	4	–	6
Тема 8. Реалізація специфічних вимог до *Web*-додатків	13	–	4	–	5
Разом за змістовим модулем 2	**45**	**8**	**12**	**–**	**17**
Підготовка до екзамену	9	–	–	–	9
Передекзаменаційні консультації	2	–	–	2	–
Екзамен	3	–	–	3	–
Усього годин за модулями	**144**	**26**	**34**	**84**	

5. Теми лабораторних занять

Лабораторне заняття – це форма навчального заняття, за якої студент під керівництвом викладача проводить розробку власного *Web*-додатка. У ході лабораторних робот студент набуває професійних компетентностей та практичних навичок роботи з засобами розробки *Web*-додатків. За результатами виконання завдання на лабораторному занятті студенти оформлюють індивідуальні звіти, розміщують початковий код *Web*-додатка у *Git*-репозиторії, презентують і захищають звіт перед викладачем (табл. 5.1).

Таблиця 5.1

Перелік тем лабораторних занять

Назва теми	Програмні питання	Кількість годин	Література
Змістовий модуль 1 Розробка клієнтської частини *Web*-додатка			
Тема 1. Вступ до розробки *Web*-додатків	*Лабораторна робота 1.* "Проектування та верстання представлень *Web*-додатка"	4	Основна: [1 – 3]. Додаткова: [9; 14; 16]
Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS.* *Тема 4*. Відображення та форматування даних	*Лабораторна робота 2.* "Керування подання *Web*-додатків"	10	Основна: [1 – 3]. Додаткова: [9; 14; 16]
Тема 5. Взаємодія між користувачем та *Web*-додатком	*Лабораторна робота 3.* "Взаємодія між користувачем та *Web*-додатком"	8	Основна: [1 – 3]. Додаткова: [5 – 10; 14; 16]
Змістовий модуль 2 Розробка серверної частини *Web*-додатка			
Тема 6. Розробка API серверної частини *Web*-додатка	*Лабораторна робота 4.* "Розробка API серверної частини *Web*-додатка"	4	Основна: [4]. Додаткова: [11]
Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером	*Лабораторна робота 5.* "Реалізація взаємодії клієнтської та серверної частин *Web*-додатка"	4	Основна: [1 – 4]. Додаткова: [11; 12]
Тема 8. Реалізація специфічних вимог до *Web*-додатків	*Лабораторна робота 6.* "Спільна робота над проектом"	4	Основна: [1 – 3]. Додаткова: [14 – 16]
Усього годин		**34**	

Приклади завдання лабораторної роботи

1. Створити робочу директорію проекту, помістивши в неї результати виконання лабораторної роботи 1 всіх членів команди.

2. Створити репозиторій на *GitHub* для зберігання і спільної роботи над проектом.

3. Підключитися до сайта *AngularJS*.

4. Розробити меню (за аналогією з програмою прикладом), яке дозволить перемикатися між різними уявленнями.

5. Зберегти результати в репозиторій.

6. Виконати проектування моделі даних уявлення.

7. Створити для розробленого уявлення контролер і реалізувати в ньому метод ініціалізації (заповнення даними) моделі даних її фіктивною реалізацією.

8. Трансформувати уявлення в шаблон і додати його елементам директиви для: виведення значень виразів, двонаправленого зв'язку даних моделі подання з її окремими елементами, управління форматуванням елементів подання, виведення колекцій даних.

6. Самостійна робота

Самостійна робота студента (СРС) – це форма організації навчального процесу, за якої заплановані завдання виконуються студентом самостійно під методичним керівництвом викладача.

Мета СРС – засвоєння в повному обсязі навчальної програми та формування у студентів загальних і професійних компетентностей, які відіграють суттєву роль у становленні майбутнього фахівця вищого рівня кваліфікації.

Навчальний час, відведений для самостійної роботи студентів денної форми навчання, визначається навчальним планом і становить 58 % (84 години) від загального обсягу навчального часу на вивчення дисципліни (144 години). У ході самостійної роботи студент має перетворитися на активного учасника навчального процесу, навчитися свідомо ставитися до оволодіння теоретичними і практичними знаннями, вільно орієнтуватися в інформаційному просторі, нести індивідуальну відповідальність за якість власної професійної підготовки. СРС містить: опрацювання лекційного матеріалу; опрацювання та вивчення рекомендованої літератури, основних термінів та понять за темами дисципліни; підготовку

до лабораторних занять; поглиблене опрацювання окремих лекційних тем або питань; виконання індивідуальних завдань лабораторних робіт; пошук (підбір) та огляд літературних джерел за заданою проблематикою дисципліни; аналітичний розгляд наукової публікації; контрольну перевірку студентами особистих знань за запитаннями для самодіагностики; підготовку до контрольних робіт та інших форм поточного контролю; підготовку до модульного контролю; систематизацію вивченого матеріалу з метою підготовки до семестрового екзамену.

Необхідним елементом успішного засвоєння матеріалу навчальної дисципліни є самостійна робота студентів із вітчизняною та закордонною спеціальною літературою, технічною документацією до бібліотек та спеціалізованими форумами для розробників програмного забезпечення. Основні види самостійної роботи, які запропоновані студентам для засвоєння теоретичних знань з навчальної дисципліни, наведені в табл. 6.1.

Таблиця 6.1

Завдання для самостійної роботи студентів та форми її контролю

Назва теми	Зміст самостійної роботи студентів	Кількість годин	Форми контролю СРС	Література
1	2	3	4	5
Змістовий модуль 1 Розробка клієнтської частини *Web*-додатка				
Тема 1. Вступ до розробки *Web*-додатків	Вивчення лекційного матеріалу, огляд теоретичного матеріалу з теми	8	Письмова контрольна робота за темою 1	Основна: [2]
Тема 2. Формування екосистеми розробки *Web*-додатків	Пошук, підбір та огляд літературних джерел за заданою тематикою	21	Письмова контрольна робота за темою 2	Основна: [2]. Додаткова: [14 – 16]
Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS*	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи	8	Письмова контрольна робота за темами 1 – 3	Основна: [1 – 3]. Додаткова: [9; 14;16]

