

Міністерство освіти і науки України  
Державний заклад  
«Луганський національний університет імені Тараса Шевченка»

Навчально-науковий інститут фізики, математики та інформаційних  
технологій

Кафедра інформаційних технологій та систем

**Старіков Владислав Володимирович**

**ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ  
ІНФОРМАЦІЙНИХ СИСТЕМ ТА РОЗРОБКА ДОВІДНИКА  
СПІВРОБІТНИКІВ НАВЧАЛЬНО-НАУКОВОГО ІНСТИТУТУ  
ФІЗИКИ, МАТЕМАТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НА  
ЗАСАДАХ WEB-ТЕХНОЛОГІЙ**

**кваліфікаційна робота**

**здобувача вищої освіти другого (магістерського) рівня  
освітньої програми «Мультимедійні системи»  
за спеціальністю 121 Інженерія програмного забезпечення**

Особистий підпис \_\_\_\_\_ Владислав СТАРІКОВ

Науковий керівник \_\_\_\_\_ Світлана ПЕРЕЯСЛАВСЬКА,  
кандидат педагогічних наук, доцент  
кафедри інформаційних технологій  
та систем

Завідувач кафедри \_\_\_\_\_ Микола СЕМЕНОВ,  
кандидат педагогічних наук, доцент  
кафедри інформаційних технологій  
та систем

Полтава – 2023

## АНОТАЦІЯ

**Старіков В.В.**

**Тема:** Дослідження технологій проектування інформаційних систем та розробка довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Спеціальність:** 121 «Інженерія програмного забезпечення».

**Установа:** ЛНУ імені Тараса Шевченка, 2023р.

**Магістерська робота містить:** 88 с., 15 рис., 9 табл., 37 джерел.

**Об'єкт дослідження** – Веб-технології проектування інформаційних систем.

**Предмет дослідження** - інформаційна система довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Мета роботи** – дослідження технологій проектування інформаційних систем та розробка довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Результати роботи** – в роботі розглянуто поняття та можливості інформаційних систем. Зокрема, розглянуто принципи створення інформаційних систем. Виявлено особливості веб-орієнтованих інформаційних систем. Представлено загальну характеристику веб-технологій, проаналізовано веб-стандарти і вимоги до програмних платформ і веб-застосунків. Аналізується загальна методологія створення ІС, зокрема методології RAD, RUP, UML.

Визначено дані, що формують потоки часто виконуваних запитів, які необхідні співробітникам директорату, кафедрам інституту, а також виділені регламентовані запити до бази. Обґрунтовано вибір СУБД MySQL архітектури клієнт-сервер для реалізації бази даних створеної інформаційної системи. За допомогою CASE-системи Power Designer розроблені концептуальна і фізична моделі. Розроблено довідник співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Ключові слова:** ІНТЕРНЕТ, ВСЕСВІТНЯ ПАВУТИНА, WORLD WIDE WEB, МОВА HTML, WEB-ТЕХНОЛОГІЇ, WEB-СИСТЕМИ, WEB-ОРІЄНТОВАНІ СИСТЕМИ, САЙТ, МОВА ПРОГРАМУВАННЯ, ТЕХНОЛОГІЯ, ІНФОРМАЦІЙНІ СИСТЕМИ, СКРИПТ, МОДУЛЬ.

## ANNOTATION

**Starikov Vladyslav**

**Theme:** Research of technologies for designing information systems and development of a handbook for employees of the Educational and Scientific Institute of Physics, Mathematics and Information Technologies on the basis of Web technologies.

**Speciality:** 121 "Software Engineering".

**Institution:** Luhansk Taras Shevchenko National University (LTSNU), 2023 year.

**Master's work of:** 88 p., 15 im, 37 sources.

**A research object of:** - Web technologies of designing information systems.

**The article of research-** information system of the directory of employees of the Educational and Scientific Institute of Physics, Mathematics and Information Technologies based on Web technologies.

**An aim of research is** - research on technologies for designing information systems and development of a handbook for employees of the Educational and Scientific Institute of Physics, Mathematics and Information Technologies on the basis of Web technologies.

**Job performanes** - the concept and possibilities of information systems are considered in the work. In particular, the principles of creating information systems are considered. The peculiarities of web-oriented information systems are revealed. The general characteristics of web technologies are presented, web standards and requirements for software platforms and web applications are analyzed. The general methodology of IS creation is analyzed, in particular the RAD, RUP, UML methodologies.

The data forming the streams of frequently executed requests, which are necessary for employees of the directorate, departments of the institute, as well as allocated regulated requests to the database, have been determined. The choice of the MySQL DBMS of the client-server architecture for the implementation of the database of the created information system is justified. Conceptual and physical models were developed using the Power Designer CASE system. A directory of employees of the Educational and Scientific Institute of Physics, Mathematics and Information Technologies was developed on the basis of Web technologies.

**Keywords:** INTERNET, WORLDWIDE WEB, WORLD WIDE WEB, HTML LANGUAGE, WEB TECHNOLOGIES, WEB SYSTEMS, WEB ORIENTED SYSTEMS, SITE, PROGRAMMING LANGUAGE, TECHNOLOGY, INFORMATION SYSTEMS, SCRIPT, MODULE.

