Одеський національний політехнічний університет
Інститут комп'ютерних систем
Кафедра системного програмного забезпечення

Пояснювальна записка

до дипломної роботи бакалавра

на тему «Система обліку платних курсів. Модуль обліку платних курсів»

Виконав: студент IV курсу, групи АС-122
напряму підготовки
6.050103 — Програмна інженерія
Горбешко Б. М.
Керівник Оніщенко Т. В.
Рецензент Шапорін Р. О,

Одеса — 2016

Одеський національний політехнічний університет

Інститут комп'ютерних систем
Кафедра системного програмного забезпечення
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050103 — Програмна інженерія

ЗАТВЕРДЖУЮ

Завідувач кафедри СПЗ
_____ (В. В. Любченко)
«___»_____20___року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ
Горбешко Богдану Миколайовичу

1. Тема роботи «Система обліку платних курсів. Модуль обліку платних курсів»
керівник роботи старший викладач Оніщенко Тетяна Вікторівна
затверджені наказом ректора ОНПУ від «27» квітня 2016 року № 246-в
2. Строк подання студентом роботи 29.06.2016
3. Вихідні дані до роботи завдання на розробку, список рекомендованої літератури
4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити) Визначення бізнес-вимог, Проектування програмного продукту, Конструювання програмного продукту, Розгортання програмного продукту
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Мозкова карта, Діаграма варіантів використання, Діаграма WBS, Діаграма Ганта, Діаграма програмних класів, Алгоритм розрахунку вартості курсу, Алгоритм аутентифікації та авторизації, Діаграма концептуальних класів, Форма входу, Головне меню, Сповіщення

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 10.01.2016

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Специфікація вимог	10.01.16 – 01.02.16	
2	Планування проекту	02.02.16 – 23.02.16	
3	Проектування	24.02.16 – 17.03.16	
4	Реалізація	18.03.16 – 08.04.16	
5	Тестування	09.04.16 – 30.04.16	
6	Впровадження програми	01.05.16 – 22.05.16	
7	Написання пояснювальної записки	23.05.16 – 12.06.16	

Студент _____ Горбешко Б. М.

Керівник роботи _____ Оніщенко Т. В.

РЕФЕРАТ

Пояснювальна записка до дипломної роботи: 51 с., 11 рис., 46 табл., 2 додатки, 13 джерел.

Метою створення системи є розробка комп'ютерної системи обліку платних курсів для автоматизація набору слухачів на курси та реєстрацій сплат, а також для надання майбутнім слухачам можливості ознайомитися з наявними курсами та залишити заявку на реєстрацію за допомогою зовнішніх програмних сервісів.

Методи розробки базуються на мовах програмування PHP та JavaScript, сервері баз даних MySQL, Web-сервері Apache та бібліотеці Kendo UI.

Як результат роботи виконана програмна реалізація системи обліку платних курсів для кафедри СПЗ із можливістю створення звітів та підключення зовнішніх сервісів для прийому заявок.

Ключові слова: RIA, Apache, PHP, MySQL, бухгалтерський облік, JavaScript, Kendo UI.

ABSTRACT

The aim of work is the development of a paid course accounting computer system. The system is for the automation of picking listeners for courses and for the payment enrolling. Also it should enable potential listeners with the use of external software services to get informed with avaiable courses and to submit an application for registration.

Methods of developing technology are based on PHP and JavaScript programming languages, MySQL database server, Apache Web server and Kendo UI library.

Results: the software realization of a paid course accounting system is completed and the system supports creating reports and connecting external services for the accepting of applications.

Keywords: RIA, Apache, PHP, MySQL, accounts management, JavaScript, Kendo UI.

ЗМІСТ

ВСТУП

На сьогодні автоматизація документообороту так же необхідна, як автоматизація бухгалтерського обліку в середині 90-х років. Причин цьому багато. По-перше, інформацію необхідно обробляти якомога швидше та якісніше, інколи інформаційні потоки не менш важливі, ніж матеріальні. По-друге, втрата інформації чи її потрапляння в чужі руки може дуже дорого обійтися. Можна виділити ряд спільних для організацій, де робота з документами ведеться традиційним способом, проблем:

– документи губляться;
– накопичуються документи, призначення та джерело яких незрозумілі;
– документи та інформація, що в них знаходиться, потрапляє в чужі руки;
– витрачається багато часу на пошук потрібного документу;
– на підготовку і узгоду документів витрачається багато часу.

Автоматизація документообігу необхідна в будь-якій організації, незалежно від масштабу та типу власності. Але програми не можуть бути універсальними, покривати весь спектр функціональності різних організацій, при цьому залишаючись простими у використанні, надійними в роботі та доступними у фінансовому плані. Універсальність системи призводить до ускладнень у роботі системи, оскільки така система повинна забезпечити, окрім основної функціональності, ще й засоби конфігурації під певну предметну область.

Керування навчальними курсами, що проводяться на кафедрі ВНЗ, обов’язково супроводжується документами, кількість яких із часом зростає. Крім того, проводиться різноманітний аналіз наявних даних, що потребує деяких витрат часу, наприклад, підготовка звіту про боржників, підсумків успішності студентів. Тому створення ефективної системи документообігу з максимальної віддачею та мінімальними затратами є важливим завданням.

Наразі існує багато програмних продуктів, створених з метою автоматизації бізнес-процесів на підприємствах, але далеко не завжди університет має змогу придбати інформаційну систему необхідного рівня, не кажучи про те, що впровадження сторонніх розробок й адаптація програмного продукту до особливостей певної організації завжди породжує багато проблем.

У перспективі застосування систем для обліку можна розділити на вже готові прикладні застосування, платформи для конфігурації предметної області та

розробки абсолютно нової системи, що задовольняє всім прикладним задачам. Стрімко розвиваються системи конфігурування предметної області на кшталт «1С: Підприємство», що характеризуються стрімким зростанням попиту, розвитком; подібні платформи можна «глибоко» конфігурувати, що дозволяє адаптувати їх у великій кількості сфер діяльності для різних задач. Але у них є і недоліки: необхідність наявності на підприємстві експертів із підтримки та розробки таких систем (зокрема, програміста, та в деяких випадках — адміністратора), що спричиняє необхідність витрат на оплату їх послуг та, як наслідок — збільшує вартість розробки та обслуговування такої системи. [1]

Готові прикладні застосування, у свою чергу, не завжди можуть забезпечити повну функціональність для вирішення прикладних завдань, не завжди доступні для придбання у фінансовому плані та часто не надають ліцензії на зміну вихідного коду. Тоді підприємство, у випадку економічної вигідності (вартість розробки та супроводу прийнятна для замовника), приймає рішення щодо розробки нової системи, що повністю задовольняє вимогам.

Серед існуючих аналогів розглянуто системи для автоматизації бухгалтерського обліку та документообігу в організаціях, зокрема із готовими конфігураціями для навчальних закладів, систему ведення електронних курсів, а також платформу для формування форм та звітів. За результатами функціонального аналізу жоден аналог не покриває всю необхідну функціональність. При цьому деякі дозволяють реалізувати функціональність, якої бракує, окремим програмним модулем та інтегрувати його з системою, однак і таке використання не є виправданим з урахуванням малої частки покритої функціональності, обмежених можливостей засобів інтеграції та вартості систем.

Метою роботи є створення системи обліку платних курсів для покращення продуктивності в частині реєстрації слухачів та їх сповіщення про розклад курсів за рахунок підвищення функціональності в частині функціональної придатності шляхом розширення функції функціями для самостійного попереднього запису слухачів та ознайомлення з розкладом через мережу Інтернет.

Завданням роботи є розробка програмного продукту для підвищення якості роботи персоналу, що зумовлено реалізацією певних функціональних можливостей системи, таких як керування курсами, слухачами, викладачами, оплатами за курси, формування звітів, обробка заявок.

Створення програмного забезпечення супроводжується процесом розробки. Існує кілька моделей процесу розробки, кожна з яких описує свій підхід у вигляді задач чи діяльності, що має місце в ході процесу. Основними дисциплінами, з яких складається сам процес, є: бізнес-моделювання, аналіз вимог, планування, розробка архітектури, кодування, тестування, налагодження, документування, впровадження та супровід.

1 ВИЗНАЧЕННЯ БІЗНЕС-ВИМОГ

1.1 Визначення бізнес-вимог

1.1.1 Опис предметної області

Суб’єкти:

– Слухачі — особи, що можуть записуватися на курси або вже записані на деякі курси.

– Викладачі — особи, що ведуть певні курси.

– Оператори — допоміжний персонал та довірені особи, що займаються записом слухачів, приймають оплати та формують звіти.

– Адміністратори — особи, що безпосередньо завідують курсами, дають завдання операторам та можуть виконувати їх функції.

Для опису структури та об’єктів предметної області створено мозкову карту (рис. А.1).

На карті предметну область розділено за інформаційними потоками та властивостями, об’єктами та суб’єктами, які вони містять.

1.1.2 Назва продукту

«К&П 2014» — скорочення від «Курси та платежі», 2014 — рік розробки базової версії продукту.

1.1.3 Проблеми

1. Відсутність актуальної інформації про наявні курси в мережі Інтернет.
2. Необхідність слухачам відвідувати кафедру СПЗ для реєстрації на курс, черги, ручне введення контактних даних через операторів.
3. Зберігання відомостей про курси у текстових документах та ручне формування звітів.
4. Слухачі забувають розклад курсів, а викладачі шукають вільну аудиторію для проведення курсів.

1.1.4 Аналогічні системи

Проведено функціональний аналіз аналогічних систем (таблиця 1.1) [2] [3] [4] [5] [6].

Таблиця 1.1 — Функціональний аналіз аналогічних систем

Проблема / Назва	1	2	3	4	5
1С: Бухгалтерія	Зовнішній сайт + web-сервіс		Бібліотека стандартних підсистем + бібліотека електронних документів	Бібліотека електронних документів	Зовнішня розсилка сповіщень + web-сервіс
Парус	Зовнішній сайт + сервіс реплікації		Конструктор галузевих рішень + сервіс звітності		Зовнішня розсилка сповішень + сервіс реплікації
1С-Бітрікс: Внутрішній портал навчального закладу	Групи за предметами		—	—	Жива стрічка, календар
Moodle	Створення та пошук курсів	Самостійна реєстрація на курси	—	—	Календар
Веб-застосунок MS Access + SharePoint	Представлення списку	Форми	Шаблонні звіти або інтеграція з MS Access для робочого столу для формування складних звітів		Представлення таблиць, розсилка сповіщень за допомогою макросів MS Access для робочого столу

Колонки відповідають номерам проблем з пункту 1.1.3. Вказано не тільки наявні реалізації функцій, але й наявні засоби для їх реалізації, якщо такі функції не реалізовано.

1.1.5 Цілі

– Зменшення часу реєстрації на курс

– Зменшення часу формування звітів з даних про курси

– Зменшення часу отримання інформації про наявні курси, розклад курсів та вільні аудиторії

1.2 Визначення власних вимог

1.2.1 Діаграма варіантів використання

На рис. А.2 наведено діаграму варіантів використання. Умовні позначення: C — створення, R — перегляд, U — редагування, D — видалення.

Багато варіантів містять CRUD-операції, при цьому окремі операції доступні різним наборам акторів. Слід приділити цьому особливу увагу при розробці.

1.2.2 Опис варіантів використання

У таблицях 1.2 – 1.27 наведено сценарії варіантів використання системи рівня користувача.

Таблиця 1.2 — Видалити курс

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Оператор бажає прибрати з системи помилково створений або скасований курс
Передумови	Курс є в системі, оператор аутентифікований та авторизований
Гарантії успіху	Курс щез зі списку курсів
Основний сценарій	1. Оператор обирає курс.
	2. Оператор видаляє обраний курс.
Розширення	—

Таблиця 1.3 — Додати курс

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Заплановано новий курс і оператор має намір відкрити на нього набір слухачів
Передумови	Курс відсутній в системі, оператор аутентифікований та авторизований
Гарантії успіху	Курс з’явився в системі
Основний сценарій	1. Оператор створює новий курс.
	2. Оператор вводить інформацію про курс.
	3. Оператор зберігає курс.
Розширення	2.А. Оператор не ввів назву курсу.
	2.А.1. Система вимагає ввести назву курсу.
	2.А.2. Перехід до п. 2.

Таблиця 1.4 — Редагувати курс

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Змінилися характеристики існуючого курсу й оператор бажає оновити його дані в системі
Передумови	Курс є в системі, оператор аутентифікований та авторизований
Гарантії успіху	Система видає нові дані курсу
Основний сценарій	1. Оператор обирає курс.
	2. Оператор змінює потрібні дані.
	3. Оператор зберігає курс.
Розширення	—

Таблиця 1.5 — Редагувати слухача

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Змінено чи уточнено ПІБ чи контактні дані слухача й оператор бажає оновити їх у системі
Передумови	Слухач є в системі, оператор аутентифікований та авторизований
Гарантії успіху	Система видає нові дані слухача
Основний сценарій	1. Оператор обирає слухача.
	2. Оператор змінює потрібні дані.
	3. Оператор зберігає слухача.
Розширення	—

Таблиця 1.6 — Видалити слухача

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Слухач так і не записався на жоден курс і тримати з ним зв’язок більше немає сенсу, тому оператор бажає видалити його з системи
Передумови	Слухач є в системі, оператор аутентифікований та авторизований
Гарантії успіху	Система не видає слухача
Основний сценарій	1. Оператор обирає слухача.
	2. Оператор видаляє обраного слухача.
Розширення	—

Таблиця 1.7 — Переглянути курси

Діючі особи	Слухач, оператор
Область дії	Система
Рівень	Мета користувача

Продовження таблиці 1.7

Учасники та інтереси	Користувач бажає отримати перелік наявних у системі курсів для ознайомлення або, якщо це оператор — для виконання операцій над ними
Передумови	Користувач аутентифікований та авторизований
Гарантії успіху	Користувач отримав перелік курсів
Основний сценарій	1. Користувач запитує перегляд курсів.
Розширення	—

Таблиця 1.8 — Додати слухача

Діючі особи	Оператор, слухач
Область дії	Система, кафедра СПЗ
Рівень	Мета користувача
Учасники та інтереси	З’явився новий слухач, що не записувався через мережу Інтернет, і оператор бажає додати його до системи
Передумови	Слухач відсутній в системі, оператор аутентифікований та авторизований
Гарантії успіху	Слухач з’явився в системі
Основний сценарій	1. Оператор створює нового слухача.
	2. Оператор запитує невідому йому інформацію у слухача.
	3. Оператор вводить інформацію про слухача.
	4. Оператор зберігає слухача.
Розширення	—

Таблиця 1.9 — Переглянути слухача

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Оператор бажає отримати перелік наявних у системі слухачів для ознайомлення чи виконання операцій над ними
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	Оператор отримав перелік слухачів

Продовження таблиці 1.9

Основний сценарій	1. Оператор запитує перегляд слухачів.
Розширення	—

Таблиця 1.10 — Створити звіт за період

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Оператор бажає передати друкований звіт у бухгалтерію у кінці певного навчального періоду
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	Отримано придатний до друку документ, що містить звітність за курсами з заданого періоду
Основний сценарій	1. Оператор обирає звіт за період.
	2. Оператор задає діапазон дат.
	3. Оператор підтверджує створення звіту.
Розширення	2.А. Кінцева дата менше початкової або діапазон майбутній.
	2.А.1. Система повідомляє про невірний діапазон дат. Перехід до п.2.