Закінчення табл. 6.1

1	2	3	4	5
Тема 4. Відображення та форматування даних	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи	8	Письмова контрольна робота за темами 1 – 4	Основна: [1 – 3]. Додаткова: [9; 14; 16]
Тема 5. Взаємодія між користувачем та *Web*-додатком	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи	8	Письмова контрольна робота за темами 2 – 5	Основна: [1 – 3]. Додаткова: [5 – 10; 14; 16]
Усього за змістовим модулем 1		**53**		
Змістовий модуль 2 Розробка серверної частини *Web*-додатка				
Тема 6. Розробка *API* серверної частини *Web*-додатка	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи	6	Письмова контрольна робота за темами 3 – 6	Основна: [4]. Додаткова: [11]
Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи	6	Письмова контрольна робота за темами 4 – 7	Основна: [1 – 4]. Додаткова: [11, 12]
Тема 8. Реалізація специфічних вимог до *Web*-додатків	Вивчення лекційного матеріалу, підготовка до лабораторної роботи, підготовка до контрольної роботи. Пошук, підбір та огляд літературних джерел за заданою тематикою	5	Письмова контрольна робота за темами 5 – 8	Основна: [1 – 3]. Додаткова: [14 – 16]
Усього за змістовим модулем 2		**17**		
Підготовка до екзамену та проведення підсумкового контролю		14		Основна: [1 – 4]. Додаткова: [5 – 16]
Усього		**84**		

6.1. Контрольні запитання для самодіагностики

Тема 1. Вступ до розробки *Web*-додатків

1. Дайте визначення терміна *Web*-додаток.
2. Назвіть переваги та недоліки *Web*-додатків.
3. Що таке архітектура програмного забезпечення?
4. Які складові містить дворівнева архітектура?
5. Дайте визначення термінам "клієнт" і "сервер"?
6. Які етапи розроблення програмного забезпечення вам відомі?
7. Назвіть приклади програмних *Web*-серверів.
8. Що таке *DNS*?
9. Визначите призначення браузерів.
10. У чому полягає сутність етапу "Аналіз вимог"?
11. У чому різниця між налаштуванням програмного продукту та його тестуванням?
12. Які види тестування (за об'єктом) вам відомі?
13. У чому сутність процесу впровадження програмного продукту?
14. Які принципи екстремального програмування вам відомі?

Тема 2. Формування екосистеми розробки *Web*-додатків

1. Визначте призначення *Node.js.*
2. Визначте призначення *Git Bash.*
3. Визначте призначення систем контролю версій.
4. Які етапи процесу роботи систем контролю версій вам відомі?
5. Визначте призначення систем менеджерів пакетів.
6. Які команди менеджеру npm вам відомі?
7. Дайте загальну характеристику кодуванню версій програмних бібліотек?
8. Визначте призначення пакетного менеджера *Bower*.
9. Які команди пакетного менеджера *Bower* вам відомі?
10. Визначте призначення редактору програмного коду *Sublime Text 2.*
11. Перелічіть основні плагіни *Sublime Text 2*, які потрібні для розробки клієнтської частини *Web*-додатка.

12. Визначте призначення плагіна *Sublime Text 2 Package Control.*

13. Дайте визначення процесу розгортання (*deploy*) *Web*-додатка.

14. У чому сутність конкатенації файлів *Web*-додатка?

15. У чому сутність мініфікації файлів *Web*-додатка?

16. Визначте призначення *Grunt JS*.

17. Визначте поняття "завдання" в термінології *Grunt JS*.

18. Перелічіть типові завдання, які доцільно доручити *Grunt JS*.

19. Чи можна в *Grunt JS* створити декілька режим роботи.

20. В яких файлах зберігається конфігурації, необхідні для роботи Grunt JS?

21. Що таке *Batarang*?

22. Якими засобами можна визначити стилі, які застосовуються до елемента *Web*-сторінки?

23. Як у браузері можна зробити покроковий перегляд скрипту, що виконується?

24. Визначте призначення консолі браузера.

25. Як у браузері визначити перелік файлів, що були завантажені, та швидкість цього процесу?

26. Як очистити кеш браузера?

Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS*

1. Дайте загальну характеристику фреймворка *AngularJS*.

2. Як визначити поле видимості додатка *AngularJS*.

3. В яких тегах зазвичай використовується директива *ng-app*?

4. Дайте загальну характеристику методу *module* бібліотеки *AngularJS*.

5. Що означають параметри методу *module* бібліотеки *AngularJS*.

6. Як ініціювати масив *JavaScript*?

7. Як ініціювати об'єкт *JavaScript*?

8. Як ініціювати масив об'єктів *JavaScript*?

9. Визначте основні принципи REST?

10. Дайте загальну характеристику шаблону проектування MVC.

11. Дайте визначення та характеристику поняттю *Model*.

12. Дайте визначення та характеристику поняттю *View*.

13. Дайте визначення та характеристику поняттю *Controller*.

14. Визначте призначення негайно виконуваних функцій *JavaScript*.

15. Чому доцільно використовувати негайно виконувані функції *JavaScript*?

16. Який синтаксис мають негайно виконувані функції *JavaScript*?

17. Як змінити значення властивості об'єкта *JavaScript*?

18. Як звернутися до елемента масиву *JavaScript*?

19. Яким чином доцільно зберігати подання та їх контролери?

20. Дайте визначення поняттю "програмний модуль".

21. У чому різниця між файлом та модулем?

22. Перелічіть основні угоди щодо назв файлів.

23. Які каталоги має містити *Web*-додаток?

24. Яким чином створити контролер модуля *AngularJS*?

25. Поясніть призначення параметрів методу *controller AngularJS*.

26. Яким чином задається область видимості певного контролера *AngularJS*?

27. Які властивості має функція в мові *JavaScript*?

28. Запишіть два вирази мовою *JavaScript*, перший має передавати в функцію A функцію B, другий – результат виконання функції B.

Тема 4. Відображення та форматування даних

1. Що означає вираз {{ a + b }} у поданні *Web*-додатку на *AngularJS*?

2. Чи можна використовувати такий вираз {{ a + b }} у контролерах *Web*-додатка на *AngularJS*?

3. Визначте призначення директиви *ngBind*.

4. Які вам відомі логічні операції у *JavaScript*?

5. Чому буде дорівнювати вираз (a && !a), якщо a == true?

6. Чому буде дорівнювати вираз (a || !a), якщо a == false?

7. Чому буде дорівнювати вираз (a >= 5), якщо a == 5?

8. Чому буде дорівнювати вираз (a == 0), якщо a == false?

9. Визначте призначення директив *ngShow* та *ngHide*.

10. За яких умов доцільного використовувати директиви *ngShow* та *ngHide*.

11. Наведіть приклад завдань, коли доцільного використовувати директиви *ngShow* та *ngHide*.

12. Визначте призначення директив *ngSwitch* та *ngIf*.

13. У чому відмінності директив *ngSwitch* та *ngIf*?

14. У чому сутність директиви *ngInclude*?

15. Визначте призначення директиви *ngRepeat*.

16. Синтаксис *ngRepeat*.

17. Що відображає зміна *$index* всередині *ngRepeat*?