## ЗМІСТ

<b>ВСТУП.....</b>	<b>6</b>
<b>РОЗДІЛ 1. ОСНОВНІ ПОНЯТТЯ ІНФОРМАЦІЙНИХ СИСТЕМ .....</b>	<b>11</b>
1.1 Поняття та можливості інформаційних систем .....	11
1.2. Принципи створення інформаційних систем.....	12
1.3. Особливості веб-орієнтованих інформаційних систем.....	17
<b>РОЗДІЛ 2 ТЕОРІЯ І ПРАКТИКА РОЗВИТКУ WEB-ТЕХНОЛОГІЙ.....</b>	<b>23</b>
2.1. Загальна характеристика веб-технологій .....	23
2.1.1. Основні тенденції розвитку веб-технологій .....	23
2.1.2. Веб-стандарти.....	27
2.2. Методологія побудови інформаційних систем на основі веб-технологій .....	35
2.3. Проектні рішення побудови інформаційних систем на основі веб-технологій .....	41
<b>РОЗДІЛ 3. РОЗРОБКА ДОВІДНИКА СПІВРОБІТНИКІВ НАВЧАЛЬНО-НАУКОВОГО ІНСТИТУТУ ФІЗИКИ, МАТЕМАТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НА ЗАСАДАХ WEB-ТЕХНОЛОГІЙ .....</b>	<b>54</b>
3.1. Структура та опис документообігу навчально-наукового інституту фізики, математики та інформаційних технологій .....	54
3.2. Основні об'єкти предметної області.....	57
3.3. Нормалізація даних.....	59
3.4. Розробка концептуальної моделі структури даних .....	60
3.5. Архітектура клієнт-сервер .....	63
3.6. Вибір та обґрунтування СУБД для реалізації бази даних .....	64
3.7. Обмеження цілісності даних.....	73
3.8. Опис створення бази даних.....	74
3.9. Опис створення програмного додатку на мові PHP .....	78
<b>ВИСНОВКИ .....</b>	<b>83</b>
<b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....</b>	<b>85</b>

<b>ДОДАТКИ.....</b>	<b>89</b>
Додаток 1. Концептуальна модель бази даних .....	89
Додаток 2. Фізична модель бази даних.....	90
Додаток 3. Лістинг файлу index.php.....	91
Додаток 4. Лістинг файлу layout.php.....	92
Додаток 5. Лістинг файлу templates/kafedras.php .....	93
Додаток 6. Лістинг файлу templates/listKafedra.php .....	94

## ВСТУП

Інформатизація процесів наукової та управлінської діяльності як технологічної компоненти наукових установ є важливим практичним завданням проведення адміністративної реформи і модернізації науки та освіти в Україні.

У Національній програмі інформатизації освіти і науки визначено стратегію розв'язання проблеми забезпечення інформаційних потреб та інформаційної підтримки діяльності в різних сферах загальнодержавного значення, зокрема в освіті та науці. Забезпечення інформатизації наукової та управлінської діяльності спрямовано на вирішення стратегічних завдань цієї програми щодо формування системи національних інформаційних ресурсів і створення загальнодержавної мережі інформаційного забезпечення науки та освіти. Це сприяє вирішенню актуальної для України проблеми формування сучасного інформаційного суспільства, що законодавчо визначається у Законі України.

Зміст інформатизації діяльності окремої організаційної структурної одиниці (підприємства, корпорації, установи, зокрема наукової) обумовлюється двома стратегічними завданнями: створення репрезентативного представництва у глобальному інформаційному середовищі та формування корпоративного інформаційного середовища, в якому суб'єкти цієї структурної одиниці провадять свою діяльність. Сучасний підхід до вирішення цих завдань полягає у створенні інформаційних систем двох типів: презентаційного порталу або сайту, доступного для всіх відвідувачів, та корпоративної інформаційної системи, що підтримує документообіг та основні бізнес-процеси діяльності організації. На сьогодні поширеною практикою в розробленні подібних систем є використання вебтехнологій, завдяки яким створюється зручне і звичне для користувача Інтернет-середовище у вигляді офіційного інтернет- та корпоративного інтранет-порталів, де питання комунікації та взаємодії ґрунтуються на загальновизнаних стандартах і протоколах.

Реалізація корпоративної інформаційної системи на основі веб-технологій для окремої предметної області потребує не тільки аналізу вимог і специфікації завдань, а й ґрунтовного аналізу, вибору таких засобів, методів, які доцільно застосовувати саме в цій предметній області.

Водночас інформатизація специфічних бізнес-процесів конкретної предметної області потребує розроблення власних методик застосування веб-технологій у створенні та впровадженні інформаційних систем. Звідси випливає необхідність розроблення науково обґрунтованої методологічної основи для побудови ефективної інформаційної системи з урахуванням особливостей предметної області.

Особливість предметної області, яка стосується наукових установ, характеризується видом їхньої основної діяльності – науковою діяльністю. Організація та зміст наукової діяльності регламентуються державними нормативними документами, зокрема Законом України «Про наукову та науково-технічну діяльність», а конкретизація організаційних форм, структури, послідовності процесів як у науковій, так і в управлінській діяльності установи визначається відомчою нормативною базою.

Довідник співробітників навчально-наукового інституту фізики, математики та інформаційних технологій дозволить підвищити ступінь поінформованості керівника та управлінського персоналу про хід та результати діяльності підприємства, а так само забезпечувати їх необхідними даними, плановими і аналітичними розрахунками, інформацією для прийняття ефективних оперативних управлінських рішень. При цьому підвищується обґрунтованість і об'єктивність прийнятих рішень за рахунок підвищення достовірності інформації, що надходить про стан справ на підприємстві.