Таблиця 1.11 — Створити звіт за курсом

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Оператор бажає передати друкований звіт у бухгалтерію по закінченню певного курсу
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	Отримано придатний до друку документ, що містить звітність за заданим курсом
Основний сценарій	1. Оператор обирає курс.
	2. Оператор викликає створення звіту.
Розширення	—

Таблиця 1.12 — Видалити розклад

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Курс перенесено на невизначений термін і поточний розклад для нього більше не актуальний, тож оператор бажає видалити цей розклад з системи
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	У курсу немає розкладу
Основний сценарій	1. Оператор обирає курс.
	2. Оператор видаляє розклад для курсу.
Розширення	—

Таблиця 1.13 — Створити розклад

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Для нового курсу затверджено розклад і оператор бажає додати його до системи
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	На курс призначено введений розклад
Основний сценарій	1. Оператор обирає курс.
	2. Оператор вводить дату проведення заняття та вказує тип заняття (лекція чи практика).
	3. Оператор повторює п. 2 для всіх призначених занять.
Розширення	2.А. Дата заняття виходить за діапазон дат проведення курсу.
	2.А.1. Система повідомляє про невірну дату. Перехід до п.2.

Таблиця 1.14 — Редагування розкладу

Діючі особи	Оператор

Продовження таблиці 1.14

Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	У розклад курсу внесено зміни й оператор бажає відбити їх у системі
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	На курс призначено відредагований розклад
Основний сценарій	1. Оператор обирає курс.
	2. Оператор додає заняття, яких немає у збереженій версії розкладу.
	3. Оператор видаляє неактуальні заняття, які є у збереженій версії розкладу.
	4. Оператор змінює дату для занять, які перенесено.
	5. Оператор змінює тип для занять, які змінили тип порівняно зі збереженою версією розкладу.
Розширення	4.А. Дата заняття виходить за діапазон дат проведення курсу.
	4.А.1. Система повідомляє про невірну дату. Перехід до п.4.

Таблиця 1.15 — Створити заявку

Діючі особи	Слухач, система
Область дії	Зовнішня система
Рівень	Мета користувача
Учасники та інтереси	Слухач бажає зареєструватися на курс
Передумови	Оператор аутентифікований та авторизований у зовнішній системі
Гарантії успіху	Заявка з’явилася у сповіщення для операторів системи
Основний сценарій	1. Слухач обирає курс.
	2. Слухач вводить свої ПІБ та контактні дані.
	3. Слухач відправляє заявку, зовнішня система передає заявку системі.

Продовження таблиці 1.15

Розширення	—

Таблиця 1.16 — Переглянути розклад

Діючі особи	Слухач, оператор, викладач
Область дії	Система, зовнішня система
Рівень	Мета користувача
Учасники та інтереси	Користувач бажає переглянути розклад для певного курсу
Передумови	Оператор або викладач — аутентифікований та авторизований у системі; слухач — у зовнішній системі
Гарантії успіху	Користувач отримав розклад для обраного курсу
Основний сценарій	1. Користувач обирає курс.
	2. Користувач запитує перегляд розкладу.
Розширення	—

Таблиця 1.17 — Обробити новий обліковий запис

Діючі особи	Адміністратор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Адміністратор має намір обробити нові сповіщення
Передумови	Адміністратор аутентифікований та авторизований
Гарантії успіху	Обліковий запис доступний для аутентифікації або видалений
Основний сценарій	1. Адміністратор обирає сповіщення.
	2. Адміністратор підтверджує сповіщення.
Розширення	2.А. Адміністратор відхилює сповіщення.

Таблиця 1.18 — Переглянути облікові записи

Діючі особи	Адміністратор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Адміністратор бажає ознайомитися з тим, які користувачі наразі є в системі та які вони мають рівні прав доступу

Продовження таблиці 1.18

Передумови	Адміністратор аутентифікований та авторизований
Гарантії успіху	Адміністратор отримав перелік облікових записів
Основний сценарій	1. Адміністратор запитує перегляд облікових записів.
Розширення	—

Таблиця 1.19 — Скинути пароль

Діючі особи	Адміністратор, користувач
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	До адміністратора особисто звернувся користувач, який забув пароль, і адміністратор згоден скинути пароль, щоб користувач встановив новий
Передумови	Адміністратор аутентифікований та авторизований
Гарантії успіху	Користувач може повторно аутентифікуватися з новим паролем
Основний сценарій	1. Адміністратор обирає обліковий запис користувача.
	2. Адміністратор викликає скидання паролю.
	3. Користувач заходить в систему зі своїм логіном та новим паролем.
Розширення	—

Таблиця 1.20 — Видалити обліковий запис

Діючі особи	Адміністратор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Адміністратор бажає видалити обліковий запис користувача, якому більше не потрібен доступ до системи у якості оператора, адміністратора чи викладача
Передумови	Адміністратор аутентифікований та авторизований
Гарантії успіху	Обраний обліковий запис більше недоступний для аутентифікації та відсутній у переліку облікових записів

Продовження таблиці 1.20

Основний сценарій	1. Адміністратор обирає обліковий запис.
	2. Адміністратор видаляє обраний обліковий запис.
Розширення	—

Таблиця 1.21 — Налаштувати коефіцієнт

Діючі особи	Адміністратор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Змінилися коефіцієнти розрахунку вартості курсу і адміністратор бажає відкорегувати автоматичний розрахунок
Передумови	Адміністратор аутентифікований та авторизований
Гарантії успіху	Вартості курсу розраховуються з урахуванням нового коефіцієнту
Основний сценарій	1. Адміністратор обирає коефіцієнт.
	2. Адміністратор встановлює нове значення коефіцієнту.
Розширення	—

Таблиця 1.22 — Обробити заявку

Діючі особи	Оператор, слухач
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Оператор має намір обробити нові сповіщення
Передумови	Оператор аутентифікований та авторизований
Гарантії успіху	Слухач із вказаними в заявці даними з’явився в системі та записаний на вказаний у заявці курс або заявка просто зникла зі сповіщень
Основний сценарій	1. Оператор обирає сповіщення.
	2. Оператор підтверджує додання даних з заявки.
	3. Система впевнюється, що слухача, який подав заявку, ще не додано до системи.
	4. Слухач приходить на кафедру СПЗ і підписує договір.

Продовження таблиці 1.22

Розширення	2.А. Оператор відхиляє додання даних з заявки.
	3.А. Слухач, що подаяв заявку, вже існує в системі.
	3.А.1. Оператор обирає слухача.
	3.А.2. Оператор зливає дані користувача з даними заявки. Перехід до п. 4.

Таблиця 1.23 — Авторизуватися

Діючі особи	Користувач
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Користувач має намір проводити певні дії у системі
Передумови	Користувач має обліковий запис у системі
Гарантії успіху	Користувач авторизований
Основний сценарій	1. Користувач вводить логін та пароль.
Основний сценарій	1.А. Користувач із таким логіном не існує або пароль невірний.
	2. Система повідомляє про відхилення авторизації. Перехід до п.1.

Таблиця 1.24 — Зареєструватися

Діючі особи	Користувач
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Користувач має намір проводити певні дії у системі
Передумови	Користувач не має облікового запису у системі
Гарантії успіху	Адміністратори отримали сповіщення про реєстрацію нового користувача, обліковий запис не доступний для аутентифікації

Продовження таблиці 1.24

Основний сценарій	1. Користувач вводить логін та пароль.
Розширення	1.А. Користувач із таким логіном вже існує.
	1.А.1. Система повідомляє слухачеві, що слід обрати інший логін. Перехід до п.1.

Таблиця 1.25 — Записати слухача на курс

Діючі особи	Оператор, слухач
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Слухач сповістив, що бажає записатися на курс, але не подавав заявку через мережу Інтернет, і оператор бажає самостійно записати його на курс
Передумови	Слухач присутній у системі, оператор аутентифікований та авторизований
Гарантії успіху	Слухач є серед слухачів курсу
Основний сценарій	1. Оператор обирає слухача.
	2. Оператор обирає курс.
	3. Оператор записує слухача на курс.
	4. Слухач підписує договір.
Розширення	—

Таблиця 1.26 — Відрахувати слухача з курсу

Діючі особи	Оператор
Область дії	Система
Рівень	Мета користувача
Учасники та інтереси	Слухач передумав або не сплатив вчасно курс і не відвідує заняття
Передумови	Слухач присутній у системі і записаний на курс, оператор аутентифікований та авторизований
Гарантії успіху	Слухача немає серед слухачів курсу
	1. Оператор обирає слухача.

Продовження таблиці 1.26

	2. Оператор обирає курс.
	3. Оператор запитує сплачену слухачем суму. Система відповідає.
	4. Оператор повертає слухачеві сплачені кошти.
Основний сценарій	5. Оператор відраховує слухача з курсу.
	3.А. Слухач не сплачував кошти за курс.
Розширення	3.А.1. Перехід до п.5.

Таблиця 1.27 — Прийняти оплату

Діючі особи	Оператор, слухач
Область дії	Система, кафедра СПЗ
Рівень	Мета користувача
Учасники та інтереси	Слухач приніс певну суму коштів і оператор бажає внести її до системи
Передумови	Слухач присутній у системі і записаний на курс, оператор аутентифікований та авторизований
Гарантії успіху	Загальна оплата слухачем за курс змінилася на прийняту суму
	1. Оператор приймає та перераховує кошти.
	2. Оператор обирає слухача.
	3. Оператор обирає курс.
	4. Оператор впевнюється, що слухач має заборгованість за курс.
Основний сценарій	5. Оператор вводить суму коштів.
	4.А. Слухач приніс більше коштів, ніж має заборгованість.
	4.А.1. Оператор вводить суму, рівну боргу слухача.
	4.А.2. Оператор повертає слухачеві здачу.
	4.Б. Слухач не має заборгованості.
Розширення	4.Б.1. Оператор повертає слухачеві кошти.

Розглянуто основні та очікувані альтернативні сценарії. Альтернативні сценарії впливають здебільшого на необхідність попередньої валідації даних.

1.3 Функціональні вимоги

Узагальнені варіанти використання включено до функціональних вимог та проаналізовано за принципом MoSCoW [7]:

– Життєвий цикл
– Організаційні процеси
– Навчання
– Опанування Kendo UI
– Створення інфраструктури
– Керування проектом
– Основні процеси
– Розробка
– Аналіз вимог
– [M] BB: реєстрація
– [M] BB: авторизація
– [M] BB: робота з викладачами
– [M] BB: робота з курсами
– [W] BB: робота з розкладом
– [M] BB: робота зі слухачами
– [M] BB: запис слухачів на курс
– [M] BB: робота з обліковими записами
– [C] BB: робота з коефіцієнтами
– [S] BB: створення звітів
– [S] BB: обробка заявок
– [M] BB: обробка облікових записів
– Впровадження
– Допоміжні процеси
– Документування
– Забезпечення якості
– Модульне тестування
– Тестування інтерфейсу користувача
– Вирішення проблем

1.4 Нефункціональні вимоги

– Відображати та редагувати дані, що вводяться в систему, у таблицях

– Відображати на екрані одну або дві різні таблиці одночасно

– Вкладені таблиці для сутностей, пов’язаних зв’язком M–M

– Гарячі клавіші для додання нового запису, видалення запису, збереження запису

– Підтримка стабільних версій браузерів Firefox та Chrome, Internet Explorer 8–11

– Час реакції $\leqslant$ 5 секунд

– Імовірність збою — 0.01

– Підтримка резервного копіювання даних та відновлення з резервних копій

1.5 Планування розробки

Представлення: клієнт. Бізнес-логіка: сервер. Дані: БД.

Технології розробки: мова PHP5.4, СКБД MySQL 5, Web-технології, формат даних JSON.

Інструменти розробки: редактор Vim, система контролю версій Git, бібліотека тестування запитів Dogpatch.

Діаграми: рис. А.3, А.4.

2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

2.1 Концептуальне проектування

Діаграму концептуальних класів наведено на рис. 2.1.

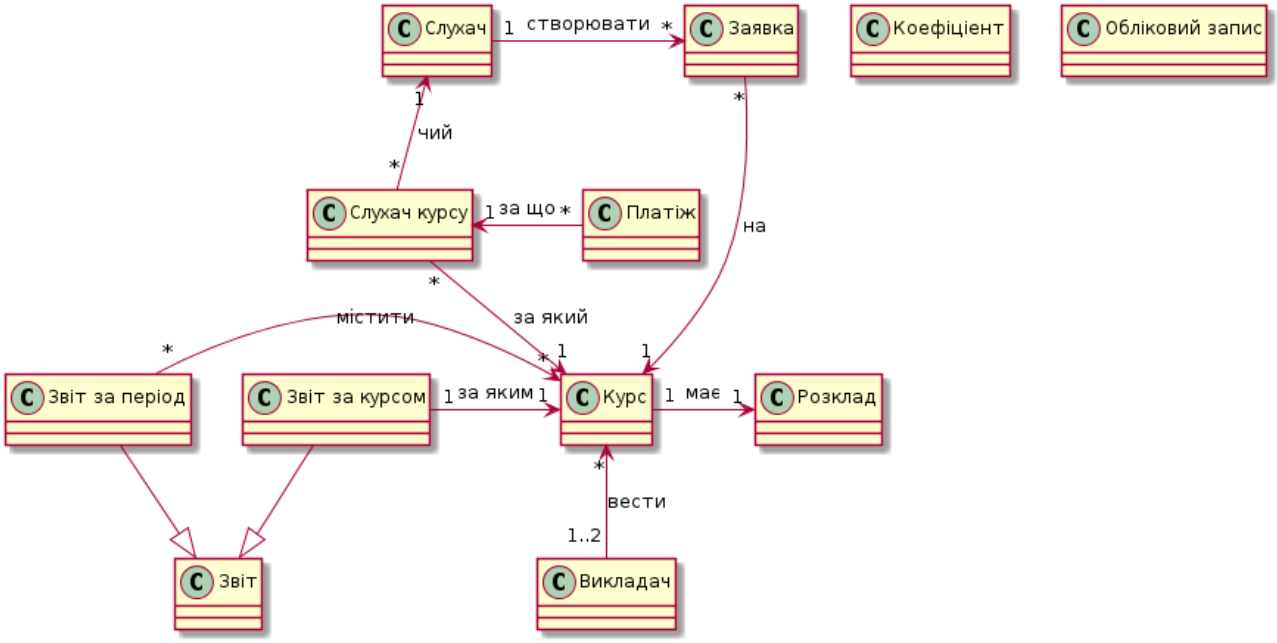

Рисунок 2.1 — Діаграма концептуальних класів

Клас «Курс» характеризується високою зв’язністю [8], тобто це головний клас у системі.

2.2 Логічне проектування

Діаграму програмних класів наведено на рис. А.5.