18. Як визначити останній елемент колекції, що виводиться за допомогою *ngRepeat*?

19. Визначите призначення директиви *ngClass*?

20. Що означає вираз Text?

21. У яких випадках доцільно використовувати директиви *ngRepeat Start* та *ngRepeatEnd*?

22. Визначте поняття "фільтр".

23. Визначте призначення фільтрів.

24. Дайте загальну характеристику фільтрів.

25. Що означає вираз {{myLongString | limitTo:80 | lowercase}}?

26. Яке призначення фільтра *filter*?

27. Яке призначення фільтра *orderBy*?

28. Запишіть код-приклад, у якому доступ до фільтра буде відбуватися у методах контролера.

Тема 5. Взаємодія між користувачем та *Web*-додатком

1. Дайте визначення терміна "подія".

2. Які вам відомі вбудовані директиви для оброблення подій?

3. Запишіть код подання, який реалізує таке: після натиснення кнопки має бути виконаний метод *save*().

4. У чому сутність двонаправленого зв'язку?

5. Визначте призначення директиви *ngModel*.

6. У чому відмінність директив *ngBind* та *ngModel*?

7. Наведіть код-приклад використання директиви *ngInput*.

8. Яким чином можна задати максимальну довжину рядка текстового поля?

9. Яким чином можна задати перевірку рядка текстового поля за допомогою регулярного виразу?

10. Наведіть код-приклад щодо реалізації індикативних перемикачів (*CheckBox*).

11. Наведіть код-приклад щодо реалізації альтернативних перемикачів (*RadioButton*).

12. Наведіть код-приклад динамічного формування низпадального списку (*Select*).

13. Визначте призначення вбудованих класів *ngPristine* та *ngDirty*.

14. Визначте призначення вбудованих класів *ngValid* та *ngInvalid*.

Тема 6. Розробка API серверної частини *Web*-додатка

1. Дайте визначення терміна *Web API*.

2. Чим відрізняється *Web API* від звичайної MVC серверної частини?

3. Опішить процес створення проекту серверної частини.

4. Визначте призначення контролера *Web API*.

5. Яким угодам повинна відповідати назва контролера *Web API*.

6. У чому сутність процесу маршрутизації *Web API*?

7. Як називається та де зберігається файл із кодом щодо налаштування маршрутів?

8. Наведіть код-приклад для налаштування маршруту для багатомовного сайта.

9. Визначте призначення методів контролера *Web API*.

10. У чому різниця між методами *GET* та *POST*?

11. У чому різниця між методами *POST* та *PUT*?

12. Якими інструментальними засобами можна перевірити працездатність *Web API*?

Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером

1. Розкрийте сутність асинхронних запитів.

2. Які переваги надає використання асинхронних запитів?

3. У яких форматах може здійснюватися передача даних у асинхронних запитах?

4. Що таке *JSON*?

5. Запишіть інформацію про студентів у форматі *JSON*.

6. Які види асинхронних *HTTP*-запитів вам відомі?

7. Назвіть правила роботи з протоколом *HTTP*.

8. Які вам відомі коди відповідей на *HTTP*-запити?

9. Які методи *AngularJS* дозволять перетворити *JavaScript* об'єкт у формат JSON та навпаки дані у форматі *JSON* у *JavaScript* об'єкт?

10. Визначите призначення служби *$http*.

11. Які вам відомі методи служби *$http*?

12. Наведіть приклади коду налаштування *HTTP*-запитів.

13. Що таке *Promise*?

14. Визначте призначення служби *$q*.

15. Як здійснити об'єднання асинхронних запитів у ланцюжки? Наведіть приклад коду.

16. Як здійснити об'єднання відкладених результатів? Наведіть приклад коду.

Тема 8. Реалізація специфічних вимог до *Web*-додатків

1. Визначите призначення служби *$route*?

2. Що таке *SPA*?

3. Яким чином доцільно організувати маршрутизацію на клієнтській частини *Web*-додатка?

4. Яким чином можна запобігати зміні маршруту засобами *AngularJS*?

5. Визначте призначення директиви *ngView*.

6. Які стороні бібліотеки вам відомі?

7. Яким чином здійснюється підключення сторонніх бібліотек до *AngularJS додатку?*

8. Які ви знаєте стратегії перекладів користувальницького інтерфейсу?

7. Індивідуально-консультативна робота

Індивідуально-консультативну роботу здійснюють за графіком індивідуально-консультативної роботи у формі індивідуальних занять, консультацій, перевірки виконання індивідуальних завдань, перевірки та захисту завдань, що винесені на поточний контроль, тощо.

Формами організації індивідуально-консультативної роботи є:

а) за засвоєнням теоретичного матеріалу:

консультації: індивідуальні (запитання – відповідь), групові (розгляд типових прикладів – ситуацій);

б) за засвоєнням практичного матеріалу:

консультації індивідуальні та групові;

в) для комплексного оцінювання засвоєння програмного матеріалу:

індивідуальне здавання виконаних робіт.

8. Методи навчання

У процесі викладання навчальної дисципліни для активізації навчально-пізнавальної діяльності студентів передбачене застосування як активних, так і інтерактивних навчальних технологій, серед яких: лекції проблемного характеру, міні-лекції, робота в малих групах, мозкові атаки, кейс-метод, презентації, ознайомлювальні (початкові) ігри, метод проектної роботи, метод сценаріїв, банки візуального супроводу (табл. 8.1 і 8.2).

Таблиця 8.1

Розподіл форм та методів активізації процесу навчання за темами навчальної дисципліни

Тема	Практичне застосування навчальних технологій
Тема 1. Вступ до розробки *Web*-додатків	Лекція проблемного характеру, робота в малих групах, банки візуального супроводу
Тема 2. Формування екосистеми розробки *Web*-додатків	Міні-лекції, банки візуального супроводу, метод сценаріїв
Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS*	Лекція проблемного характеру, мозкові атаки, робота в малих групах, презентація, банки візуального супроводу
Тема 4. Відображення та форматування даних	Лекція проблемного характеру, робота в малих групах, презентація, банки візуального супроводу
Тема 5. Взаємодія між користувачем та *Web*-додатком	Лекція проблемного характеру, робота в малих групах, презентація, банки візуального супроводу, метод сценаріїв
Тема 6. Розробка *API* серверної частини *Web*-додатка	Лекція проблемного характеру, робота в малих групах, презентація, банки візуального супроводу
Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером	Лекція проблемного характеру, робота в малих групах, презентація, банки візуального супроводу
Тема 8. Реалізація специфічних вимог до *Web*-додатків	Лекція проблемного характеру, робота в малих групах, презентація, банки візуального супроводу

Основні відмінності активних та інтерактивних методів навчання від традиційних визначаються не тільки методикою і технікою викладання, але й високою ефективністю навчального процесу, який виявляється у: високій мотивації студентів; закріпленні теоретичних знань на практиці; підвищенні самосвідомості студентів; формуванні здатності приймати

самостійні рішення; формуванні здатності до ухвалення колективних рішень; формуванні здатності до соціальної інтеграції; набуття навичок вирішення конфліктів; розвитку здатності до знаходження компромісів.