Своєчасність отримання всієї необхідної інформації керівниками та працівниками всіх рівнів управління на підприємстві з єдиного інформаційного фонду дозволяє значно підвищити ефективність управління, забезпечити чітку узгодженість рішень, прийнятих на різних рівнях і в різних структурних підрозділах.

Поштовхом для створення програмного комплексу, призначеного організувати оперативне отримання інформації про співробітників навчально-наукового інституту фізики, математики та інформаційних технологій, послужила неможливість і складність подальшого управління підприємством «ручними» методами. Кількість працівників і нових спеціальностей підготовки постійно збільшується, тому що розширюється структура навчального закладу, що тягне за собою збільшення кількості кафедр, науково-педагогічного складу кафедр. Список виконуваних директором інституту завдань поповнюється новими найменуваннями. З плином часу змінюються умови їх вирішення. Всі ці фактори призвели до необхідності створення програмного комплексу.

Дані з бази можуть бути використані співробітниками, які займаються організацією навчально-виховного процесу в навчально-науковому інституті фізики, математики та інформаційних технологій, створенням навчально-методичного та матеріально-технічного забезпечення системи підготовки фахівців в ННІФМІТ, створенням умов для науково-дослідної роботи студентів і викладачів інституту, веденням обліку успішності студентів методистами та секретарями директорату. При плановій перевірці якості освіти також можуть бути використані дані, що надаються системою. Співробітник директорату повинен мати можливість отримати інформацію про викладацький склад кафедри, наукові звання, займані посади та ін.

**Об'єкт дослідження** – Веб-технології проектування інформаційних систем.

**Предмет дослідження** - інформаційна система довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Мета роботи** – дослідження технологій проектування інформаційних систем та розробка довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.



Для досягнення поставленої мети необхідно вирішити наступні завдання:

1. Розглянути поняття та можливості інформаційних систем, принципи створення інформаційних систем. Виявити особливості веб-орієнтованих інформаційних систем.
2. Провести аналіз технологій проектування інформаційних систем на засадах Веб-технологій.
3. Проаналізувати предметну область, документообіг та інформаційні потреби користувачів інформаційної системи навчально-наукового інституту фізики, математики та інформаційних технологій.
4. Розробити інфологічну та даталогічну моделі структури даних модуля довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій.
5. Розробити довідник співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

**Методи дослідження:** техніко-економічні із використанням комп'ютерних технологій, технічний аналіз, методи моделювання інформаційних процесів.

Практичною цінністю роботи є розроблений довідник співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

У першому розділі в результаті огляду літературних джерел проаналізовано поняття та можливості інформаційних систем. Розглянуто принципи створення інформаційних систем. Виявлено особливості веб-орієнтованих інформаційних систем.

В другому розділі представлено загальну характеристику веб-технологій, а саме: подано базові поняття, розглянуто основні тенденції розвитку, проаналізовано веб-стандарти і вимоги до програмних платформ і

веб-застосунків. Для розгляду особливостей побудови ІС на основі веб-технологій спершу аналізується загальна методологія створення ІС, зокрема методології RAD, RUP, UML. Далі визначаються проектні рішення щодо побудови ІС як інтернет/інтранет об'єкта.

У третьому розділі в процесі аналізу функціонування навчально-наукового інституту фізики, математики та інформаційних технологій були розглянуті предметна область, інформаційні потреби користувачів довідника співробітників. Визначено дані, що формують потоки часто виконуваних запитів, які необхідні співробітникам директорату, кафедрам інституту, а також виділені регламентовані запити до бази. Виконана нормалізація таблиць бази даних для адресної книги, спроектована інфологічна модель структури даних. Обгрунтовано вибір СУБД MySQL архітектури клієнт-сервер для реалізації бази даних створеної інформаційної системи. Проведено даталогічне проектування БД. Задано обмеження цілісності в СУБД MySQL на оновлення і видалення даних в зв'язаних таблицях бази даних. За допомогою CASE-системи Power Designer розроблені концептуальна і фізична моделі. Розглянуто розробка довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

## РОЗДІЛ 1. ОСНОВНІ ПОНЯТТЯ ІНФОРМАЦІЙНИХ СИСТЕМ

### 1.1 Поняття та можливості інформаційних систем

В інформатиці поняття система дуже поширено і має безліч значень, найчастіше воно використовується стосовно набору технічних засобів і програм. Додавання до поняття “система” слова “інформаційна” відбиває мету її створення і функціонування. Інформаційні системи забезпечують збір, збереження, обробку, пошук, видачу інформації, що необхідна в процесі прийняття рішень задач з будь-якої області.

Інформаційна система – це взаємозалежна сукупність засобів, методів і персоналу, що використовуються для збереження, обробки і видачі інформації в інтересах досягнення мети, що поставлена[1].

Процеси, що забезпечують роботу інформаційної системи будь-якого призначення, умовно можна представити у вигляді схеми, яку зображено на рисунку 1.1.



Рис. 1.1. Процеси в інформаційній системі

Схема складається з блоків:

- 1) введення інформації з зовнішніх чи внутрішніх джерел;
- 2) обробка вхідної інформації і представлення її в зручному для користувача вигляді;
- 3) введення інформації для представлення споживачам чи передачі в іншу систему;
- 4) зворотний зв'язок – це інформація, яка переробляється людьми даної організації для корекції вхідної інформації.

У вузькому тлумаченні під інформаційними розуміють системи, які призначені для збереження інформації в спеціальному чином організованій формі, що забезпечують її введення та різні маніпуляції з нею, у тому числі пошук по деяким ознакам, підготовку зведень чи звітів.