Як і на діаграмі концептуальних класів, клас Course характеризується високою зв’язністю та має найбільшу кількість атрибутів. Також високу зв’язність має AccessControl, що відповідає висновку з пункту 1.2.1.

2.3 База даних

Таблиця 2.1 — Опис структури таблиці «Слухачі» (listeners)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idListener	int(6)	Ні	A_I
	Ім’я	Name	varchar(30)	Так	
	Прізвіще	Surname	varchar(30)	Ні	

Продовження таблиці 2.1

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
	По батькові	Patronymic	varchar(30)	Так	
	Університетська група	UGroup	varchar(6)	Так	
	Телефон	Phone	varchar(20)	Так	
	E-mail	Email	varchar(40)	Так	
FK	Ким змінено	affectedBy	varchar(32)	Ні	

Таблиця 2.2 — Опис структури таблиці «Курси» (courses)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idCourse	int(6)	Ні	A_I
	Назва	Name	varchar(50)	Ні	
	Опис	Description	varchar(400)	Так	
	Ознака індивідуального курсу	isIndividual	bit(1)	Ні	
FK	Викладач	idTeacher	int(6)	Так	
FK	Другий викладач	idTeacher2	int(6)	Так	
	Ціна	Price	decimal(10,2)	Так	
	Стан	state	tinyint(1)	Ні	0..2 (Йде набір, набрано, завершений)
FK	Ким змінено	affectedBy	varchar(32)	Ні	

Таблиця 2.3 — Опис структури таблиці «Слухачі курсу» (Course_Listeners)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idCL	int(6)	Ні	A_I
FK	Курс	idCourse	int(6)	Ні	
FK	Слухач	idListener	int(6)	Ні	
	Оцінка	mark	tinyint(3)	Так	
FK	Ким змінено	affectedBy	varchar(32)	Ні	

Продовження таблиці 2.3

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.

Таблиця 2.4 — Опис структури таблиці «Платежі» (payments)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idPayment	int(6)	Ні	A_I
FK	Слухач курсу	idCL	int(6)	Ні	
	Кошти	delta	int(6)	Ні	
	Мітка часу	timestamp	timestamp	Ні	CURRENT_ TIMESTAMP

Таблиця 2.5 — Опис структури таблиці «Викладачі» (teachers)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idTeacher	int(6)	Ні	A_I
	Ім’я	Name	varchar(30)	Так	
	Прізвіще	Surname	varchar(30)	Ні	
	По батькові	Patronymic	varchar(30)	Так	
	Телефон	Phone	varchar(20)	Так	
	E-mail	Email	varchar(40)	Так	
FK	Ким змінено	affectedBy	varchar(32)	Ні	
FK	Вчений ступінь	degree	tinyint(2)	Так	

Таблиця 2.6 — Опис структури таблиці «Розцінки» (prices)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	degree	tinyint(2)	Ні	A_I
	Вчений ступінь	deglab	varchar(30)	Ні	
	Зарплата	salary	decimal(10,2)	Ні	

Таблиця 2.7 — Опис структури таблиці «Коефіцієнти» (coefficients)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Назва	name	varchar(64)	Ні	
	Мітка	label	varchar(128)	Так	
	Значення	value	float	Так	

Таблиця 2.8 — Опис структури таблиці «Заняття» (lessons)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Ідентифікатор	idLesson	int(11)	Ні	A_I
FK	Курс	idCourse	int(11)	Ні	
	Дата	date	date	Так	
	Час початку	time	time	Ні	
	Тип	type	tinyint(1)	Ні	0..1 (Лекція, Практика)

Таблиця 2.9 — Опис структури таблиці «Користувачі» (users)

Ключ	Назва	Ім’я поля	Тип	NULL	Дод.
PK	Логін	login	varchar(32)	Ні	
	Хеш паролю	password	text	Ні	
	Сіль	salt	text	Ні	
	Тип	type	tinyint(1)	Ні	0..2 (Оператор, Адміністратор, Переглядач)
	Ключ сесії	sessionid	text	Так	

2.4 Оцінка алгоритмічної складності

На рис. А.6 зазначено складну операцію розрахунку вартості за даними з БД. Вона реалізується наступним SQL-запитом:

```
SELECT ifnull(s1,0)*ifnull(c1,0)+ifnull(s2,0)*ifnull(c2,0) FROM ((
 SELECT salary as s1 FROM prices WHERE degree IN (
  SELECT degree FROM teachers WHERE idTeacher IN (
   SELECT idTeacher FROM courses WHERE idCourse=?
))) s1 LEFT OUTER JOIN (
 SELECT salary as s2 FROM prices WHERE degree IN(
  SELECT degree FROM teachers WHERE idTeacher IN (
   SELECT idTeacher2 FROM courses WHERE idCourse=?
))) s2 ON 1=1) LEFT OUTER JOIN ((
 SELECT count(*) as c1 FROM lessons WHERE idCourse=? AND type=0
) c1 ON 1=1 LEFT OUTER JOIN (
 SELECT count(*) as c2 FROM lessons WHERE idCourse=? AND type=1
) c2) ON 1=1;
```

План виконання запиту:

1. Вибірка вмісту таблиці courses
2. Проекція кортежів за значенням атрибуту idCourse
3. Проекція атрибутів за атрибутом idCourse
4. Вибірка вмісту таблиці teachers
5. Проекція кортежів за значенням атрибуту idTeacher
6. Проекція атрибутів за атрибутом degree
7. Вибірка вмісту таблиці prices
8. Проекція кортежів за значенням атрибуту degree
9. Проекція атрибутів за атрибутом salary
10. Використання вибірки з п. 1.
11. Проекція кортежів за значенням атрибуту idCourse
12. Проекція атрибутів за атрибутом idCourse
13. Використання вибірки з п. 4.
14. Проекція кортежів за значенням атрибуту idTeacher
15. Проекція атрибутів за атрибутом degree
16. Використання вибірки з п. 7.
17. Проекція кортежів за значенням атрибуту degree
18. Проекція атрибутів за атрибутом salary
19. Вибірка вмісту таблиці lessons
20. Проекція кортежів за значенням атрибуту idCourse
21. Проекція кортежів за значенням атрибуту type
22. Підрахунок кортежів
23. Використання проекції з п. 20.
24. Проекція кортежів за значенням атрибуту type
25. Підрахунок кортежів
26. З’єднання результатів пп. 9 та 18.
27. З’єднання результатів пп. 22 та 25.
28. З’єднання результатів пп. 26 та 27.
29. Розрахунок математичного виразу за результатом п. 28.

Операції проекції кортежів у пп. 2, 3, 5, 6, 8, 9, 11, 12, 14, 15, 17 та 18 повинні повертати один кортеж. Пошук у таблицях courses та teachers відбувається за первинним ключем, а таблиця prices складається з кількох кортежів. Для проекцій з пп. 20, 21 та 24 доцільно створити у таблиці lessons індекс за атрибутом idCourse.

За атрибутом type індекс недоцільний, оскільки проекції за ним відбуваються не з усієї таблиці, а можливих значення тільки два. SQL-запит для створення індексу:

```
CREATE INDEX course_of_lesson ON lessons (idCourse);
```

Циклів в алгоритмах на рис. А.6 та А.7 немає, тож їх складність — *O(1)* [9].

2.5 Опис зовнішних інтерфейсів

Інтерфейс користувача побудовано із використанням бібліотеки Kendo UI, дизайн визначається наявними для неї темами оформлення. Доступні самостійні концептуальні класи присутні у головному меню та розташовані за частотою використання: найактивніша робота проводиться зі слухачами та курсами, викладачі та користувачі заповнюються на початку роботи і потім змінюються рідко, звіти здаються рідко, розцінки та коефіцієнти змінюються у виключних випадках. В кінці головного меню розміщено кнопку виходу з системи. Екран авторизації є окремим та лаконічним, містить лише форму та кнопку перемикання форм реєстрації та входу.

Для роботи серверної частини системи потрібен веб-сервер Apache 2.x, інтерпретатор PHP5 не нижче 5.4, СКБД MySQL або MariaDB 5. Для користування клієнтською частиною потрібен web-браузер із підтримкою EcmaScript 5; тестування проводиться у поточних версіях браузерів Mozilla Firefox та Chromium для десктопу.

Система може працювати у межах однієї машини, через локальну мережу та через мережу Інтернет, в залежності від мережевих підключень та налаштувань машини, на якій встановлено серверну частину. Підключення, відповідно, може здійснюватись будь-яким доступним дротовим або бездротовим каналом підключення до локальної мережі або мережі Інтернет: Ethernet, Wi-Fi, ADSL, HSPA, Dial-up та ін. Оскільки звіти генеруються у форматі PDF, для їх перегляду потрібна програма — переглядач PDF: Mozilla Firefox, Google Chrome, Adobe Reader, Foxit Reader тощо. [10] Для друку потрібен принтер, підключений безпосередньо до машини, на якій відкрито файл звіту, або доступний для неї через мережу.

3 КОНСТРУЮВАННЯ ПРОГРАМНОГО ПРОДУКТУ

3.1 Опис програмних технологій

Система поділяється на дві частини: сервер та клієнт.

Сервер реалізовано на мові PHP5. [11] Це скриптова мова програмування, орієнтована на задачі web-розробки. Завдяки відсутності етапу компіляції зменшується час розробки та відлагодження. Динамічна типізація мови дозволяє прозоро працювати як з числами, так і з їх текстовим представленням, а також спрощує перевірку наявності об'єктів у булевих виразах. Мова надає вбудовані засоби для обробки рядків, параметрів HTTP-запитів та взаємодії з різними СКБД. Окрім цього, PHP є єдиною підтримуваною мовою на багатьох хостингах для web-сайтів та web-застосунків. З цієї ж причини у якості СКБД обрано MySQL 5. Задля використання нових можливостей, зокрема, скороченого оголошення масивів, мінімальною підтримуваною версією є PHP 5.4.

Клієнт реалізовано на скриптовій мові програмування ECMAScript 5 (більше відомій як JavaScript) та описових мовах HTML 5 та CSS 3. Вони є фактичними web-стандартами та підтримуються сучасними web-браузерами, такими як Firefox, Chrome, Edge, Safari, Internet Explorer, Android Browser тощо. Альтернативи не є кросбраузерними, приміром, мова VBScript підтримується лише у Internet Explorer, а мова Dart — лише у Chrome. Крім того, використання web-технологій дозволяє створювати на основі клієнту застосунки для настільних та мобільних комп'ютерів за допомогою спеціальних середовищ на основі браузерних рушіїв (напр., node-webkit, PhoneGap, AppJS).

3.2 Опис програмних бібліотек

Оскільки клієнт є «товстим», на сервер покладені лише задачі передачі та обробки даних між клієнтом та базою даних, тож немає потреби у використанні систем керування контентом, фреймворків та тому подібних бібліотек. Проте мова PHP не має вбудованих засобів для побудови PDF-документів, тож для цієї задачі використано бібліотеку MPDF, яка формує документ з HTML-шаблону. Серед альтернатив розглядалася також бібліотеку TCPDF, проте при спробі використання вона виявила недостатню підтримку можливостей HTML та CSS.

Модульні тести для сервеу також написані мовою PHP та використовують бібліотеку Dogpatch, розширену та доповнену для потреб тестування системи. Бі-

бліотека DogPatch дозволяє виконувати HTTP-запити та таким чином перевіряти, чи вірні відповіді та HTTP-коди надають запити за певних умов. Додано функції для обробки відповідей у JSON, що дозволяють перевірити наявність певного об’єкту у масиві об’єктів.

Клієнт використовує бібліотеки jQuery та Kendo UI. jQuery є прошарком над API, що доступні для браузерного JavaScript, який надає компактний синтаксис та кросбраузерні реалізації для таких частовживаних операцій, як робота з DOM та AJAX-запитами. Kendo UI надає елементи керування (віджети) із широкою фукціональністю, у тому числі такі, що не надаються засобами HTML. Крім того, вона надає багатофункціональний елемент керування для представлення та редагування табличних даних, а також самостійно здійснює запити з цими даними до серверу, слід лише описати URL та формати даних. [12] Для призначення гарячих клавіш використовується плагін для jQuery jquery.hotkeys.

3.3 Особливості створення програмних модулів з урахуванням мови програмування

Інтерпретатор та мова PHP5 розраховані на використання у якості CGI, тобто web-сервер використовує URI як відносний шлях до файлу, запускає файл на виконання та віддає результат його роботи. Альтернативні способи використання URI, зокрема, маскування структури програмних модулів за допомогою так званих «людинозрозумілих URL» та маршрутизації запитів до викликів методів класів вимагають низькорівневих перехоплень за допомогою правил ModRewrite для Apache та аналогічних засобів для інших web-серверів. Тому натомість архітектуру серверної частини системи реалізовано без застосування об’єктно-орієнтованого програмування, проте із наслідуванням деяких його принципів. Модулі організовані по директоріях за концептуальними класами і реалізують інтерфейси шляхом розміщення у директоріях файлів з однаковими іменами, способами передачі параметрів та форматами вхідних та вихідних даних — це дозволяє за реалізації роботи з різними концептуальними класами у клієнті змінювати лише назву директорії. Функції для зв’язку між концептуальними класами винесено в окремі модулі, спільні функції винисено в окремі бібліотеки функцій.

Клієнт реалізовано як односторінковий застосунок. Головний інтерфейс системи реалізований HTML-сторінкою з меню; за натисненням пунктів меню ство-

рюються та відкриваються певні вікна засобами JavaScript. Проте сторінка входу та реєстрації, як незалежна від головного інтерфейсу сутність, винесена в окрему HTML-сторінку, до якої не підключено JavaScript-бібліотек. Обидві сторінки можуть змінюватись в залежності від режиму, повідомлень системи, прав доступу тощо, тому виводяться скриптами на мові PHP, що є HTML-шаблонами зі вставками коду на PHP. Власних функцій на JavaScript небагато і слугують вони здебільшого прошарком між описами даних та реалізованими засобами бібліотеки Kendo UI елементами керування, тож зібрані в один модуль. За розвитку системи та додання нових функцій може знадобитися декомпозиція цього модулю.

3.4 Особливості створення структур даних

У якості формату передачі даних від сервера до клієнту обрано JSON. Це текстовий формат на основі мови JavaScript, що надає компактний безнадлишковий (порівняно з похідними від SGML мовами) синтаксис та дозволяє структурувати рядки та числа у вкладених масивах та об’єктах (асоціативних масивах). [13] JSON формується засобами мови PHP, що доступні з версії 5.1. Дані від клієнта до сервера передаються у тілі POST-запиту у форматі HTML-форм. Структури даних, що використовуються при роботі з бібліотекою Kendo UI, визначаються документацією до цієї бібліотеки.

3.5 Модульне тестування

Перед початком тестування слід заповнити порожню базу даних тестовими даними (табл. 3.1–3.7). Тестування алгоритму розрахунку вартості курсу проведено методом «білої скрині» (таблиця 3.8), алгоритму аутентифікації та авторизації — методом «чорної скрині» (таблиця 3.9).