Лекції проблемного характеру – один із найважливіших елементів проблемного навчання студентів. Вони передбачають поряд із розглядом основного лекційного матеріалу встановлення та розгляд кола проблемних питань дискусійного характеру, які недостатньо розроблені в науці й мають актуальне значення для теорії та практики. Лекції проблемного характеру відрізняються поглибленою аргументацією матеріалу, що викладається. Вони сприяють формуванню у студентів самостійного творчого мислення, прищеплюють їм пізнавальні навички. Студенти стають учасниками наукового пошуку та вирішення проблемних ситуацій.

Міні-лекції передбачають викладення навчального матеріалу за короткий проміжок часу й характеризуються значною ємністю, складністю логічних побудов, образів, доказів та узагальнень. Вони проводяться, як правило, як частина заняття-дослідження. Міні-лекції відрізняються від повноформатних лекцій значно меншою тривалістю. Зазвичай міні-лекції тривають не більше 10 – 15 хвилин і використовуються для того, щоб стисло донести нову інформацію до всіх слухачів. Міні-лекції часто застосовуються як частини цілісної теми, яку бажано викладати повноформатною лекцією, щоб не втомлювати аудиторію. Тоді інформація надається по черзі кількома окремими сегментами, між якими застосовуються інші форми й методи навчання.

Робота в малих групах дає змогу структурувати лабораторні заняття за формою і змістом, створює можливості для участі кожного студента в роботі за темою заняття, забезпечує формування особистісних якостей та досвіду соціального спілкування.

Мозкові атаки – метод вирішення невідкладних завдань, сутність якого полягає в тому, щоб висловити якомога більшу кількість ідей за дуже обмежений проміжок часу, обговорити і здійснити їх селекцію.

Презентації – виступи перед аудиторією, що використовуються для представлення певних досягнень, результатів роботи групи звіту про виконання індивідуальних завдань, проектних робот. Презентації можуть бути як індивідуальними, наприклад виступ одного слухача, так і колективними, тобто виступи двох та більше слухачів.

Метод сценаріїв полягає в розробленні ймовірних моделей поведінки та розвитку конкретних явищ у перспективі.

Банки візуального супроводу сприяють активізації процесу навчання за темами навчальної дисципліни за допомогою наочності.

Таблиця 8.2

Використання методик активізації процесу навчання

Тема навчальної дисципліни	Практичне застосування методик	Методики активізації процесу навчання
Тема 1. Вступ до розробки *Web*-додатків	*Лабораторна робота 1.* "Проектування та верстання представлень *Web*-додатка"	Метод сценаріїв, презентація, робота в малих групах, мозкові атаки
Тема 3. Розробка клієнтської частини *Web*-додатка засобами *AngularJS.* *Тема 4.* Відображення та форматування даних	*Лабораторна робота 2*. "Керування подання *Web*-додатків"	Презентація, робота в малих групах, мозкові атаки
Тема 5. Взаємодія між користувачем та *Web*-додатком	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	Презентація, робота в малих групах, мозкові атаки
Тема 6. Розробка *API* серверної частини *Web*-додатка	*Лабораторна робота 4*. "Розробка *API* серверної частини *Web*-додатка"	Презентація, робота в малих групах, мозкові атаки
Тема 7. Взаємодія клієнтської частини *Web*-додатка із сервером	*Лабораторна робота 5.* "Реалізація взаємодії клієнтської та серверної частин *Web*-додатка"	Презентація, робота в малих групах
Тема 8. Реалізація специфічних вимог до *Web*-додатків	*Лабораторна робота 6.* "Спільна робота над проектом"	Презентація, робота в малих групах

9. Методи контролю

Система оцінювання сформованих компетентностей (див. табл. 2.1) у студентів ураховує види занять, які згідно з програмою навчальної дисципліни передбачають лекційні, лабораторні заняття, а також виконання самостійної роботи. Оцінювання сформованих компетентностей у студентів здійснюється за накопичувальною 100-бальною системою. Відповідно до Тимчасового положення "Про порядок оцінювання результатів навчання студентів за накопичувальною бально-рейтинговою системою" ХНЕУ ім. С. Кузнеця, контрольні заходи містять:

поточний контроль, що здійснюється протягом семестру під час проведення лекційних, лабораторних занять і оцінюється сумою набраних балів (максимальна сума – 60 балів; мінімальна сума, що дозволяє студенту складати іспит, – 35 балів);

підсумковий/семестровий контроль, що проводиться у формі семестрового екзамену, відповідно до графіка навчального процесу.

Поточний контроль із цієї навчальної дисципліни проводять в таких формах:

активна участь у дискусії та презентації матеріалу на лабораторних заняттях;

захист лабораторних робіт;

проведення письмової контрольної роботи;

проведення диктанту за лекційним матеріалом.

Підсумковий/семестровий контроль проводиться у формі семестрового екзамену. **Семестрові екзамени** – форма оцінювання підсумкового засвоєння студентами теоретичного та практичного матеріалу з окремої навчальної дисципліни, що проводиться як контрольний захід.

Порядок проведення поточного оцінювання знань студентів. Оцінювання знань студента під час лабораторних занять та виконання індивідуальних завдань проводиться за накопичувальною системою за такими критеріями:

розуміння, ступінь засвоєння теорії та методології проблем, що розглядаються;

ступінь засвоєння фактичного матеріалу навчальної дисципліни;

ознайомлення з рекомендованою літературою, а також із сучасною літературою з питань, що розглядаються;

вміння поєднувати теорію з практикою під час розв'язання задач;

логіка, структура, стиль викладу матеріалу в письмових роботах і під час виступів в аудиторії, вміння обґрунтовувати свою позицію, здійснювати узагальнення інформації та робити висновки.

Максимально можливий бал за конкретним завданням ставиться за умови відповідності індивідуального завдання студента або його усної відповіді всім зазначеним критеріям. Відсутність тієї або іншої складової знижує кількість балів. У процесі оцінювання індивідуальних завдань увага також приділяється якості, самостійності та своєчасності здачі виконаних завдань викладачу, згідно з графіком навчального процесу. Якщо якась із вимог не буде виконана, то бали будуть знижені.