Сама ідея інформаційних систем і деякі принципи їх організації виникли задовго до появи персональних комп'ютерів. Бібліотеки, архіви, адресні бюро, телефонні довідники, словники – усе це інформаційні системи.

У сучасній інформатиці розглядаються такі інформаційні системи, що використовують для збереження та обробки інформації комп'ютерну техніку. Саме комп'ютеризація додала інформаційним системам їх сучасний вигляд, на кілька порядків підвищила ефективність і розширила сферу їх застосування.

З інформаційними системами працюють дві категорії фахівців: кінцеві користувачі, що керують, та фахівці з обробки даних.

Кінцевий користувач – це той, хто використовує систему чи інформацію, що вона випускає. Це визначення не поширюється на фахівців з автоматизованої обробки даних, що професійно аналізують, проектують і розробляють систему. По способах використання устаткування інформаційних систем розрізняють два типи кінцевих користувачів: безпосередній, котрий у ручному режимі прямо взаємодіє з устаткуванням системи, і непрямий, котрий лише використовує інформацію та одержує її від інформаційної системи, але, не контактує з її устаткуванням.

Процес проектування інформаційної системи має певну кількість етапів або кроків, послідовне виконання яких дозволяє розробити та впровадити в експлуатацію ефективну автоматизовану інформаційну систему, що буде виконувати всі поставлені задачі та відповідати вимогам користувачів до можливостей, порядку роботи та результатів роботи системи[2].

## **1.2. Принципи створення інформаційних систем**

У теорії та практиці створення інформаційних систем виділяють три підходи: локальний, глобальний та системний.

Суть локального підходу полягає в тому, що інформаційні системи створюють послідовним нарощуванням задач, які розв'язуються на персональному комп'ютері.

Він передбачає необмежений розвиток інформаційних систем, а тому кожна із них неможливо пізнати в цілому. Крім того, проект на предмет його повноти взагалі не розглядається та втрачається можливість науково обґрунтувати вибір і оцінити напрямки розвитку інформаційної системи, комплекс технічних засобів, а також побудувати її модель. До позитивних сторін цього підходу відносять: відносно швидку віддачу, наочність задач, можливість розробки невеликими “замкненими” групами, простоту керування створенням систем. Недоліками називають неможливість забезпечення раціональної організації комплексів задач, дублювання, постійну перебудову програм та організації задач, що призводить до дискредитації самої ідеї створення інформаційної системи.

При глобальному підході спочатку розробляють проект немовби повної, завершеної системи, а потім її впроваджують. Як правило, цей підхід призводить до морального старіння проекту ще до його впровадження, оскільки час його розробки може перевищувати період оновлення технічних, програмних та інших засобів, що використовуються в ньому.

Системний підхід до створення інформаційної системи – це комплексне вивчення економічного об'єкта як одного цілого з представленням частин його як цілеспрямованих систем і вивчення цих систем та взаємовідносин між ними. При системному підході об'єкт розглядається як сукупність взаємопов'язаних елементів однієї складної динамічної системи, яка перебуває в стані постійних змін під впливом багатьох внутрішніх і зовнішніх факторів, пов'язаних процесами перетворення вхідного набору ресурсів в інші вихідні ресурси.

Системний підхід має такі принципи:

- 1) кінцевої мети – абсолютний пріоритет кінцевої (глобальної) мети;

2) єдності – розгляд системи як цілого, так і сукупності частин (елементів);

3) зв'язності – розгляд будь-якої частини разом з її зв'язками з оточенням;

4) модульної побудови – корисно виділяти модуль в систему та розглядати її як сукупність модулів;

5) ієрархії – корисно вводити ієрархію частин (елементів) і (чи) їх ранжування;

6) функціональності – спільний розгляд структури і функцій з пріоритетом функцій над структурою;

7) розвитку – врахування змін системи, її здатності до розвитку, розширення, заміни частин, нагромадження інформації;

8) децентралізації – поєднання рішень, які приймаються, та керування централізацією і децентралізацією;

9) невизначеність – врахування невизначеностей та несподіванок у системі.

Характерними ознаками системного підходу є:

1) одночасне охоплення проектуванням великої кількості задач;

2) максимальна типізація та стандартизація рішень;

3) ключова роль баз даних;

4) локальне впровадження та збільшення функціональних задач;

5) багатоаспектне уявлення про структуру інформаційної системи як про систему, що складається з кількох класів компонентів, та відносна автономна їх розробка.

У методологічному відношенні системний підхід базується на ідеях цілісності, цілеспрямованості, організованості об'єктів, що вивчаються, їх внутрішній активності та динамізмі.

Про системність об'єктів свідчить те, що їх можна поділяти, оскільки лише вони мають структуру. Процеси декомпозиції і композиції є засобами отримання інформації для здійснення аналізу та синтезу систем.

Декомпозиція – це процес поділу систем на елементи, зручні для якихось операцій з нею, а саме розподіл до елементів, які приймаються за неподільні об'єкти[3].

Головна мета декомпозиції – розподіл системи на простіші частини. При зменшенні складності системи, ми забезпечуємо умови для аналізу та синтезу компонентів, для проектування, побудови, впровадження, експлуатації та вдосконалення систем управління.

Розподіл звичайно виконують у такий спосіб, щоб компоненти піддавались якій-небудь класифікації. Рекомендується зважати на природну декомпозицію, відбиту в існуючій структурі управління, обов'язках посадових осіб, документообігу, що діє, і тому подібне. Доцільно проводити багаторазову декомпозицію по кількох різних напрямках.