Таблиця 3.1 — Тестові дані для таблиці courses

Атрибут	Значення	
idCourse	1	2
Name	a	b
Description	a	b

Продовження таблиці 3.1

Атрибут	Значення	
isIndividual	0	0
idTeacher	1	1
idTeacher2	2	2
Price	15	0
state	0	0
affectedBy	user1	user2

Таблиця 3.2 — Тестові дані для таблиці Course_Listeners

Атрибут	Значення			
idCL	1	2	3	4
idCourse	1	1	1	2
idListener	1	2	3	1
mark	0	0	0	0
affectedBy	user1	user1	user1	user1

Таблиця 3.3 — Тестові дані для таблиці teachers

Атрибут	Значення	
idTeacher	1	2
Name	a	b
Surname	a	b
Patronymic	a	b
Phone	289	2893
Email	b@a.b	c@d.c
degree	1	2
affectedBy	user1	user2

Таблиця 3.4 — Тестові дані для таблиці prices

Атрибут	Значення		
degree	1	2	3
deglab	Б. с.	Доц. к. т. н.	Проф.
salary	15.62	23.24	35.38

Таблиця 3.5 — Тестові дані для таблиці lessons

Атрибут	Значення					
idLesson	1	2	3	4	5	6
idCourse	1	1	1	2	2	2
date	1.4.15	2.4.15	3.4.15	4.4.15	5.4.15	6.4.15
time	13:00	13:00	13:00	13:00	13:00	13:00
type	1	1	2	1	2	2

Таблиця 3.6 — Тестові дані для таблиці users

Атрибут	Значення		
login	adm1	user1	user2
password	$hash('asdf')$	$hash('qwer')$	$hash('zxcv')$
salt	$< hash1 >$	$< hash2 >$	$< hash3 >$
type	0	1	2
sessionid	$< key1 >$	NULL	q

Таблиця 3.7 — Тестові дані для таблиці coefficients

Атрибут	Значення		
name	bonus	others	personal
label	Нарахування на заробітну плату	Інші послуги та утримки	Зарплата персоналу
value	1.12	1.23	1.05

Таблиця 3.8 — Тестові випадки для алгоритму розрахунку вартості курсу

Test Case №	**Вхідні дані**		**Очікуваний результат**	**Результат тестування (успішний (passed) / неуспішний (failed))**
	id	full		
1	1	true	60	Passed
2	2	true	15	Passed
3	1	false	20	Passed
4	2	false	15	Passed

Таблиця 3.9 — Тестові випадки для алгоритму аутентифікації та авторизації

Test Case №	**Вхідні дані**		**Очікуваний результат**	**Результат тестування (успішний (passed) / неуспішний (failed))**
	login	pass		
1	’adm1’	’asdf’	Вхід	Passed
2	’adm1’	’qwer’	Відмова	Passed
3	’adm1’	NULL	Відмова	Passed
4	’user3’	’qwer’	Відмова	Passed
5	’user3’	’sdfj’	Відмова	Passed
6	’user2’	’zxcv’	Відмова	Passed

3.6 Функціональне тестування

Проведено функціональне тестування, сценарій для якого створено на основі основних та альтернативних сценаріїв варіантів використання (табл. 3.10)

Таблиця 3.10 — Тестові випадки для варіантів використання

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
1	Перейти на сторінку реєстрації	Відкрилася сторінка реєстрації	Passed
2	Ввести логін test111 та двічі — пароль test111	Повідомлення про успішну реєстрацію	Passed
3	Ввести логін test222 та двічі — пароль test222	Повідомлення про успішну реєстрацію	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
4	Перейти на сторінку входу	Відкрилася сторінка входу	Passed
5	Ввести логін asdf та пароль asdf	Система відхилює авторизацію	Passed
6	Ввести логін test111 та пароль test111	Система відхилює авторизацію	Passed
7	Ввести логін asdf та пароль adsf	Відкрився головний інтерфейс системи (рядок меню)	Passed
8	Відкрити таблицю викладачів	З’явилася таблиця	Passed
9	Створити викладача («Ппшш» «Шррр» «Тссс» «+3820» «ba@b.c»), зберегти зміни та оновити список	Викладач присутній в списку та дані не пошкоджено	Passed
10	Змінити ім’я щойно створеного викладача на «Шссс», зберегти зміни та оновити список	Ім’я викладача змінилося, дані не пошкоджено	Passed
11	Відкрити таблицю курсів	З’явилася таблиця	Passed
12	Створити курс («» «Шррр» «Ппшш Ш. Т.» «29» «Йде набір») та зберегти зміни	Система вимагає ввести назву курсу	Passed
13	Створити курс («Ппшш» «Шррр» «Ппшш Ш. Т.» «Ппшш Ш. Т.» «29» «Йде набір»), зберегти зміни та оновити список	Курс присутній в списку та дані не пошкоджено	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
14	Змінити назву щойно створеного курсу на «Ппрр», зберегти зміни та оновити список	Назва курсу змінилася, дані не пошкоджено	Passed
15	Відкрити для щойно створеного курсу форму додання заняття	Відкрилася форма	Passed
16	Створити для щойно створеного курсу заняття («01.04.2016» «13:30» «Лекція»), зберегти зміни та оновити список	Заняття присутнє в списку та дані не пошкоджено	Passed
17	Змінити час щойно створеного заняття на «13:45», зберегти зміни та оновити список	Час змінився, дані не пошкоджено	Passed
18	Створити для щойно створеного курсу заняття («02.04.2016» «13:30» «Практика»), зберегти зміни та оновити список	Заняття присутнє в списку та дані не пошкоджено	Passed
19	Видалити заняття за дату 01.04.2016 та оновити список	Заняття відсутнє у списку	Passed
20	Встановити для заняття, що залишилося, тип «Лекція», зберегти зміни та оновити список	Тип змінився, дані не пошкоджено	Passed
21	Відкрити таблицю слухачів	З’явилася таблиця	Passed
22	Створити слухача («Шшпп» «Ршшш» «Сттт» «АЯ-313» «+3289» «db@d.b» «») зберегти зміни та оновити список	Слухач присутній в списку та дані не пошкоджено	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
23	Змінити ім’я щойно створеного слухача на «Ртдм», зберегти зміни та оновити список	Ім’я слухача змінилося, дані не пошкоджено	Passed
24	Розгорнути для щойно створеного слухача підтаблицю запису на курси	Розгорнулася підтаблиця	Passed
25	Записати слухача на курс «Ппрр»	Курс з’явився в підтаблиці	Passed
26	Перейти в таблицю «Курси»	Таблиця перекрила інші таблиці	Passed
27	Розгорнути для курсу «Ппрр» підтаблицю слухачів	Підтаблиця розгорнулася і містить слухача «Шшпп Р. С.»	Passed
28	Встановити слухачеві «Шшпп Р. С.» оплату «29», зберегти зміни та оновити список	Оплата 29 грн., боргу немає	Passed
29	Відкрити форму створення звіту за період	Відкрилася форма	Passed
30	Обрати дати 29.02.2016 та 28.02.2016 і створити звіт	Система не приймає дати	Passed
31	Обрати дати 29.02.2037 та 01.03.2037 і створити звіт	Система не приймає дати	Passed
32	Обрати дати 01.04.2016 та 02.04.2016 і створити звіт	Відкривається або завантажується PDF-файл, що містить дані, відповідні даним у таблицях інтерфейсу	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
33	Видалити заняття для курсу «Ппрр»	Курс «Ппрр» не містить занять	Passed
34	Видалити курс «Ппрр», зберегти зміни та оновити список	Курс відсутній у списку	Passed
35	Перейти в таблицю «Викладачі», видалити викладача Ппшш Шссс Тссс, зберегти зміни та оновити список	Викладач відсутній у списку	Passed
36	Створити у зовнішній системі три заявки, при цьому третю — на ПІБ «Шшпу Ртдм Стут», та зачекати 20 секунд	Лічильник сповіщень збільшився на 3	Passed
37	Відкрити сповіщення	Відображається панель зі сповіщеннями	Passed
38	Підтвердити реєстрацію облікового запису test111	Сповіщення зникло зі списку	Passed
39	Відхилити реєстрацію облікового запису test222	Сповіщення зникло зі списку	Passed
40	Підтвердити першу щойно додану заявку та перейти до таблиці «Слухачі»	Сповіщення зникло зі списку сповіщень, слухач із заявки з’явився в таблиці	Passed
41	Відхилити другу щойно додану заявку та оновити таблицю «Слухачі»	Сповіщення зникло зі списку сповіщень та не з’явилося в таблиці	Passed
42	Злити третю щойно додану заявку з запропонованим «Шшпп Ртдм Сттт», обравши ПІБ із заявки, та оновити таблицю «Слухачі»	Слухач відображається в таблиці з новим ПІБ	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
43	Перейти в таблицю «Слухачі», видалити слухача Шшпу Ртдм Стут, зберегти зміни та оновити список	Слухач відсутній у списку	Passed
44	Відкрити таблицю коефіцієнтів	З’явилася таблиця	Passed
45	Запам’ятати значення коефіцієнту зарплати персоналу, встановити його у 1.3, зберегти зміни та оновити таблицю	Коефіцієнт змінився, дані не пошкоджено	Passed
46	Відновити значення коефіцієнту зарплати персоналу, зберегти зміни та оновити таблицю	Коефіцієнт змінився, дані не пошкоджено	Passed
47	Вийти з системи	Відкрився екран входу	Passed
48	Увійти з логіном test111 та паролем test111	Відобразилося меню системи	Passed
49	Вийти з системи, увійти з логіном asdf та паролем adsf, відкрити таблицю користувачів	З’явилася таблиця	Passed
50	Скинути пароль для користувача test111, вийти з системи, увійти з логіном test111 та паролем test222	Відобразилося меню системи	Passed
51	Вийти з системи, увійти з логіном asdf та паролем adsf, відкрити таблицю користувачів, видалити користувача test111, зберегти зміни та оновити таблицю	Користувач test111 зник зі списку	Passed

Продовження таблиці 3.10

Test Case №	**Дія**	**Очікуваний результат**	**Результат тестування**
52	Вийти з системи, увійти з логіном test111 та паролем test222	Система відхилює авторизацію	Passed
53	Увійти з логіном test222 та паролем test222	Система відхилює авторизацію	Passed

4 РОЗГОРТАННЯ ПРОГРАМНОГО ПРОДУКТУ

4.1 Інструкція з встановлення

1. Встановити та налаштувати web-сервер Apache, інтерпретатор PHP5 не нижче версії 5.4, СКБД MySQL 5 або MariaDB 5. Для запуску сервера та клієнта на одній машині можна скористатися XAMPP з https://www.apachefriends.org/ru/index.html.

2. Скачати та розпакувати архів: https://github.com/bodqhrohro/knp2014/archive/master.zip.

3. Розпакувати вміст каталогу src з архіву до каталогу сторінок сайту. за необхідності налаштувати домен.

4. Встановити PHPMyAdmin (https://www.phpmyadmin.net/), перейти на вкладку «Імпорт» та вибрати файл deploy.sql з каталогу src. Впевнитися, що операція імпорту пройшла успішно.

5. Відкрити у текстовому редакторі файл config.php з кореневого каталогу сайту та вказати там IP чи домен СКБД, логін та пароль для доступу до БД, а також домен, на якому розміщено сайт. Зберегти файл.

6. Відкрити домен, на якому розміщено сайт, у браузері. Впевнитися, що відображається форма входу (рис 4.1).

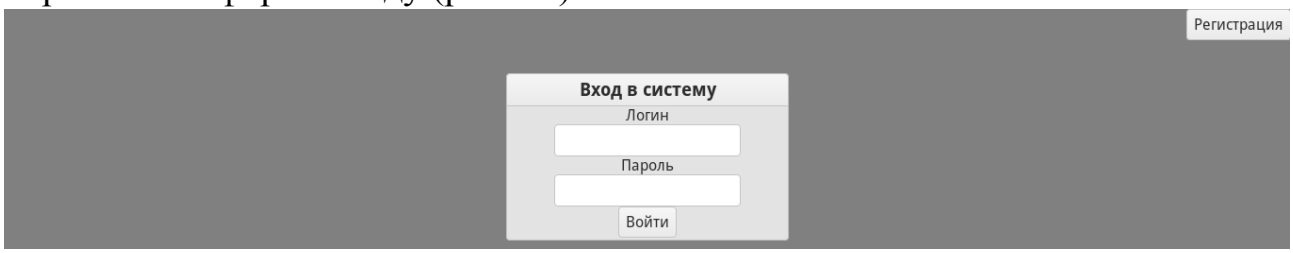

Рисунок 4.1 — Форма входу

4.2 Інструкція з використання

Демонстраційну копію системи розміщено на http://php-bodqhrohro.rhcloud.com/knp2014/.

В щойно встановленій системі є чотири тестових користувачі: адміністратор asdf, оператор fdsa, переглядачі pnd та qwer. Паролі для них, відповідно: adsf, fdsa, pnd, qwer. Можна реєструвати нових користувачів. Натисніть кнопку «Реєстрація» у верхньому правому кутку. введіть логін нового користувача та пароль. Підтвердіть форму, перейдіть знов на сторінку входу та спробуйте увійти під ко-

ристувачем asdf. Відкриється головний інтерфейс системи — стрічка меню (рис 4.2).

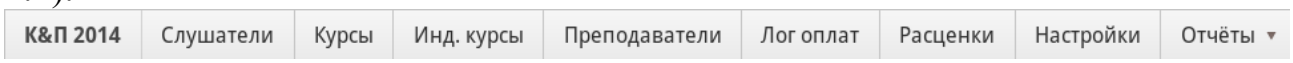

Рисунок 4.2 — Головне меню

При реєстрації нового користувача адміністратори отримують сповіщення (рис 4.3). Відкрийте з головного меню панель сповіщень та ввімкніть режим «Детально».

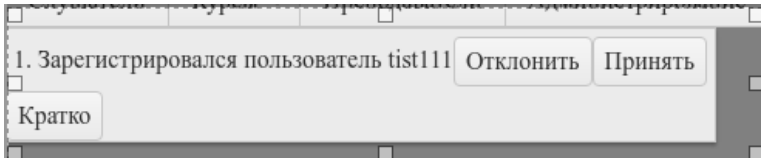

Рисунок 4.3 — Сповіщення

Можна підтвердити користувача, після чого під ним можна буде зайти, або видалити його. Таким же чином оброблюються заявки слухачів на запис на курси. Щоб сховати панель, натисніть пункт «Сповіщення» ще раз.

Пункт «Вихід» головного меню викликає завершення сесії користувача та перехід на екран входу. Підменю «Звіти» містить пункти, що відкривають форми налаштування звітів. Решта пунктів відкривають таблиці. Натисніть, приміром, пункт «Слухачі». Відкриється таблиця для роботи зі слухачами.

Можна додавати нові рядки за допомогою кнопки на панелі інструментів, видаляти їх за допомогою кнопки зправа та редагувати, клацнувши на потрібній комірці. Змінені комірки підсвічуються. Всі зміни в таблиці не синхронізуються із сервером автоматично — для цього слугує кнопка «Зберегти зміни». Якщо вміст якоїсь комірки перешкоджає збереженню, вона залишиться підсвіченою.