Поточний контроль проводять на кожній лекції у вигляді письмової контрольної роботи тривалістю до 5 хвилин та містить 2 – 3 запитання з попереднього навчального матеріалу.

Критерії оцінювання позааудиторної самостійної роботи студентів. Загальними критеріями, за якими здійснюється оцінювання позааудиторної самостійної роботи студентів, є: глибина і міцність знань, рівень мислення, вміння систематизувати знання за окремими темами, вміння робити обґрунтовані висновки, володіння категорійним апаратом, навички і прийоми виконання практичних завдань, вміння знаходити необхідну інформацію, здійснювати її систематизацію та оброблення, самореалізація на лабораторних заняттях.

Порядок підсумкового контролю з навчальної дисципліни. Підсумковий контроль знань та компетентностей студентів із навчальної дисципліни здійснюється на підставі проведення семестрового екзамену. Екзаменаційний білет охоплює програму дисципліни і передбачає визначення рівня знань та ступеня опанування студентами компетентностей (див. табл. 2.1).

Завданням екзамену є перевірка розуміння студентом програмного матеріалу в цілому, логіки та взаємозв'язків між окремими розділами, здатності творчого використання накопичених знань, вміння формулювати своє ставлення до певної проблеми навчальної дисципліни тощо. В умовах реалізації компетентнісного підходу екзамен оцінює рівень засвоєння студентом компетентностей, що передбачені кваліфікаційними вимогами. Кожен екзаменаційний білет складається із 6 завдань, які передбачають вирішення типових професійних завдань фахівця на робочому місці та дозволяють діагностувати рівень теоретичної підготовки студента і рівень його компетентності з навчальної дисципліни.

Екзаменаційний білет містить п'ять діагностичних та одне евристичне завдання, які оцінюються відповідно до Тимчасового положення "Про порядок оцінювання результатів навчання студентів за накопичувальною бально-рейтинговою системою" ХНЕУ ім. С. Кузнеця.

Студент, який із поважних причин, підтверджених документально, не мав можливості брати участь у формах поточного контролю, тобто не склав змістовий модуль, має право на його відпрацювання у двотижневий термін після повернення до навчання за розпорядженням декана факультету відповідно до встановленого терміну.

Студент не може бути допущений до складання екзамену, якщо кількість балів, одержаних за результатами перевірки успішності під час поточного та модульного контролю відповідно до змістового модуля впродовж семестру, в сумі не досягла 35 балів. Після екзаменаційної сесії декан факультету видає розпорядження про ліквідацію академічної заборгованості. У встановлений термін студент добирає залікові бали.

Студента слід вважати атестованим, якщо сума балів, одержаних за результатами підсумкової/семестрової перевірки успішності, дорівнює або перевищує 60. Мінімально можлива кількість балів за поточний і модульний контроль упродовж семестру – 35 та мінімально можлива кількість балів, набраних на екзамені, – 25.

Результат семестрового екзамену оцінюється в балах (максимальна кількість – 40 балів, мінімальна кількість, що зараховується, – 25 балів) і проставляється у відповідній графі екзаменаційної "Відомості обліку успішності".

Підсумкову оцінку з навчальної дисципліни розраховують з урахуванням балів, отриманих під час екзамену, та балів, отриманих під час поточного контролю за накопичувальною системою. Сумарний результат у балах за семестр складає: "60 і більше балів – зараховано", "59 і менше балів – не зараховано" та заноситься у залікову "Відомість обліку успішності" навчальної дисципліни. У випадку отримання менше 60 балів студент обов'язково здає залік після закінчення екзаменаційної сесії у встановлений деканом факультету термін, але не пізніше двох тижнів після початку семестру. У випадку повторного отримання менше 60 балів декан факультету призначає комісію у складі трьох викладачів на чолі із завідувачем кафедри та визначає термін перескладання заліку, після чого приймається рішення відповідно до чинного законодавства: "зараховано" – студент продовжує навчання за графіком навчального процесу, а якщо "не зараховано", тоді декан факультету пропонує студенту повторне вивчення навчальної дисципліни протягом наступного навчального періоду самостійно.

Обмеження на використання джерел інформації під час екзамену відсутні. Наявність доступу до мережі Інтернет обов'язкова.

Харківський національний економічний університет імені Семена Кузнеця

Освітній ступінь	Бакалавр
Напрям підготовки	Видавничо-поліграфічна справа
Семестр	7
Навчальна дисципліна	Розробка *Web*-додатків

Екзаменаційний білет 3 – 25

Завдання 1 (діагностичне). Зверстати *Web*-сторінку "Склад футбольної команди" засобами HTML/CSS. Сторінка повинна мати головне меню додатка, заголовок сторінки, таблицю з такими колонками: "Прізвище гравця", "Номер", "Амплуа", "Дата, коли прийшов до клубу", "Дата, коли покинув клуб".

Завдання 2 (діагностичне). Реалізувати подання *Web*-сторінки, яке має відображати дані в таблиці.

Завдання 3 (діагностичне). Реалізувати програмний код контролера, який має реалізує методи для: завантаження даних із серверної частини та збереження даних на сервері.

Завдання 4 (діагностичне). Створити серверну частину *Web*-додатка та створити відповідну модель даних (клас). Ініціювати колекцію *mock-data* для відправлення на клієнтську частину *Web*-додатка.

Завдання 5 (діагностичне). Реалізувати *API* контролер із методами для відправки колекції *mock-data* на клієнтську частину *Web*-додатка та для опрацювання запиту на збереження даних.

Затверджено на засіданні кафедри комп'ютерних систем і технологій
Протокол № ___ від "___" _____ 20__ року

Завідувач кафедри _____ Екзаменатор _____

Підсумкові бали за екзамен складаються із суми балів за виконання всіх завдань, що округлені до цілого числа за правилами математики.

Алгоритм вирішення кожного завдання містить окремі етапи, які відрізняються за складністю, трудомісткістю та значенням для розв'язання завдання. Тому окремі завдання та етапи їх розв'язання оцінюються відокремлено один від одного таким чином:

Завдання 1 (діагностичне). Максимум 8 балів.

Контрольовані параметри:

1. Технологія, яка використовувалося для підключення сторонніх бібліотек:

а) інсталяція засобами менеджерів пакетів (*Bower*, *NuGet*) – 1 бал;

б) завантаження з сайта – 0,5 бала;

в) посилання на *Web*-репозиторій – 0,3 бала.

2. Використання сучасних CSS фреймворків – 1 бал.

3. Наявність власних стилів необхідних та достатніх для виконання вимог завдання та підключених до сторінки – 2 бали.