Загальна мета, критерії функціонування та основні обмеження на роботу системи звичайно формуються на початку створення системи.

Так, при декомпозиції можуть застосовуватись різні засоби, методи та ознаки поділу системи. Розподіл може мати матеріальну, функціональну, алгоритмічну та іншу основу. Однак сам процес декомпозиції кінцевий, оскільки розподіл відбувається до створення елементів, які приймаються за неподільні об'єкти.

Якість створення інформаційних систем визначається її ефективністю та надійністю.

Надійність інформаційної системи – це її властивість зберігати в часі в встановлених межах значення всіх параметрів, які характеризують здатність системи виконувати потрібні функції в заданих режимах і умовах експлуатації. Надійність інформаційної системи має властивості безвідмовності, ремонтпридатності, а інколи й довговічності.

Рівень надійності інформаційної системи залежить від таких факторів:

1) складу та рівня надійності технічних засобів, їх взаємодії та надійної структури;

- 2) складу та рівня надійності програмних засобів, їх можливостей і взаємозв'язку в структурі програмного забезпечення інформаційної системи;
- 3) раціонального розподілу задач, які розв'язуються системою, між технічними засобами, програмним забезпеченням і персоналом;
- 4) рівня кваліфікації персоналу, організації робіт і рівня надійності дій персоналу інформаційної системи;
- 5) режимів, параметрів і організаційних форм технічної експлуатації комплексу технічних засобів;
- 6) ступеня використання різних видів резервування (структурного, інформаційного, часового, алгоритмічного, функціонального);
- 7) ступеня використання методів і засобів технічної діагностики;
- 8) реальних умов функціонування інформаційної системи.

Ефективність інформаційної системи визначається порівнянням результатів від функціонування інформаційної системи і затрат усіх видів ресурсів, необхідних для її створення, функціонування та розвитку. До показників затрат ресурсів відносять матеріальні, людські, фінансові, часові та ін.

Ефективність інформаційної системи оцінюють у таких випадках:

- 1) при формуванні вимог, що висуваються до інформаційної системи;
- 2) при аналізі інформаційних систем, які створюються чи функціонують, на відповідність заданим критеріям;
- 3) при виборі найкращого варіанта створення, функціонування та розвитку інформаційної системи;
- 4) при синтезі найдоцільнішого варіанта побудови інформаційної системи за критерієм “ефективність-затрати”.

Засобами реалізації функціонування інформаційної системи є інформаційне, програмне та апаратне забезпечення:

- 1) інформаційне забезпечення містить у собі не лише інформаційні ресурси як предмет праці та інформацію як продукт праці, а й засоби і методи ведення усієї інформаційної бази — об'єкта



- управління. Отже, до інформаційного забезпечення належать також методи класифікації і кодування інформації, способи організації нормативно-довідкової інформації, побудови баз даних, зокрема побудови і ведення інформаційної бази і таке інше;
- 2) технічне забезпечення об'єднує сукупність усіх технічних засобів, використовуваних при функціонуванні комп'ютерної інформаційної системи;
  - 3) програмне забезпечення являє собою сукупність програм на носіях даних і програмних документів, які призначені для відлагодження, функціонування і перевірки працездатності системи.

### **1.3. Особливості веб-орієнтованих інформаційних систем**

Збереження і обробка інформації в інтернеті відбувається за допомогою так званих “веб-орієнтованих” інформаційних систем, які можуть і використовуватися в локальній мережі. Веб-орієнтовані ІС побудовані з використанням веб-додатків(Web-Application) – допоміжних програмних засобах, призначених для автоматизованого виконання будь-яких дій на веб-серверах і на стороні користувача. При цьому в якості користувацьких інтерфейсів веб-додатки використовують веб-браузери. До числа засобів створення веб-додатків відносяться клієнтські і серверні технології.

Процес створення веб-орієнтованих інформаційних систем нічим не відрізняється від написання програм. Стадії визначення вимог, аналізу, реалізації, проектування, тестування цей процес теж проходить. При роботі веб-сайту, неважно з використанням яких технологій він написаний, користувач так само як і у звичайній програмі, вводить дані і отримує інформацію, працює з вікнами і меню, зберігає дані на сервері і отримує відповіді. Але є одна різниця, яка полягає в тому, що саме програмне забезпечення працює не на комп'ютері користувача, а на віддаленому сервері мережі, а доступ до даних можна отримати із будь-якої точки світу де є кабельні

мережі чи телефонний зв'язок. З однієї сторони це зручно, а з іншої з'являються вимоги до програмного забезпечення, яке створюється.

Особливості веб-орієнтованих інформаційних систем:

- 1) надійність;
- 2) багатокористувацька робота;
- 3) проблема швидкодії;
- 4) незалежність від операційної системи клієнта;
- 5) знаходження на одному місці;
- 6) для користувача не потрібна ніяка програма;
- 7) користувач не являється адміністратором;
- 8) в ролі адміністратора розробник системи;
- 9) від користувача нічого не потрібно;
- 10) малий розмір;
- 11) переносимість;
- 12) простота;
- 13) архітектура веб-додатків не видна для користувача.

Надійність таких систем полягає у тому, що вони повинні працювати без збоїв, а не просто працювати. Неможна перегружати систему декілька раз на день, не можна сказати: «Вийдіть із системи тому, що необхідно перегрузити сервер», оскільки всі користувачі працюють віддалено в різних містах або навіть в різних континентах коли до системи є доступ через Інтернет.