Таблиці «Слухачі» та «Курси» мають у кожному рядку підтаблиці, що дозволяють записувати слухачів на курси та працювати з оплатами. Розгортаються підтаблиці клацанням по трикутнику з лівого краю рядка. У підтаблицях не можна безпосередньо редагувати комірки, оплата вводиться у формі, що викликається кнопкою «Змінити». При поверненні грошей слухачеві вводиться від’ємна оплата. Додаються слухачі або групи з випадаючого списку; для слухачів він представлений у вигляді дерева з прапорцями, що дозволяє зручно додавати на курс

цілі університетські групи та шукати слухачів за групами замість довгого алфавітного списку чи форми пошуку. Слухачі не з університету або з невідомої групи відображаються у піддереві «Інші».

Коли треба згенерувати звіт, виберіть потрібний звіт з випадаючого меню. За необхідності введіть діапазон дат та натисніть кнопку «Створити звіт». Якщо звіт не відкривається у браузері, його можна зберегти та відкрити будь-яким переглядачем PDF та з нього ж відправити на друк.

ВИСНОВКИ

У даній дипломній роботі поставлено і вирішено завдання розробки системи обліку платних курсів із підтримкою самостійного запису слухачів через мережу Інтернет.

Експериментальним шляхом встановлено, що час, який витрачає асистент на реєстрацію слухача вручну, становить близько 5 хвилин. За використання системи реєстрація займає 4 хвилини, а за умови самостійного попереднього запису — менше хвилини. Таким чином скорочується черга слухачів під час набору слухачів на курс, а час асистентів звільняється для інших видів робочої діяльності.

В першому розділі «Визначення бізнес-вимог» дипломної роботи приведені вимоги бізнес-рівня, функціональні, не функціональні, середовища функціонування, кваліфікація користувачів, проведений аналіз існуючих аналогів, з виділенням переваг і недоліків.

В другому розділі «Проектування програмного продукту» описано проектування архітектури системи, структури та організації класів та бази даних.

В третьому розділі «Конструювання програмного продукту» представлені набір інструментальних засобів розробки та алгоритм програми, тестування функціональності системи та приклад її використання.

В четвертому розділі «Розгортання програмного продукту» наведено інструкції з встановлення та використання системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автоматизація документообігу (обліку) [Електронний ресурс]. — Режим доступу: URL: http://erp-project.com.ua/index.php/uk/korisni-materiali/statti/avtomatizatsiya/339-avtomatizatsiya-dokumentoobigu-obliku. — 1С ХАРКІВ ПРОЕКТ.

2. 1С: Предприятие 8 [Електронний ресурс]. — Режим доступу: URL: v8.1c.ru/. — 1С: Предприятие 8.

3. Платформа [Електронний ресурс]. — Режим доступу: URL: http://www.parus.com/products/system/platform/. — Корпорация ПАРУС.

4. Внутренний портал учебного заведения [Електронний ресурс]. — Режим доступу: URL: http://www.1c-bitrix.ru/solutions/edu/university/. — 1С-Битрикс.

5. Features [Електронний ресурс]. — Режим доступу: URL: https://docs.moodle.org/30/en/Features. — MoodleDocs.

6. Справка по Access - Служба поддержки Office [Електронний ресурс]. — Режим доступу: URL: https://support.office.com/ru-ru/article/Справка-по-Access-29d7b83c-3b06-41ca-b38b-483b6d5efb1b. — Справка и обучение по Microsoft Office — поддержка Office.

7. Система приоритетов MoSCoW - Product management [Електронний ресурс]. — Режим доступу: URL: http://aclebedev.ru/moscow-method/. — Product management - Наблюдения, мысли и опыт управления продуктом в IT.

8. Зв'язність (програмування) [Електронний ресурс]. — Режим доступу: URL: https://uk.wikipedia.org/wiki/Зв'язність_(програмування). — Вікіпедія.

9. Оценка программ [Електронний ресурс]. — Режим доступу: URL: http://www.structur.h1.ru/ocenka.htm. — Структуры и алгоритмы.

10. Бесплатные программы для просмотра PDF [Електронний ресурс]. — Режим доступу: URL: http://biblprog.org.ua/ru/pdf/. — BIBLPROG.

11. PHP Manual [Електронний ресурс]. — Режим доступу: URL: http://php.net/manual/en/index.php. — PHP: Hypertext Processor.

12. Kendo UI HTML Framework [Електронний ресурс]. — Режим доступу: URL: http://www.telerik.com/kendo-ui. — Telerik.

13. JSON: The Fat-Free Alternative to XML [Електронний ресурс]. — Режим доступу: URL: http://www.json.org/xml.html. — JSON.