4. Верстка *HTML*-сторінки відповідає вимогам завдання (містить усі необхідні елементи) – 2 бали.

5. Якість форматування коду *HTML*-сторінки – 2 бали.

Завдання 2 (діагностичне). Максимум 7 балів.

Контрольовані параметри:

1. Подання сторінки дозволяє вирішити поставлені в роботі завдання – 1 бал.

2. Наявність лаконічного та зрозумілого текстового обґрунтування використання директив *Angular JS* – 2 бали.

3. Використання фільтрів для форматування даних – 1 бал.

4. Відповідність до угоди до формування представлення – 1 бал.

5. Реалізація *Web*-додатка як:

а) SPA (Single Page Application) з налаштування маршрутів засобами *$routeProvider* та *ngView* – 2 бали;

б) інший спосіб – 0,5 бала.

Завдання 3 (діагностичне). Максимум 8 балів.

Контрольовані параметри:

1. Методи контролеру не виконують в повному обсязі своє призначення – 3 бали.

2. Використання сторонніх бібліотек для спрощення опрацювання даних – 2 бали.

3. Кожна помилок, виявлена *Sublime Linter*, штрафується 0,1 бала.

4. Відповідність до угоди оформлення коду контролерів – 2,5 бала:

а) зберігання контролера у вигляді окремого файла-модуля – 0,5 бала;

б) оформлення контролеру у вигляді функції, що негайно виконується – 0,5 бала;

в) використання патерну vm – 0,3 бала;

г) використання замикань для методів контролеру – 0,2 бала;

е) присвоєння функцій методам контролерів – 0,5 бала;

є) дотримання правил найменування властивостей та методів контролера – 0,5 бала.

5. Якість форматування коду – 0,5 бала.

Завдання 4 (діагностичне). Максимум 6 балів.

Контрольовані параметри:

1. Лаконічність та повнота моделі даних *Web*-сторінки – 1 бал.

2. Використання модуля *$scope* для роботи з моделлю даних – 1 бал.

3. Наявність методу, що імітує повернення даних із серверу – 1 бал.

4. Якість методів для відправлення запитів та отримання відповідей на/з сервера – 2 бали.

5. Відповідність угоді про назву – 0,5 бала.

6. Якість форматування коду – 0,5 бала.

Завдання 5 (діагностичне). Максимум 5 балів.

Контрольовані параметри:

1. Працездатність серверної частини – 1,5 бала.

2. Наявність API контролера та відповідних методів для опрацювання запитів клієнтської частини – 1 бал.

3. Наявність та якість моделі у вигляді класів на серверній частині – 1 бал.

4. Наявність тестових даних, які має повернути API контролер на клієнтську частину – 1 бал.

5. Якість форматування коду – 0,5 бала.

Загальні бали. Максимум 6 балів.

Контрольований параметр, за який нараховується загальні бали, є працездатність проекту в цілому – 6 балів.

10. Розподіл балів, які отримують студенти

Система оцінювання рівня сформованості професійних компетентностей студентів денної форми навчання наведена в табл. 10.1.

Таблиця 10.1

Система оцінювання рівня сформованості професійних компетентностей

Професійні компетентності	Навчальний тиждень	Години		Форми навчання		Оцінка рівня сформованості компетентностей	
						Форми контролю	Макс. бал
1	2	3		4		5	6
Змістовий модуль 1. Розробка клієнтської частини *Web*-додатка							
Здатність розуміти процес розробки *Web*-додатка	1	Ауд.	2	Лекція	*Тема 1.* Вступ до розробки *Web*-додатків	–	–
		Ауд.	2	Лабораторна робота (ЛР)	*Лабораторна робота 1.* "Проектування та верстання представлень *Web*-додатка"	–	–
		СРС	2	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
Здатність реалізовувати клієнтську частину *Web*-додатка	2	Ауд.	2	Лекція	*Тема 3.* Розробка клієнтської частини *Web*-додатка засобами *AngularJS*	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 1.* "Проектування та верстання представлень *Web*-додатка"	Перевірка ЛР 1	5
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	3	Ауд.	2	Лекція	*Тема 3.* Розробка клієнтської частини *Web*-додатка засобами *AngularJS*	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 2.* "Керування подання *Web*-додатків"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	4	Ауд.	2	Лекція	*Тема 4.* Відображення та форматування даних	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 2.* "Керування подання *Web*-додатків"	–	–

Продовження табл. 10.1

1	2	3		4		5	6
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	5	Ауд.	2	Лекція	*Тема 4.* Відображення та форматування даних	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 2*. "Керування подання *Web*-додатків"	Перевірка ЛР 2	7
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	6	Ауд.	2	Лекція	*Тема 4.* Відображення та форматування даних	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 2*. "Керування подання *Web*-додатків"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	7	Ауд.	2	Лекція	*Тема 5.* Взаємодія між користувачем та *Web*-додатком	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	8	Ауд.	2	Лекція	*Тема 5.* Взаємодія між користувачем та *Web*-додатком	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	Перевірка ЛР 3	8
		СРС	6	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–

Продовження табл. 10.1

1	2	3		4		5	6
	9	Ауд.	2	Лекція	*Тема 5.* Взаємодія між користувачем та *Web*-додатком	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
Змістовий модуль 2. Розробка серверної частини *Web*-додатка							
Здатність реалізовувати *API* серверної частини *Web*-додатка	10	Ауд.	2	Лекція	*Тема 6.* Розробка *API* серверної частини *Web*-додатка	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	11	Ауд.	2	Лекція	*Тема 6.* Розробка *API* серверної частини *Web*-додатка	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 3*. "Взаємодія між користувачем та *Web*-додатка"	Перевірка ЛР 4	8
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
Здатність реалізовувати взаємодію клієнтської та серверної частин *Web*-додатка	12	Ауд.	2	Лекція	*Тема 7.* Взаємодія клієнтської частини *Web*-додатка із сервером	КР за матеріалами попередніх та/або поточної лекції	1
		Ауд.	2	ЛР	*Лабораторна робота 4*. "Розробка *API* серверної части *Web*-додатка"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	13	Ауд.	2	Лекція	*Тема 7.* Взаємодія клієнтської частини *Web*-додатка із сервером	КР за матеріалами попередніх та/або поточної лекції	1