В успішних веб-проектах на сайт приходять десятки користувачів на секунду. Тому варто замислитися над оптимізацією швидкості виконання програми.

Проблема швидкодії рішається просто в локальних мережах. Тут ті, хто розробляють саме програмне забезпечення не хвилюються скільки даних передає і приймає їхня програма. Коли мала пропускна спроможність лінії для невеликих систем з яким працюють декілька десятків користувачів прокладаються оптоволоконні кабелі, установлюється високошвидкісна мережа. Однак, коли в локальній мережі з цим ще можна миритись тому, що

об'єми даних, які передаються компенсуються швидкістю мережі, яка збільшується, то створеному по такому принципу веб-додатку працювати буде неможливо. Хоча швидкість модемів постійно зростає, і багато користувачів працюють з виділеною лінією Інтернет, але тут виникає питання оплати трафіка, при якому кожний зайвий мегабайт даних буде оплачуватися з кишені користувача.

Незалежність від операційної системи клієнта (кросплатформеність) означає, що веб-орієнтована ІС повинна працювати на всіх операційних системах.

Поведінка веб-орієнтованої ІС, як і будь-якого об'єкта в цьому світі, залежить від зовнішніх і внутрішніх умов.

До внутрішніх умов функціонування можна віднести такі фактори:

- 1) налаштування сервера;
- 2) тип сервера;
- 3) тип бази даних;
- 4) зміст перемінних середовища;
- 5) зміст інформації на жорсткому диску сервера;
- 6) зміст самої бази даних.

Також особливістю веб-орієнтованих ІС є знаходження всієї програмної логіки на сервері в порівнянні з простим ПЗ, де логіка знаходиться на комп'ютері кожного користувача. Так як є одна тільки логіка програмного забезпечення її набагато простіше розповсюджувати серед користувачів. Про старий спосіб розповсюдження програми можна забути. По суті проблеми розповсюдження веб-орієнтованої ІС не існує, адже доступ можна до неї можна отримати в будь-який момент в будь-якому місці, коли вона доступна через Інтернет чи Інтранет.

Для користувача не потрібна ніяка програма. Все що йому потрібно, це просто запустити браузер і набрати в адресній строчці URL. В наші дні браузер являється стандартною програмою, яку користувач отримує при встановленні

ОС. Шукати браузер йому не потрібно, адже він уже є встановлений на його машині, і по суті, це все, що йому потрібно для його роботи.

Користувач не являється адміністратором. Як правило, коли користувач встановлює на своїй машині додаток, то йому приходится брати на себе роль адміністратора цього додатку. Йому потрібно встановлювати його, запускати, чинити, рішати виникаючі проблеми. У випадку з веб-додатком, так як воно знаходиться на веб-сервері, користувачу не потрібно турбуватися про це. Йому і не потрібно переживати про це. Користувач буде тільки щасливий від цього. А це – в свою чергу є ознакою хорошого веб-додатку. У випадку з простим додатком про таке щастя і мріяти не доводиться.

В ролі адміністратора виступає розробник ІС. Так, це ще один вантаж на плечі програміста. Але коли порівнювати вартість створення веб-додатку з вартістю утримання команди спеціалістів, які займаються встановленням, підтримкою простих додатків на машинах користувачів, то зразу можна побачити, що буде дешевше, не говорячи уже про ефективність. З точки зору бізнесу, набагато вигідніше утримувати невелику команду програмістів, які працюють в одному місці над одним додатком.

Веб-орієнтована інформаційна система не потребує нічого від користувача. Веб-додатки, з яких побудована інформаційна система не пред'являють ніяких вимог до апаратної платформи.

Такої проблеми як підтримки різних версій в минулому тепер не існує. Як тільки виходить нова версія веб-додатку, то всі без виключень користувачі отримують її негайно. Так існує тільки одна копія додатку(додатків) на білому світі, то всі старі версії негайно зникають, а користувач навіть не помічає, що у нього нова версія програми. Це також означає, що розробникам не потрібно турбуватися про підтримку старих версій програм і про підтримку зворотної сумісності.

Користувачу не потрібно завантажувати на свій комп'ютер весь веб-додаток повністю, щоб почати з ним працювати. Навіть весь інтерфейс не обов'язково загрузати. Достатньо завантажити ту його частину, яка необхідна

для виконання конкретної задачі. Завдяки цьому веб-додатки невеликі по об'єму, швидко загружаються і швидко відповідають на дії користувачів

Так як на комп'ютері користувача нічого не встановлюється, то користувач може працювати з додатком з любого місця. «Любе місце» буквально означає любе місце на Землі. Загрузити веб-додаток можна, валяючись дома на дивані, сидячи в офісі, знаходячись на Гавайях, і у всіх випадках воно буде працювати без збоїв.