ДОДАТОК А. ІЛЮСТРАЦІЇ

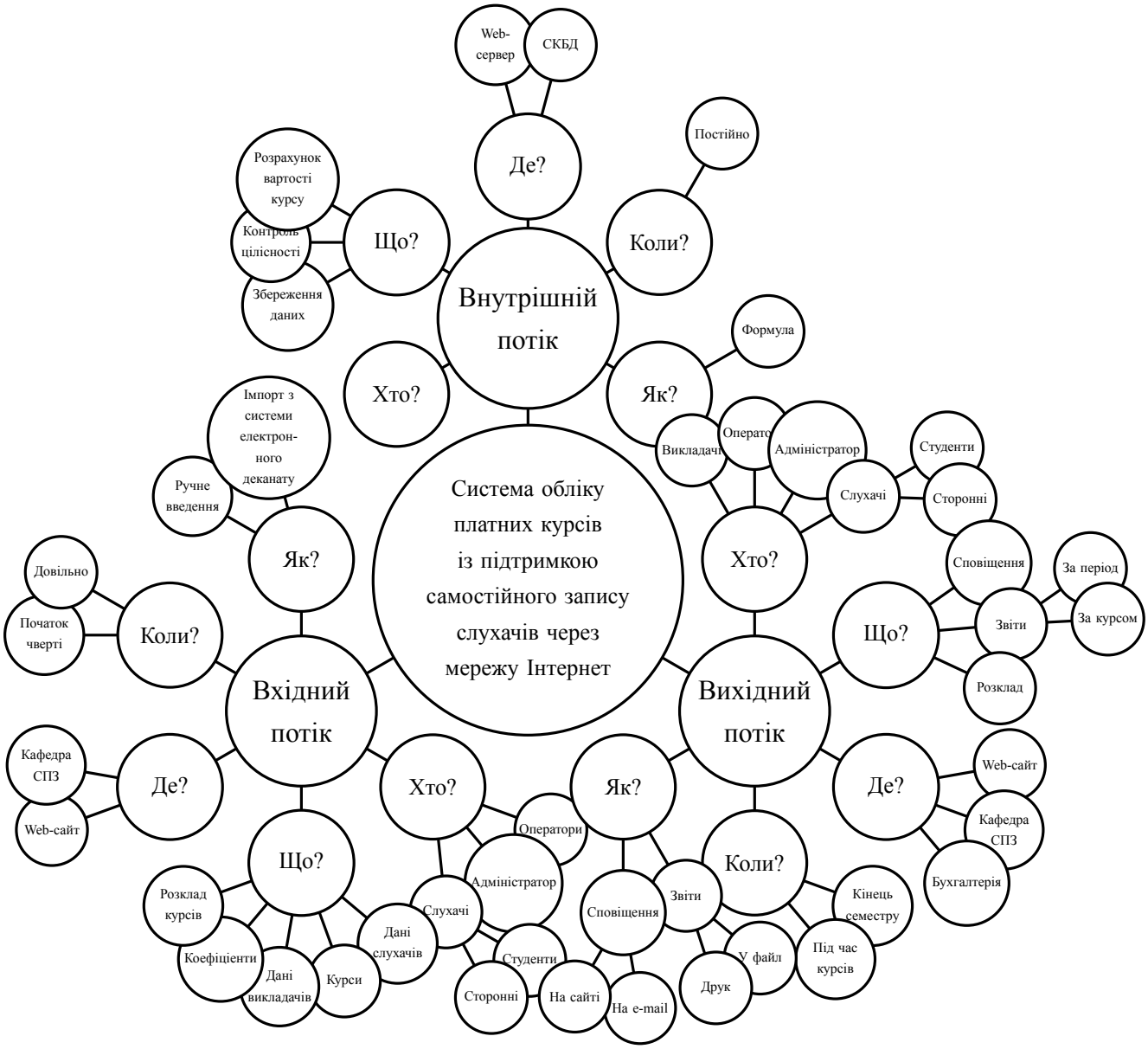

Рисунок А.1 — Мозкова карта

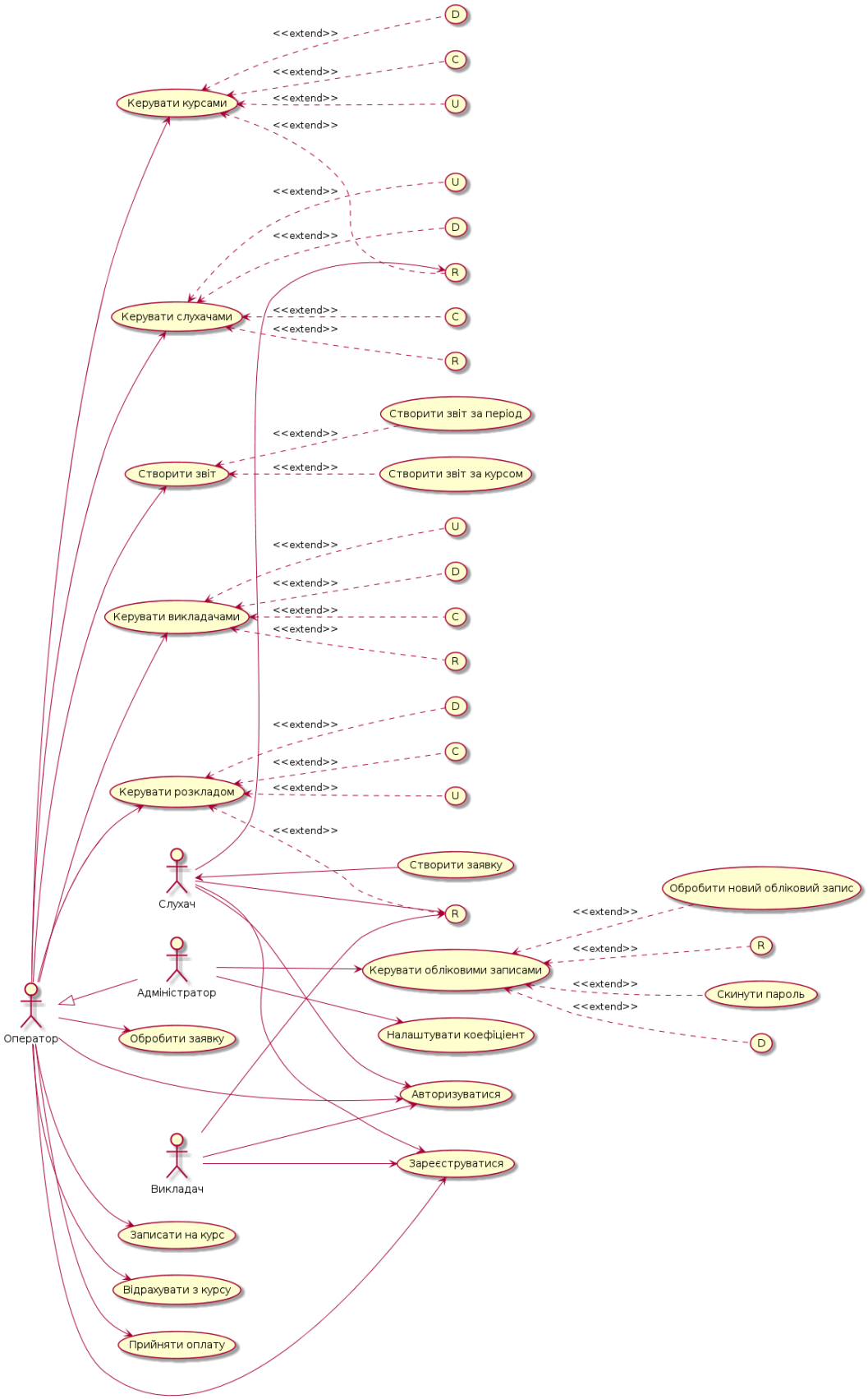

Рисунок А.2 — Діаграма варіантів використання

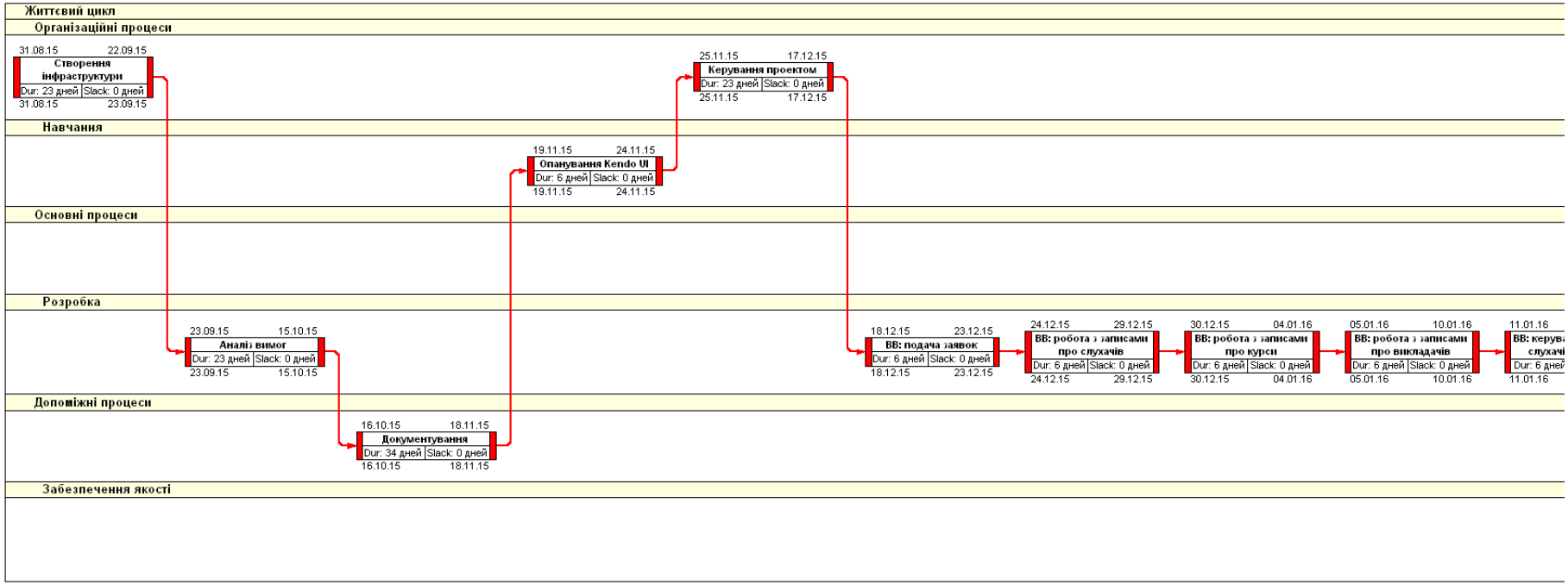

Рисунок А.3, аркуш 1 — Діаграма WBS

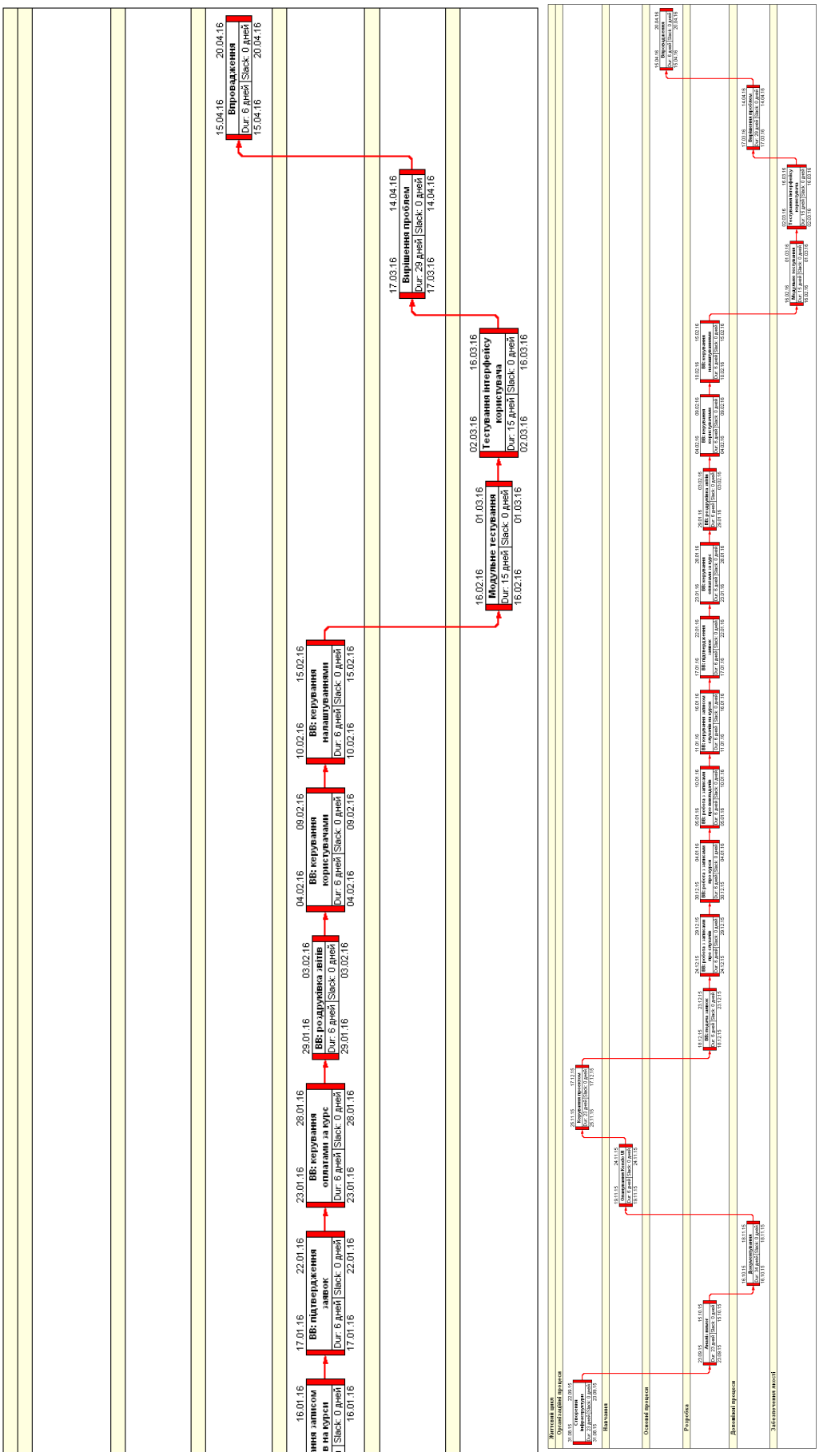

Рисунок А.4, аркуш 2 — Діаграма WBS

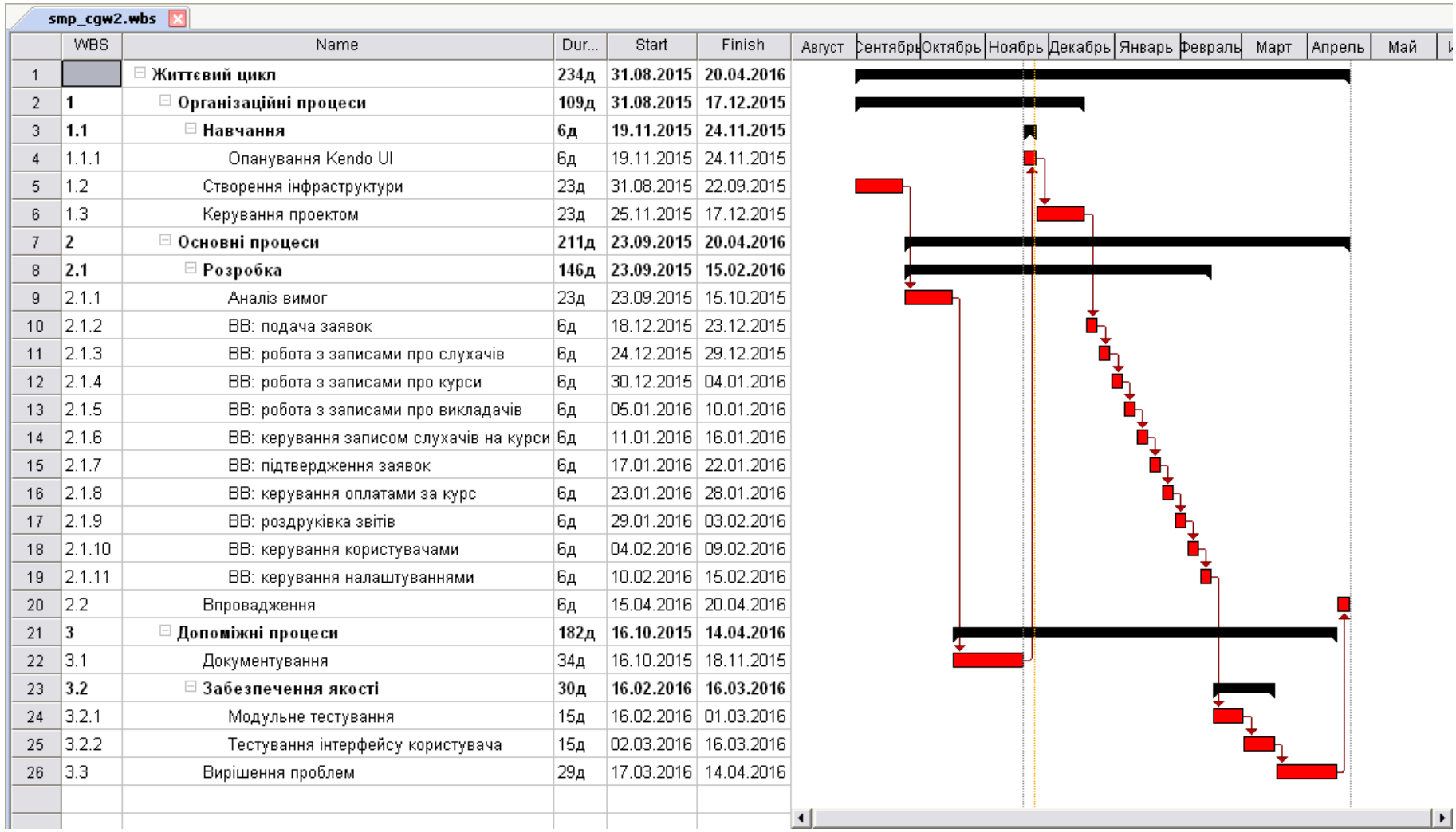

Рисунок А.5 — Діаграма Ганта

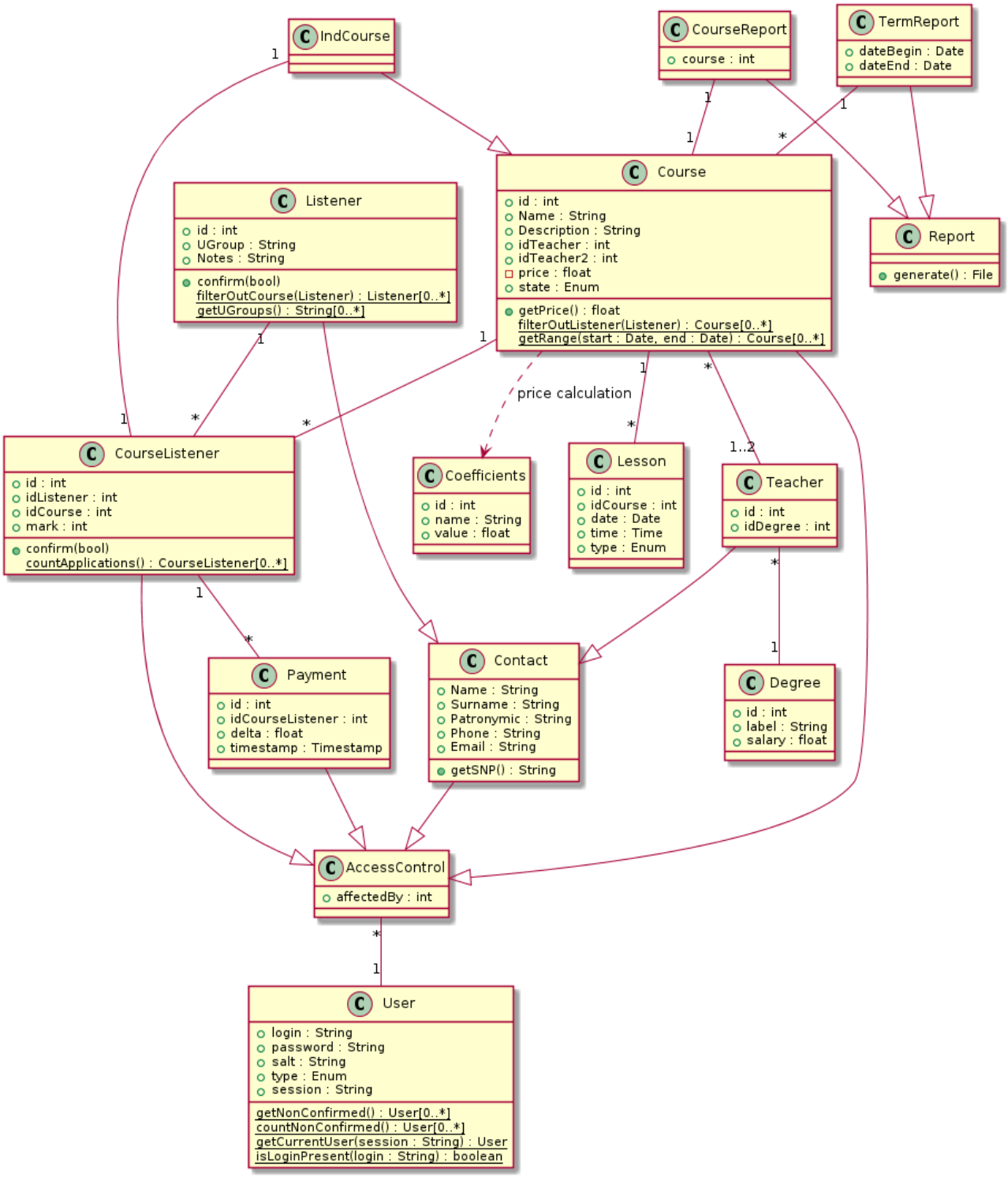

Рисунок А.6 — Діаграма програмних класів

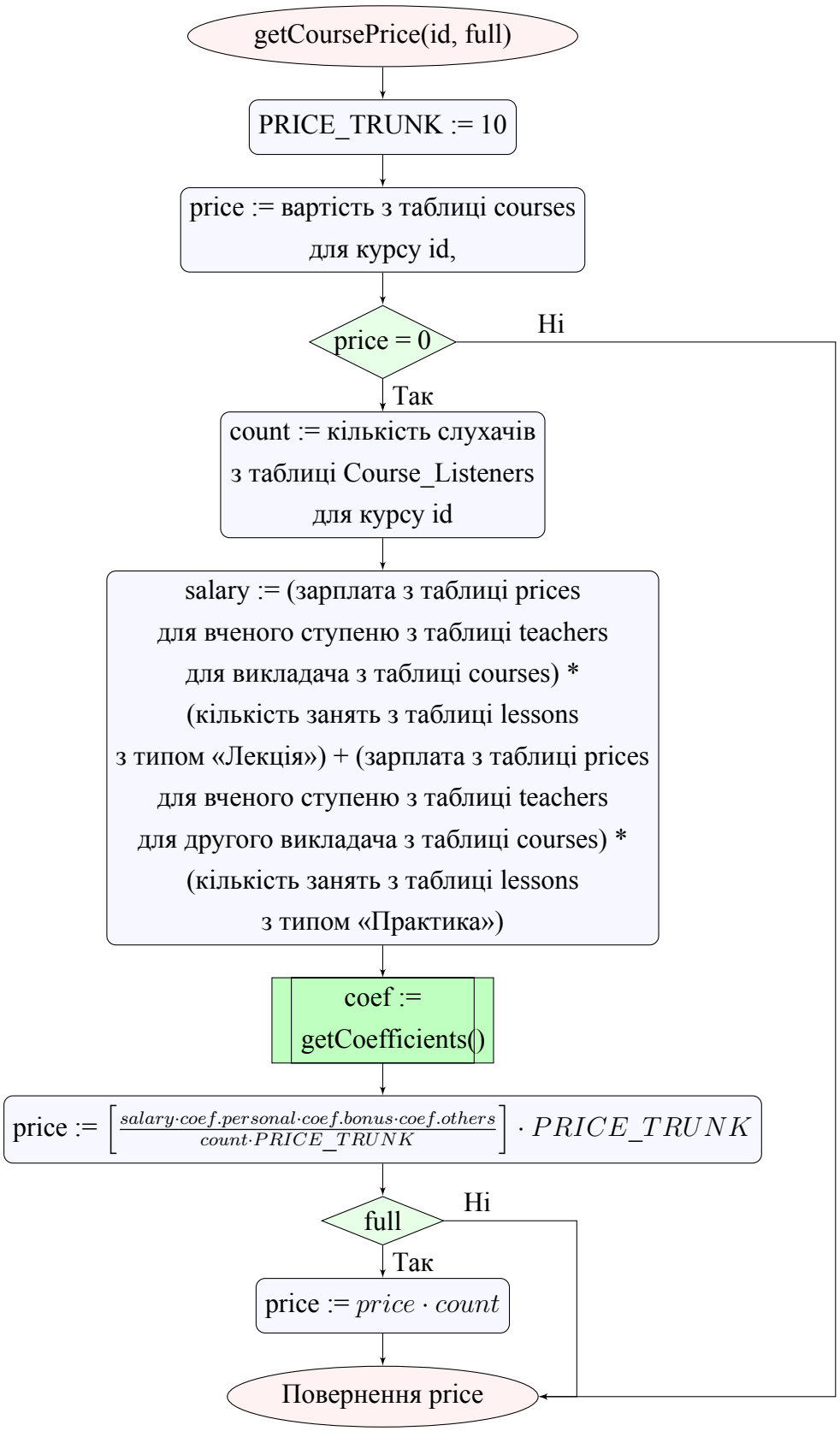

Рисунок А.7 — Алгоритм розрахунку вартості курсу

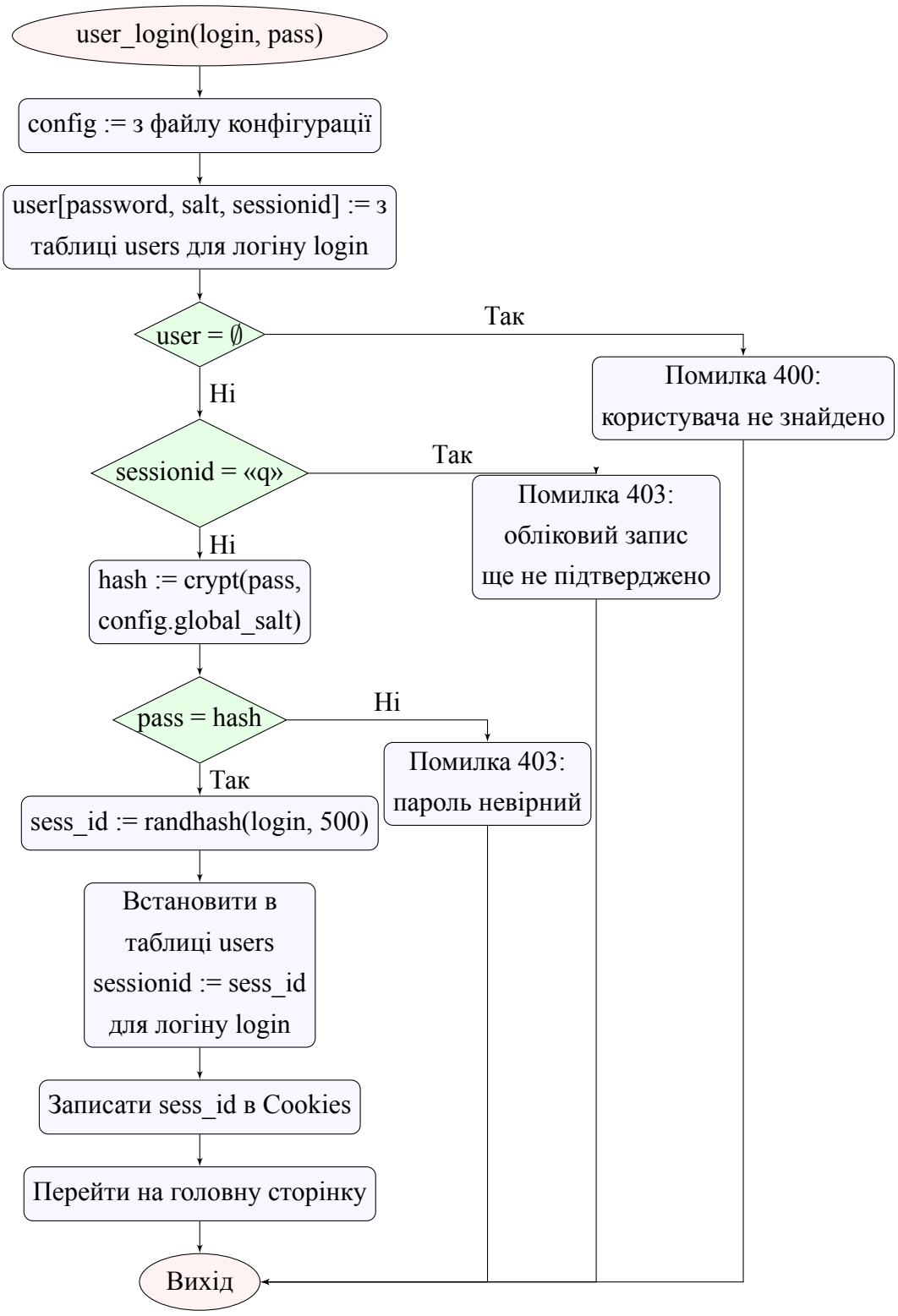

Рисунок А.8 — Алгоритм аутентифікації та авторизації

ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНОГО КОДУ

ajax.js:

```
//initializing taskbar
var courses2014={};
courses2014.winno=0;
courses2014.maxz=0;
courses2014.winlist={};
courses2014.course_states=[
 'Предложен',
 'Идёт набор',
 'Набор окончен',
 'Курс завершён'
];
courses2014.lesson_types=[
 'Лекция',
 'Практика'
];
var dsMapper = function(v,i) {
 return {i: i, v: v};
};
var verbose_notifications=0;
//initialize basic UI
$(document).ready(function(){
 $('#mainmenu').kendoMenu();
 var notparams={
  draggable: false,
  position: {
   top: 37,
   right: 0
  }
 };
 $('#notifications').kendoPopup(notparams).hide();
 getNotifications();
 setInterval(getNotifications,20000);
});
//get the window node containing given
function curWindow(elem){
 var curElem=elem;
 while (curElem){
  curElem=curElem.parentNode;
  if (curElem.id){
   if (curElem.id.indexOf('mainwin')+1){
    return curElem;
   }
  }
 }
 return false;
}
//switch to entitie's window
function openTable(entity){
 //checking if already opened
 var window=getWindow(entity);
 if (window){
  window.toFront();
  return;
 }
 //locale
 kendo.culture("ru-UA");
 //describing basic container
 var elem=$('<div
 id=\'courses2014_window_'+(++courses2014.
 //CALLBACK: closing the window
 var closeWindow=function(e){
  courses2014.winlist[entity]=false;
 };
 var winparams={
  draggable: false,
  position: {
   top: 37,
   left: 1
  },
  actions: [
   "Seek-n",
   "Seek-s",
   "Maximize",
   "Close"
  ],
  close: closeWindow
 };
 //window labels (USE ASSOCS, MZFK!!!)
 switch (entity){
  case 'course':
   winparams['title']='Курсы';
  break;
  case 'ind_course':
   winparams['title']='Индивидуальные
   курсы';
  break;
  case 'listener':
   winparams['title']='Слушатели';
```

```
 break;
 case 'teacher':
  winparams['title']='Преподаватели';
 break;
 case 'price':
  winparams['title']='Расценки';
 break;
 case 'settings':
  winparams['title']='Настройки';
 break;
 case 'user':
  winparams['title']='Пользователи';
 break;
};
//draw an empty window
elem.kendoWindow(winparams);

↪  courses2014.winlist[entity]=courses2014.winnum;
//CALLBACKS
var bindWMButtonsEvents=function(){
 $('.k-i-seek-n').click(tileWindowUp);
 $('.k-i-seek-s').click(tileWindowDown);
 $('.k-i-
↪  maximize').click(false).click(tileMaximize);
};
var tileWindowUp=function(e){
 var
↪  win=(($(e.currentTarget).closest('.k-
↪  window')).find('[role=dialog]')).data('kendoWindow');
 win.setOptions({
  height: (screen.availHeight-
↪  $('#mainmenu').outerHeight())/2-
↪  ($(e.currentTarget).closest('.k-
↪  window-titlebar')).outerHeight()
 });
};
var tileWindowDown=function(e){
 var
↪  win=(($(e.currentTarget).closest('.k-
↪  window')).find('[role=dialog]')).data('kendoWindow');
 var height=(screen.availHeight-
↪  $('#mainmenu').outerHeight())/2-
↪  ($(e.currentTarget).closest('.k-
↪  window-titlebar')).outerHeight();
 win.setOptions({
  height: height,
  position: {
   top: 200
   //top: screen.availHeight-height
  }
 });
};
var tileMaximize=function(e){
 var
↪  win=(($(e.currentTarget).closest('.k-
↪  window')).find('[role=dialog]')).data('ke
 win.setOptions({
  height: screen.availHeight-
↪  $('#mainmenu').outerHeight(),
  position: {
   top: 0
  }
 });
 return false;
};
bindWMButtonsEvents();
//1st is for View, 2nd is for Model
var response1;
//CALLBACK scheme -> datasource
var showResult1=function(response){
 //special controls definition
 switch (entity){
  case 'course':
   //teachers dropdown
↪  response['columns'][2]['editor']=getTeach

↪  response['columns'][3]['editor']=getTeach

↪  response['columns'][6]['editor']=getState
   //listeners rollover

↪  response['detailInit']=getListenersByCour
  break;
  case 'ind_course':
↪  response['columns'][2]['editor']=getTeach

↪  response['columns'][3]['editor']=getTeach

↪  response['columns'][6]['editor']=getState

↪  response['columns'][8]['editor']=listener
```

```
  break;
  case 'listener':
   //courses rollover
  response['detailInit']=getCoursesByListener;
  break;
  case 'teacher':
   //degrees dropdown
  response['columns'][5]['editor']=getDegreesList;
 }
 if (window.courses2014_editlock) {
  console.log(response['columns']);
 }
 //response['saveChanges']=function(e){
 response=commonSchemeDef(response,1);
 response1=response;
 $.ajax(request2);
};
//CALLBACK datasource -> render
var showResult2=function(response){
 var response2=response;
 response2=commonDataSourceDef(response2,entity);
 response1['dataSource']=new
 kendo.data.DataSource(response2);
 ($('<table/>').appendTo($(elem))).kendoGrid(response1);
 //bindHotkeys(elem);
};
var request1={
 url: 'entities/'+entity+'/scheme.json',
 dataType: 'json',
 success: showResult1
};
var request2={
 url:
 'entities/'+entity+'/dataSource.json',
 dataType: 'json',
 success: showResult2
};
$.ajax(request1);
}
//course selector
function
 AjaxPopupUserGroups(userid,elem){
 $(elem).html('<img
 src=\'style/Default/loading.gif\'>');
var renderResponse=function(response){
 $(elem).html(response);
};
var request1={
 url: 'ajax/returnGroupsByUser.php',
 type: 'POST',
 data: {
  userid
 },
 success: renderResponse
};
$.ajax(request1);
}
function getTeachersList(cont,opt){
 $('<input name="idTeacher"
  data-bind="value:' + opt.field +
  '"/>')
  .appendTo(cont)
  .kendoDropDownList({
    autoBind: false,
    dataSource: {
     transport: {
      read: {
       url: "ajax/returnTeachers.php",
       dataType: "json"
      }
     },
     dataTextField: 'TSNP',
     dataValueField: 'idTeacher',
     valuePrimitive: true
    });
}
/*function getListenersList(cont,opt){
 $('<input name="idListener"
  data-bind="value:' + opt.field +
  '"/>')
  .appendTo(cont)
  .kendoDropDownList({
    autoBind: false,
    dataSource: {
     transport: {
      read: {
       url: "ajax/returnListeners.php",
```

```
       dataType: "json"
      }
     }
    },
    dataTextField: 'LSNP',
    dataValueField: 'idListener'
   });
}*/
function getDegreesList(cont,opt){
 $('<input name="degree"
 ↪  data-bind="value:' + opt.field +
 ↪  '"/>')
  .appendTo(cont)
  .kendoDropDownList({
    autoBind: false,
    dataSource: {
     transport: {
      read: {
       url: "ajax/returnDegrees.php",
       dataType: "json"
      }
     }
    },
    dataTextField: 'deglab',
    dataValueField: 'degree'
   });
}
function getStateList(cont,opt){
 $('<input name="state"
 ↪  data-bind="value:' + opt.field +
 ↪  '"/>')
  .appendTo(cont)
  .kendoDropDownList({
    autoBind: false,
    dataSource: {
     data:
 ↪  courses2014.course_states.map(dsMapper)
    },
    dataTextField: 'v',
    dataValueField: 'i'
   });
}
function getLessonTypeList(cont, opt) {
 $('<input name="state"
 ↪  data-bind="value:' + opt.field +
 ↪  '"/>')
  .appendTo(cont)
  .kendoDropDownList({
    autoBind: false,
    dataSource: {
     data:
 ↪  courses2014.lesson_types.map(dsMapper)
    },
    dataTextField: 'v',
    dataValueField: 'i'
   });
}
function listenerPicker(cont,opt){
 var $container=$('<div/>').css('z-
 ↪  index',100000).attr('id','listenerPicker'
  draggable: false,
  anchor: cont,
  collision: 'fit',