Закінчення табл. 10.1

1	2	3		4		5	6
		Ауд.	2	ЛР	*Лабораторна робота 4*. "Розробка *API* серверної части *Web*-додатка"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	14	Ауд.		Лекція	–	–	–
		Ауд.	2	ЛР	*Лабораторна робота 5.* "Реалізація взаємодії клієнтської та серверної частин *Web*-додатків"	Перевірка ЛР 5	10
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
Здатність спільно працювати над проектом та здійснювати інтеграцію його частин в єдиний додаток	15	Ауд.		Лекція	–	–	–
		Ауд.	2	ЛР	*Лабораторна робота 5.* "Реалізація взаємодії клієнтської та серверної частин *Web*-додатків"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	16	Ауд.		Лекція	–	–	–
		Ауд.	2	ЛР	*Лабораторна робота 6.* "Спільна робота над проектом"	–	–
		СРС	4	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	17	Ауд.		Лекція	–	–	–
		Ауд.	2	ЛР	*Лабораторна робота 6.* "Спільна робота над проектом"	Перевірка ЛР 6	10
		СРС	6	Підготовка до занять	Пошук, підбір та огляд літературних джерел за тематикою ЛР	–	–
	Сесія	Ауд.	2	Передекзаменаційна консультація	Вирішення практичних завдань на різні теми, що входять до підсумкового контролю	–	–
		Ауд.	3	Екзамен	Виконання завдань екзаменаційного білета	Підсумковий контроль	40
		СРС	9	Підготовка до екзамену	Повторення матеріалів змістовних модулів	–	–
Усього годин		**144**		**Загальна максимальна кількість балів із дисципліни**			**100**
із них:							
аудиторні		65		*поточний контроль*			60
самостійна робота		79		*підсумковий контроль*			40

Розподіл балів у межах тем змістових модулів наведено в табл. 10.2.

Таблиця 10.2

Розподіл балів за темами

Поточне тестування та самостійна робота								Підсумковий тест (екзамен)	Сума
Змістовий модуль 1					**Змістовий модуль 2**			40	100
Т1	Т2	Т3	Т4	Т5	Т6	Т7	Т8		
2	6	6	10	10	10	10	6		

Примітка. Т1, Т2 ... Т8 – теми змістових модулів.

Максимальну кількість балів, яку може накопичити студент протягом тижня за формами та методами навчання, наведено в табл. 10.3.

Таблиця 10.3

Розподіл балів за тижнями

Теми змістового модуля			**Лабораторні заняття**	**Письмова КР**	**Усього**
Змістовий модуль 1	Тема 1	1 тиждень	–	–	0
	Тема 3	2 тиждень	–	1	1
	Тема 3	3 тиждень	5	1	6
	Тема 4	4 тиждень	–	1	1
	Тема 4	5 тиждень	–	1	1
	Тема 5	6 тиждень	7	1	8
	Тема 5	7 тиждень	–	1	1
	Тема 5	8 тиждень	8	1	9
Змістовий модуль 2	Тема 6	9 тиждень	–	1	1
	Тема 6	10 тиждень	–	1	1
	Тема 6	11 тиждень	8	1	9
	Тема 7	12 тиждень	–	1	1
	Тема 7	13 тиждень	–	1	1
	Тема 2	14 тиждень	10	–	10
	Тема 2	15 тиждень	–	–	0
	Тема 8	16 тиждень	–	–	0
	Тема 8	17 тиждень	10	–	10
Усього			**48**	**12**	**60**

Підсумкову оцінку з навчальної дисципліни визначають відповідно до Тимчасового положення "Про порядок оцінювання результатів навчання студентів за накопичувальною бально-рейтинговою системою" ХНЕУ ім. С. Кузнеця (табл. 10.4).

Таблиця 10.4

Шкала оцінювання: національна та ЄКТС

Сума балів за всі види навчальної діяльності	Оцінка ЄКТС	Оцінка за національною шкалою	
		для екзамену, курсового проекту (роботи), практики	для заліку
90 – 100	A	відмінно	зараховано
82 – 89	B	добре	
74 – 81	C		
64 – 73	D	задовільно	
60 – 63	E		
35 – 59	FX	незадовільно	не зараховано
1 – 34	F		

Оцінки за цією шкалою заносять до відомостей обліку успішності, індивідуального навчального плану студента та іншої академічної документації.

11. Рекомендована література

11.1. Основна

1. Козловский П. Разработка веб-приложений с использованием AngularJS / П. Козловский П. Дарвин ; пер. с англ. А. Н. Киселева. – М. : ДМК Пресс, 2014. – 394 с. : ил.

2. Пьюривал С. Основы разработки веб-приложений / С. П. Пьюривал. – СПб. : Питер, 2015. – 272 с.

3. Фримен А. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов / А. Фримен. – СПб. : Вильямс, 2013. – 688 с.

4. Freeman A. Pro AngularJS / A. Freeman. – NY. : APRESS, 2014. – 670 с.

11.2. Додаткова

5. Маккоу А. Веб-приложение на JavaScript / А. Маккоу. – СПб. : Питер, 2012. – 288 с.

6. Моррисон М. Изучаем JavaScript / М. Моррисон. – СПб. : Питер, 2012. – 608 с.

7. Стефанов С. JavaScript. Шаблоны / С. Стефанов ; пер. с англ. – СПб. : Символ-Плюс, 2011. – 272 с.

8. Флэнаган Д. JavaScript. Подробное руководство / Д. Флэнаган; пер. с англ. – СПб: Символ Плюс, 2008. – 992 с. : ил.

11.3. Інформаційні ресурси

9. 15 тривиальных фактов о правильной работе с протоколом HTTP [Электронный ресурс]. – Режим доступа : http://habrahabr.ru/company/yandex /blog/265569.

10. Пакетный менеджер Bower [Электронный ресурс] – Режим доступа : http://stepansuvorov.com/blog/2013/03/пакетный-менеджер-bower.

11. Разработка Web API [Электронный ресурс]. – Режим доступа : http://habrahabr.ru/post/181988.

12. Современный учебник JavaScript [Электронный ресурс]. – Режим доступу : http://learn.javascript.ru.

13. AngularJS – Superheroic JavaScript MVW Framework [Electronic resource] – Access mode : https://angularjs.org.

14. Git. Book [Electronic resource]. – Access mode : https://git-scm. com/ book/ru.

15. Grunt. The JavaScript Task Runner [Electronic resource]. – Access mode http://gruntjs.com.

16. Learn About ASP.NET Web API [Electronic resource]. – Access mode : http://www.asp.net/Web-api.

11.4. Методичне забезпечення

17. Гіковатий В. М. Розробка Web-додатків [Електронний ресурс] / В. М. Гіковатий. – Режим доступу : http://www.ikt.hneu.edu.ua/course/view. php?id=2578.