Як показує практика, веб-рішення все частіше інтегруються в інформаційну інфраструктуру підприємства, стають його невід'ємною частиною. Принципи швидкого доступу до інтернету, які добре зарекомендували себе в інтернеті прекрасно працюють і для інтранет-систем. Windows-додатки, які володіють меншою гнучкістю і більшою ресурсомісткістю все частіше уступають місце в локальній мережі веб-додаткам, які надають не статичні сторінки HTML, а динамічні, тобто ті, які керуються користувачами, звіти професійної якості. Швидка публікація інформації на внутрішньому сайті компанії і здобуття інформації з внутрішньої бази даних, доступ до всіх ресурсів за допомогою звичайного web-браузера, легке нарощування можливостей — все це робить web-додатка чудовим інструментом для роботи з інформацією. Від інтранет систем можливий перехід до екстранет-систем, забезпечуючий доступ до інформаційної структури підприємства віддалених офісів, складів, магазинів, мобільних користувачів і реалізації видів комерційної діяльності. Веб-системи – гнучкі (навіть сильно), адже кожна сторінка, яка передається клієнту, динамічно створюється на сервері у відповідності до конкретного запиту. Що передати, і як це оформити рішає веб-додаток. Веб-системи генерують користувацьких інтерфейс «в польоті». Керівникам ІТ-служб треба чітко уявляти собі, для чого використовуватиметься система, що розробляється, які проблеми вона покликана вирішувати, а також обґрунтувати її необхідність. Лише при такому підході можна сказати, що web-додаток буде дійсно робочим інструментом, а не пам'ятником втраченим

інвестиціям. До того ж швидкий розвиток інтранет-систем в даний час стримується недостатньою пропозицією їх з боку розробників. Більш ніж десятирічна історія windows-додатків і мільйони доларів інвестицій в них стримують компанії — виробники ПО від швидкого переходу до web-технологій. Вони повинні доповнювати вже наявні застосування, розширюючи кордони їх вживання. Лише тоді можна повною мірою оцінити переваги від спільної роботи Windows і web-систем в локальній мережі.

## **РОЗДІЛ 2 ТЕОРІЯ І ПРАКТИКА РОЗВИТКУ ВЕБ-ТЕХНОЛОГІЙ**

### **2.1. Загальна характеристика веб-технологій**

**Поняття веб-технології.** Це поняття є похідним від синонімічного ряду понять Web, World Wide Web, Всесвітня павутина, веб, що визначають глобальний інформаційний простір, заснований на фізичній інфраструктурі Інтернету і протоколі передавання даних HTTP. Вебтехнологія – це сукупність методів і програмно-технічних засобів, інтегрованих з метою ефективного опрацювання веб-ресурсів, які містяться у веб-просторі. Вузьке трактування поняття веб-технологій пов'язано з методами і засобами створення веб-сторінок із підтримкою мультимедіа, що поєднують у собі різні види інформації: текст, графіку, звук, анімацію та відео. Таке трактування веб-технологій охоплює базові сервіси Інтернету і не втрачає свого змісту і сьогодні. Проте в сучасних умовах, коли Інтернет є не тільки мультимедійною картинкою з текстом у веб-просторі, але потужним засобом комунікації, інтеграції, пошуку веб-ресурсів, надання різноманітних сервісів, поняття веб-технологій трактується ширше – як комплекс технічних, комунікаційних, програмних методів розв'язання завдань організації спільної діяльності користувачів із застосування мережі Інтернет [13]. У контексті завдань інформатизації менеджменту наукової діяльності поняття веб-технології будемо інтерпретувати як комплекс технічних, комунікаційних, програмних методів створення ІС «Наукові дослідження» та комплексного Інтернет-представництва НАПН України у мережі Інтернет, що визначило порядок викладу і зміст цього розділу монографії. За результатами аналізу понятійно-категоріального апарату вебтехнологій сформовано глосарій «Базові поняття і терміни вебтехнологій» [39], в якому наведено перелік і визначення 390 базових термінів інформаційно-комунікаційних і веб-технологій, пов'язаних з такими суттєвими поняттями дослідження, як інформаційна система, сайт, портал і т. д.

#### **2.1.1. Основні тенденції розвитку веб-технологій**

Розглянемо основні етапи розвитку поняття Веб.

**Веб 0.** Діяв на початку 90-х років. Вперше використовує HTML-сторінку. Контент був переважно текстовим документом із графічними ілюстраціями та посиланнями на інші ресурси, а Інтернетом користувалися тільки наукові установи, технологічно розвинені корпорації та комерційні компанії.

**Веб 1.0.** На наступному етапі розвитку веб-технологій Інтернет став тим місцем, де з'явилося все: зображення, музика, відео, тексти, будь-яка інша інформація. Аудиторія Інтернету дуже розширилась: доступ до глобальної мережі отримали майже всі власники персональних комп'ютерів. Веб 1.0 – це ретронім поняття, яке стосується статусу WWW та будь-якого стилю дизайну веб-сайту, що використовувався перед появою терміна Веб 2.0. Змістом поняття Веб 1.0 вважається «той Веб, який був до Веб 2.0», який відображає звичайну практику порівняння сайтів за типом технології, що використовується. Різниця між Веб 1.0 і Веб 2.0, як зазначено в [17], може бути сформульована таким чином: розвиток від персональних сторінок до блогів та блог-агрегаторів, від простої публікації матеріалів до участі та обговорення, від контенту сайту як результату великих інвестицій до інтерактивного процесу накопичення інформації, від систем управління контентом (CMS) до систем, заснованих на посилальних тегах (folksonomy). Автори вважали, що зазначені вище чинники якраз і сформували основні зміни в тенденціях, які привели до повномасштабного переходу на Веб 2.0. Переваги Веб 2.0 над Веб 1.0 особливо помітні у технологічних засобах, що адаптували такі інтернет-технології, як широкопasmугова мережа, вдосконалені браузері, Аїах, збільшення платформ застосунків Flash і масовому розвитку віджетів (таких, як Flickr та значки YouTube). Крім технологічного розвитку Інтернету, перехід від Веб 1.0 до Веб 2.0 є прямим результатом змін у поведінці тих, хто використовує Всесвітню павутину. Основні тенденції Веб 1.0 стосувалися проблеми безпеки і приватності в односторонньому потоку інформації через вебсайти, що містять матеріали «тільки для читання». Характерне тло Веб 1.0 вирізнялося комп'ютерною неграмотністю широких мас, поширеністю повільних типів підключення до Інтернету, обмеженістю Інтернету. За Веб 2.0



використання мережі постає як децентралізація змісту веб-сайту, що, крім традиційних споживачів, відвідується багатьма користувачами, які беруть участь у процесі додавання і корекції інформації.