  origin: 'top left'
 });
 //CALLBACK
 var initTabChooser=function(response){
  $container.html(response);
  var popup =
 ↪  $container.data('kendoPopup');
  var initTreeView=function(data) {
   var ds={};
   ds['data']=data;
   ds['scheme']={
    model: {
     id: "id",
     hasChildren: "num",
     data: "items"
    }
   };
   var tree =
 ↪  $container.find('#treeItemList').kendoTre
   dataSource: new
 ↪  kendo.data.HierarchicalDataSource(ds),
   select: function(e) {

 ↪  $input.val(tree.data('kendoTreeView').dat
    $input.trigger('change');
    popup.destroy();
    $container.remove();
   }
  });
```

```
 };
 var removeTabChooser=function() {
↪  $('#listenerPicker').remove(); };
↪  $.getJSON("ajax/returnListeners.php",initTabChooser);
 $('#modalButtons').hide();
 popup.open();
};
var request={
 url: 'lib/tabChooserTemplate.html',
 success: initTabChooser
};
$.ajax(request);
var $input = $('<input type="hidden"
↪  name="idListener"
↪  data-bind="value:' + opt.field +
↪  '"/>')
 .appendTo(cont);
}
function lessonsEditor(idCourse, name){
 var entity='lesson';
 var $container=$('<div/>')
  .css({'z-index':100001, 'max-width':
↪  '600px'})
  .attr('id','lessonsEditor')
  .appendTo('body')
  .kendoWindow({
   modal: true,
   title: 'Расписание курса «' + name +
↪  '»'
  });
 var response1;
 var showResult1=function(response){
↪  response['columns'][2]['editor']=getLessonTypeEditor;
  response1=response;
  $.ajax({
   url:
↪  'entities/'+entity+'/dataSource.json',
   dataType: 'json',
   success: showResult2
  });
 };
 var showResult2=function(response){
  var response2=response;
↪  response1=commonSchemeDef(response1,entity);
↪  response2=commonDataSourceDef(response2,e
↪  response2.schema.model.fields.idCourse.de
↪  response2['dataSource']=new
↪  kendo.data.DataSource(response2);
  $('<div/>').appendTo($container).css({
   width:'auto',
   float:'left'
  }).kendoGrid(response1);
↪  $container.data('kendoWindow').center();
 };
 $.ajax({
  url: 'entities/'+entity+'/scheme.json',
  dataType: 'json',
  success: showResult1
 });
}
function getCoursesByListener(e){
 var entity='courses_of_listener';
 var response1;
 var showResult1=function(response){
  response1=response;
 };
 var showResult2=function(response){
  var response2=response;
↪  response1=commonSchemeDef(response1,entit
↪  response2=commonDataSourceDef(response2,e
 response1['toolbar'][0]['template']='<a
↪  class=\'k-button k-button-icontext\'
↪  onclick=\'selectCoursesExListeners('+e.da
↪  type=\'javascript:void(0)\'>Записать
↪  на курс</a>';
↪  response1['columns'][5]['command'][0]['cl
 response1['dataSource']=new
↪  kendo.data.DataSource(response2);
↪  $('<table/>').appendTo(e.detailCell).kend
 };
 var setFullPaid=function(e){
  var spinner=$('<img
↪  src=\'style/Default/loading.gif\'>');
```

```
 $(e.target).prepend(spinner);
 var ids=this.dataItem($(e.target).closest('tr'));
 console.log(ids);
 var success=function(){
  spinner.hide();
 }
 $.ajax({
  url: 'entities/'+entity+'/update.php?full=1&id='+ids.idCL,
  type: 'POST',
  data: {
   payment: ids.price
  },
  success: success
 });
}
var request1={
 url: 'entities/'+entity+'/scheme.json',
 dataType: 'json',
 success: showResult1
};
var request2={
 url: 'entities/'+entity+'/dataSource.json',
 dataType: 'json',
 success: showResult2
};
$.ajax(request1);
$.ajax(request2);
}
function getListenersByCourse(e){
var entity='listeners_of_course';
var response1;
var showResult1=function(response){
 response1=response;
 $.ajax(request2);
};
var showResult2=function(response){
 var response2=response;
 response1=commonSchemeDef(response1,2);
 response2=commonDataSourceDef(response2,entity,e.data);
 response1['toolbar'][0]['template']='<a class=\'k-button k-button-icontext\' onclick=\'selectListenersExCourses('+e.da
 href=\'javascript:void(0)\'>Записать слушателя</a>';
 response1['columns'][5]['command'][0]['cl
 response1['dataSource']=new kendo.data.DataSource(response2);
 $('<table/>').appendTo(e.detailCell).kend
};
var setFullPaid=function(e){
 var spinner=$('<img src=\'style/Default/loading.gif\'>');
 $(e.target).prepend(spinner);
 var ids=this.dataItem($(e.target).closest('tr
 console.log(ids);
 var success=function(){
  spinner.hide();
 }
 $.ajax({
  url: 'entities/'+entity+'/update.php?full=1&id
  type: 'POST',
  data: {
   payment: ids.price
  },
  success: success
 });
}
var request1={
 url: 'entities/'+entity+'/scheme.json',
 dataType: 'json',
 success: showResult1
};
var request2={
 url: 'entities/'+entity+'/dataSource.json',
 dataType: 'json',
 success: showResult2
};
$.ajax(request1);
}
//CALLBACK
```

```
function pushParentTable(e){
 var grid=($($(e.sender)[0].list[0]).closest('.k-grid'));//.find('[role=grid]');//.data('kendoGrid');
 console.log(grid);
}
function callTabChooser(id,entity){
 var $container=$('<div/>').css('z-index',100000).attr('id','tabChooserPopup').appendTo(document.body);
  draggable: false,
  position: {
   top: 0,
   left: 0
  }
 });
 //CALLBACK
 var initTabChooser=function(response){
  $container.html(response);
  var initTreeView=function(data) {
   var ds={};
   ds['data']=data;
   ds['scheme']={
    model: {
     id: "id",
     hasChildren: "num",
     data: "items"
    }
   };
   ($container.find('#treeItemList')).kendoTreeView({
    checkboxes: {
     checkChildren: true
    },
    loadOnDemand: false,
    dataSource: new kendo.data.HierarchicalDataSource(ds)
   });
  };
  var removeTabChooser=function() { $('#tabChooserPopup').remove(); };
  var inList="";
  switch (entity) {
   case 'courses_of_listener':
    inList="ajax/returnCoursesExListeners.php";
    break;
   case 'listeners_of_course':
    inList="ajax/returnListenersExCourses.php";
    break;
  }
  $.getJSON(inList+"?id="+id,initTreeView);
  $container.find('#modalOKButton')).click(function()
  {
   var i,j,items,checked=[];
   var til=$('#treeItemList');
   var ds=til.data('kendoTreeView').dataSource.view();
   for (i=0;i<ds.length;i++) {
    items=ds[i].items;
    for (j=0;j<items.length;j++) {
     if (items[j].checked)
      checked.push(items[j].id);
    }
   }
   var request={
    url: 'entities/'+entity+'/create.php',
    type: 'POST',
    data: {
     ids: checked.join(':'),
     id: id
    },
    success: function(){
     refreshGrid(til);
     removeTabChooser();
    }
   };
   $.ajax(request);
  });
  ($container.find('#modalCancelButton')).click(removeTabChooser);
 };
 var request={
  url: 'lib/tabChooserTemplate.html',
  success: initTabChooser
 };
 $.ajax(request);
 $container.show();
 return $container;
```

```
}
function selectCoursesExListeners(userid,elem){
 var tcp=$('#tabChooserPopup');
 if (tcp.length) {
  tcp.remove();
 } else {
  $(elem).parent().append(callTabChooser('/users/unsubscribed_of_listener'));
 }
}
function selectListenersExCourses(courseid,elem){
 var tcp=$('#tabChooserPopup');
 if (tcp.length) {
  tcp.remove();
 } else {
  $(elem).parent().append(callTabChooser('/listeners/subscribed_of_course'));
 }
}
function subscribeListenerToCourse(elem,lid,cid){
 $.ajax({
  url: 'entities/courses_of_listener/create.php?listener='+lid+'&course='+cid,
  success: function(){
   ($(elem).closest('.k-window')).remove();
   $('.k-overlay').remove();
   var winno=courses2014.winlist[entity];
 if (winno){
  ($('#courses2014_window_'+winno).data('kendoWindow')).toFront();
  return;
 }
  }
 });
}
function recalcHeight(e){
 var rh=function(){
  var table=($(e.detailRow).closest('.k-grid'))
  table.css({height:table.height()+$(e.detailRow).height()});
 };
 setTimeout(rh,2000);
}function commonSchemeDef(response,type){
 //response['detailExpand']=recalcHeight;
 /*var refresh=function(e){
  ((elem.find('table[data-role=grid]')).data('kendoGrid')).refresh
 (); //unneeded?
 }*/
 response['pageable']={