Додатки

Додаток А
Таблиця А.1

Структура складових професійних компетентностей з навчальної дисципліни "Розробка *Web*-додатків" за Національною рамкою кваліфікацій України

Складові компетентності, яка формується в рамках теми	Мінімальний досвід	Знання	Вміння	Комунікації	Автономність і відповідальність
1	2	3	4	5	6
Тема 1. **Вступ до розробки *Web*-додатків**					
Здатність розуміти процес розробки *Web*-додатка	Мова розмітки *HTML /CSS*, загальні принципи роботи мережі Інтернет	Знання категорійного апарату процесу розробки *Web*-додатків; основних етапів розробки *Web*-додатків та їх сутність	Опрацьовувати вимоги замовників програмного забезпечення; здійснювати прототипування користувальницького інтерфейсу майбутнього *Web*-додатка	Обговорювати вимоги із замовником	Самостійно проводити аналіз інструментальних засобів прототипування користувальницького інтерфейсу
Тема 2. **Формування екосистеми розробки *Web*-додатків**					
Здатність формувати екосистему розробки *Web*-додатків	Основи роботи з командним рядком. Мова *Java Script*: робота з об'єктами	Знання інструментальних засобів розробки *Web*-додатків; технології командної роботи	Налаштовувати екосистему процесу *Web*-розробки; користуватися менеджерами пакетів та за їх допомогою інсталювати потрібні сторонні бібліотеки; користуватися системами контролю версіями	Ґрунтовно обговорювати питання вибору інструментальних засобів екосистеми та правил роботи в команді	Приймати рішення щодо доцільності використання певних інструментальних засобів екосистеми розробки *Web*-додатків

1	2	3	4	5	6
Тема 3. **Розробка клієнтської частини *Web*-додатка засобами *AngularJS***					
Здатність реалізовувати клієнтську частину *Web*-додатка	Принципи роботи *Web*-додатків на основі різних підходів. Мова розмітки *HTML/CSS*	Знання щодо поняття *"Framework"*; патерн *MVC*; загальної характеристика *Framework AngularJS*	Підключати та ініціювати *AngularJS*. Реалізовувати модульний підхід на рівні представлення. Ініціювати та виводити у подані *mock-data* для цілей тестування користувальницького інтерфейсу. Реалізовувати у представленні елементарну поведінку	Презентувати прототип *Web*-додатка та уточнювати вимоги із замовником	Самостійно здійснювати аналіз інструментальних засобів щодо розробки *Web*-додатків
Тема 4. **Відображення та форматування даних**					
Здатність реалізовувати клієнтську частину *Web*-додатка	Мова розмітки *HTML/CSS* поняття "елемент дерева *DOM*", "атрибут". Мова *Java Script*: основні операції з елементами дерева *DOM*; об'єкти, масиви	Знання директив *AngularJS*, які забезпечують виведення *Web*-сторінки або її фрагмента за умовою; виведення елементів колекції; поняття *Culture*; засоби форматування даних	Реалізовувати відображення сторінки та /або її фрагменту за певною умовою. Відображати колекції даних. Форматувати дані під час їх рендерінгу на сторінці	Презентувати прототип *Web*-додатка та уточнювати вимоги із замовником	Самостійно вирішувати завдання щодо подання *Web*-додатка
Тема 5. **Взаємодія між користувачем та *Web*-додатком**					
Здатність реалізовувати клієнтську частину *Web*-додатка	Основи алгоритмізації. Синтаксис, типи даних, основні оператори мови *Java Script.* Особливості використання функцій у мові *JavaScript*	Знання подій, які виникають підчас взаємодії користувача з елементами *Web*-сторінки. Знання директив *AngularJS*, які використовуються для опрацювання події; реалізації елементів форм;	Реалізовувати: форми для введення даних; двоспрямований зв'язок даних моделі та елементів форми; обробники подій	Презентувати прототип *Web*-додатка та уточнювати вимоги із замовником	Самостійно вирішувати завдання щодо реалізації методів опрацювання даних чи подій користувача

1	2	3	4	5	6
		двонаправленого зв'язку даних моделі на елементів форм			
Тема 6. **Розробка API серверної частини *Web*-додатка**					
Здатність реалізовувати *API* серверної частини *Web*-додатка	Знання принципів об'єктно-орієнтованого програмування. Вміння працювати у сучасних *IDE*. Знання мови програмування C#	Знання понять: "*Web API*", "*REST*", "кросдоменний запит", "маршрут", "*API* контроллер"	Реалізовувати контролери *API* та їх основні методи. Налаштовувати маршрути та доступ до *Web*-ресурсу	Ґрунтовно обговорювати особливості *Web API* та/або формати передачі даних із *back-end* розробниками	Самостійно вирішувати завдання щодо реалізації *API Web*-додатка
Тема 7. **Взаємодія клієнтської частини *Web*-додатка із сервером**					
Здатність реалізовувати взаємодію клієнтської та серверної частин *Web*-додатка	Загальні принципи роботи протоколу http мережі Інтернет. Види запитів. Мова *Java Script*: поняття Promise	Знання сутності асинхронних запитів. Знання форматів передачі даних, які використовуються для асинхронних запитів. Призначення та методи служби *$http AngularJS*. Особливості опрацювання даних під час асинхронних запитів	Реалізовувати асинхронні запити засобами служби *$http AngularJS*	Ґрунтовно обговорювати особливості *Web API* та/або формати передачі даних із *back-end* розробниками. Презентувати прототип *Web*-додатка та уточнювати вимоги з замовником	Самостійно вирішувати завдання щодо забезпечення взаємодії клієнтської та серверної частин *Web*-додатка
Тема 8. **Реалізація специфічних вимог до *Web*-додатків**					
Здатність спільно працювати над проектом та здійснювати інтеграцію його частин в єдиний додаток	Знання та вміння, отримані під час вивчення попередніх тем	Призначення служби *$route*. Стратегії перекладів користувальницького інтерфейсу	Налаштовувати маршрутизацію клієнтської частини *Web*-додатка	Презентувати прототип *Web*-додатка та уточнювати вимоги із замовником	Самостійно вирішувати завдання щодо інтеграції частин проекту в єдиний додаток

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Робоча програма навчальної дисципліни "РОЗРОБКА *WEB*-ДОДАТКІВ"

для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" всіх форм навчання

Самостійне електронне текстове мережеве видання

Укладач **Гіковатий** Володимир Михайлович

Відповідальний за видання *О. І. Пушкар*

Редактор *В. О. Бутенко*

Коректор *О. В. Анацька*

План 2016 р. Поз. № 157 ЕВ. Обсяг 42 с.

Видавець і виготовлювач – ХНЕУ ім. С. Кузнеця, 61166, м. Харків, просп. Науки, 9-А

Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру ***ДК № 4853 від 20.02.2015 р.***