  pageSize: 10,
  previousNext: false,
  messages: {
   display : "{0}-{1} из {2}"
  },
  refresh: true
 };
 response['resizable']=true;
 response['sortable']=true;
 response['scrollable']=true;//{'virtual':
 response['navigatable']=true;
 if (!response['editable'])
  response['editable']={ confirmation:
  true };
 response['toolbar']=[{name:"create",text:"Добавить"}];
 response['save']=tableSave;
 //response['toolbar'].push({template:
  '<a class=\'k-button
  k-button-icontext\' href=\'\\#\'
  onclick=\'refreshGrid(this);\'><span
  class=\'k-icon k-i-
  refresh\'></span>Обновить</a>'});
 return response;
}
function commonDataSourceDef(response,entity,data)
 if
  (response['schema']['model']['fields']['a
  response['schema']['model']['fields']['a
 response['transport']={};
 response['batch']=false;
 response['serverPaging']=false;
 response['pageSize']=10;
```

```
 response['transport']['read']={url:'entities/'+entity+'/read.php',dataType:'json',
 cache: false};
if (!window.courses2014_editlock) {
 if (entity!='user')
 response['transport']['create']={url:'entities/'+entity+'/create.php',dataType:'jso
 response['transport']['update']={url:'entities/'+entity+'/update.php',dataType:'jso
 if (entity!='settings')
 response['transport']['destroy']={url:'entities/'+entity+'/destroy.php',dataType:'j
}
if (entity=='courses_of_listener'||entity=='listeners_of_course')
{
 response['transport']['read']['url']+='?id='+data.id;
 if (!window.courses2014_editlock) {
 response['transport']['create']['url']+='?id='+data.id;
 response['transport']['update']['url']+='?id='+data.id;
 response['transport']['destroy']['url']+='?id='+data.id;
 }
}
if (entity=='lesson') {
 response['transport']['read']['url']+='?id='+data.id;
}
 return response;
}
function tableEdit(e){
 console.log(e.model);
 (e.container).addClass('edited_row');
}
function tableSave(e){
 $('edited_row').removeClass('edited_row');
}
function getWindow(entity){
 var winno=courses2014.winlist[entity];
 return
 winno?($('#courses2014_window_'+winno).data('kendoWindow')):false;
}
function refreshGrid(elem){
 var gd=$(elem).closest('[data-
 grid')).find('[data-
 role=grid]')).data('kendoGrid');
 gd.dataSource.read();
 gd.refresh();
}
function bindHotkeys(elem){
 elem=$(elem);
 elem.bind('keypress','Ctrl+r',function(){
 ($(elem.find('table[role=treegrid]')).dat
  return false;
 });
 /*elem.data("kendoWindow").wrapper.find('
 refresh").attr({'accesskey':'r'});
 elem.data("kendoWindow").wrapper.find(".k
 grid-add").attr({'accesskey':'n'});
 elem.data("kendoWindow").wrapper.find(".k
 changes").attr({'accesskey':'s'});
 elem.data("kendoWindow").wrapper.find(".k
 grid-cancel-
 changes").attr({'accesskey':'c'});*/
}
function showNotifications(){
 $('#notifications').toggle();
}
function getNotifications(){
 //CALLBACK
 var
 buttonWrapper1=function(type,params,confi
  return '<a class=\'k-button\'
 style=\'float:right\'
 onclick=\'callConfirm("'+type+'",'+JSON.s
 };
 $('#notification_count').html('<img
 src=\'style/Default/loading.gif\'>');
 var
 genNotificationList=function(response){
```

```
 $('#notification_count').html(response.length);
 var $container=$('#notifications');
 $container.html('');
 for (key in response) {
  $container.append($('<li class=\'notification\'/>').html(
   verbose_notifications?
   response[key]['string']+
   (response[key]['warning'] ? '<br><small>'+response[key]['warning']+'</small>'
   : '')+
   buttonWrapper1(response[key]['type'],response[key]['params'],1,'Принять')+
   (response[key]['params'] &&
   response[key]['params']['lid_old'] ?
   buttonWrapper1(response[key]['type'],response[key]['params'],2,'Слить')
   : '')+
   buttonWrapper1(response[key]['type'],response[key]['params'],0,'Отклонить')
   :response[key])
  );
 }
 $container.append(
  $('<a class=\'k-button\' onclick=\'verbose_notifications=!verbose_notifications;
  getNotifications();\' />')
  .text(verbose_notifications?'Кратко':'Подробно')
 );
}
var request={
 url: 'ajax/getNotifications.php'+(verbose_notifications?'?verbose=1':''),
 dataType: 'json',
 success: genNotificationList
};
$.ajax(request);
}
function transferHandler(e) {
 var data = this.dataItem($(e.target).closest("tr"));
 data.set('old', data.new);
}
function callConfirm(type,params,confirm,elem,postData){
 if (confirm == 2) {
  var $container=$('<div/>')
   .css({'z-index':100001, 'max-width': '600px'})
   .attr('id','mergeDialog')
   .appendTo('body')
   .kendoWindow({
    modal: true,
    title: 'Слияние слушателей'
   });

  $.getJSON('entities/listener/read.php?id=
  $.getJSON('entities/listener/read.php?id=
  ).then(function(old, app) {
   var grid=$('<div/>').appendTo($container).css({
    width:'auto',
    height:'auto'
   }).kendoGrid({
    editable: true,
    toolbar: [ {name: 'save', 'text': 'Сохранить и удалить заявку'} ],
    saveChanges: function(e) {
     callConfirm(type,params,confirm,elem,
     e.sender._data.reduce(function(prev, cur) {
      prev[cur.title] = cur.old;
      return prev;
     }, {}));
    },
    dataBound: function() {
     this.dataSource.view().forEach(function(e
     {
      if (entry.title == 'idListener' ||
     entry.title == 'affectedBy') {
       this.tbody.find('tr[data-uid="'+entry.uid+'"]').hide();
      }
     }.bind(this));
    },
    columns: [
     { title: 'Поле', field: 'title', editable: false },
```

```
    { title: 'Заявка', field: 'new',
↪   editable: false },
    { command: [
       { name: 'right', text: '',
↪   imageClass : 'k-icon no-text
↪   k-i-arrow-e', click:
↪   transferHandler }
      ],
      width: '56px'
    },
    { title: 'Слушатель', field: 'old' }
   ],
   dataSource:
↪   Object.keys(old[0]).map(function(key)
↪   {
    return {
     'title': key,
     'old': old[0][key],
     'new': app[0][key]
    };
   })
  });

↪   $container.data('kendoWindow').center();
  $('.no-text',
↪   $container).parent().css('min-
↪   width',
↪   0);
 });
 return;
}
var data={
 type: type,
 confirm: confirm
};
for (key in params)
 data[key]=params[key];
var removeNotification=function(){

↪   $('#mergeDialog').data('kendoWindow').close();
 $(elem).parent().slideUp();
};
var request={
 url:
↪   'ajax/confirm.php?'+Object.keys(data).map(function(key)
↪   { return key+'='+data[key];
↪   }).join('&'),
 type: postData?'POST':'GET',
 success: removeNotification,
 error: function(){errorBox(1);}
};
postData && (request.data = postData);
$.ajax(request);
}
function reportDialog(reportType){
 //CALLBACK
 var closeHandler=function(e){
  ($('#reportForm
↪   input[name=reportBeginDate]').data('kendo
  ($('#reportForm
↪   input[name=reportEndDate]').data('kendoDa
  (e.sender).destroy();
 }
 var $stub=$('<div/>');
 var winparams={
  position: {
   top: 37
  },
  actions: [
   'Close'
  ],
  title: 'Параметры отчёта',
  modal: true,
  close: closeHandler
 };

↪   $dialog=$stub.appendTo('body').kendoWindo
($stub.data('kendoWindow')).center();
 //CALLBACK
 var initForm=function(response){
  $dialog.html(response);
  if (reportType) {
   //$('#reportForm
↪   select[name=reportType]').append('<option
↪   value=\''+reportType+'\'/>');
   $('#reportForm
↪   select[name=reportType]').attr('value',reportTy
  }
  var year=(new Date().getFullYear());
  $('#reportForm
↪   input[name=reportBeginDate]').kendoDatePi
   value: (year-1)+'-09-01',
   format: '{0:yyyy-MM-dd}'
```

```
  });
  $('#reportForm
 ↪   input[name=reportEndDate]').kendoDatePicker({
    value: year+'-07-15',
    format: '{0:yyyy-MM-dd}'
  });
  //$('#reportForm
 ↪   select[name=reportType]').kendoDropDownList();
 }
 var request={
  url: 'lib/reportFormTemplate.html',
  success: initForm
 };
 $.ajax(request);
}
function errorBox(num){
 var s='';
 switch(num){
  case 1: s="Ошибка запроса"; break;
 }
 alert(s);
}
```