**Веб 2.0.** Початком етапу розвитку веб-технологій Веб 2.0 вважають 2005 р., коли поняття Веб 2.0 було розкрито в публікації Tim O'Reilly – «What Is Web 2.0» в журналі «Компьютерра» [25]. В епоху Веб 2.0 ресурсомісткі та складні для самостійного створення домашні сторінки заступили блоги, які створюються за декілька хвилин. Замість великих порталів з'явилась Wiki. Саме ці технології забезпечили активну діяльність користувача в Інтернеті, що не вимагає спеціальних знань і вмінь, а її процеси є достатньо простими. Головною особливістю Веб 2.0 є покращення та пришвидшення взаємодії веб-сайтів останніх, що привело до стрімкого зростання активності користувачів. Це проявилось в участі в Інтернет-спільнотах (зокрема, в форумах), розміщенні коментарів на сайтах, веденні персональних журналів (блогів), розміщенні посилань у WWW. Веб 2.0 зафіксував перехід WWW від одиничних дорогих комплексних рішень до сильно типізованих, дешевих, легких у використанні сайтів з можливістю ефективного обміну інформацією. Принциповою відмінністю технології Web 2.0 від технологій Web 1.0 є те, що її використання дає змогу не лише переглядати вебресурси мережі, а й завантажувати власні, здійснювати обмін цими ресурсами з іншими користувачами, діяти спільно з метою їх накопичення, брати участь в обговореннях та ін. Сьогодні термін Web 2.0 означає не стільки сукупність певних конкретних технологій, скільки філософію представлення інформації у веб-орієнтованому середовищі та побудову інформаційних відносин.

**Web 3.0.** У результаті розвитку технології Веб 2.0 сформувалася технологія Веб 3.0. Web 3.0 – це високоякісний контент і сервіси, що створюються талановитими професіоналами на технологічній платформі Web 2.0. Головна ідея Web 3.0 полягає в тому, що користувач, який до цього одноосібно був залучений до процесу формування контенту, відтепер творить колективно і його партнерами, крім інших користувачів, є експерти напрямів.

Web 3.0 – концепція розвитку інтернет-технологій, сформульована керівником Netscape.com Джейсоном Калаканісом (англ. Jason Calacanis) [14] у продовження концепції Web 2.0 Тіма О'Рейлі. Її суть у тому, що Web 2.0 є тільки технологічною платформою, а Web 3.0 дозволить на її основі силами професіоналів створити високоякісний контент і сервіси.

**Визначення Web 3.0** було опубліковано в особистому блозі Калаканіса 10 березня 2007 р. Калаканіс зазначив, що Web 2.0 дозволяє швидко і практично безкоштовно використовувати значну кількість потужних інтернет-сервісів з високими споживчими якостями, що призвело до появи величезної кількості одноманітних ресурсів і, як наслідок, девальвації цінності більшості з них. Ідея, що на основі Web 2.0 повинна виникнути нова платформа, є не стільки технологічною, скільки соціокультурною, і використовується професіоналами для створення цікавого, корисного і якісного контенту. Як приклад тенденції до переходу від Web 2.0 до Web 3.0 Калаканіс наводить німецький розділ Вікіпедії, який мірою наповнення контентом вдається до закриття на редагування недосвідченими учасниками якісних статей, вводить рецензування статей силами професійних редакторів. Термін Web 3.0 співвідносять із поняттям семантичної павутини. Концепція семантичної павутини ґрунтується на метамові опису змісту сайтів для організації автоматичного обміну між серверами. Автор терміна «Web 2.0» Тім О'Рейлі запропонував визначити Web 3.0 як «взаємодію Інтернету з фізичним світом», він також неодноразово виступав із критикою ототожнення семантичної павутини і концепції Web 3.0 [14]. У цілому Web 3.0 можна охарактеризувати як високоякісний контент і сервіси, які створюються талановитими професіоналами на технологічній платформі Web 2.0.

**Web 4.0** (узагальнене визначення) – високоякісний контент і сервіси, як створюються звичайними користувачами на технологічній платформі Web 3.0. Використання Web 4.0 дозволяє створювати контент служб і сервісу для спілкування, роботи, навчання, відпочинку тощо настільки доступними, що кожний користувач Інтернету здатен створити власний майданчик в Інтернеті,

де буде реалізуватися сам і разом зі своїм соціумом – з друзями, знайомими, колегами та однодумцями. Наразі відбувається оновлення можливостей Web 3.0 до можливостей Web 4.0.

### **2.1.2. Веб-стандарти**

Розробленню веб-стандартів передував випуск специфікації TCP/IP у 1981 р. [7], а в 1982 р. його використано для з'єднання розрізнених мереж, що стало початком ери Інтернету. Ця подія спричинила появу нового типу програм – браузерів, призначення яких полягало в забезпеченні перегляду інформації в Інтернеті. Оскільки на той момент не існувало стандартів, яким такі програми повинні були відповідати, то браузери від різних виробників накладали певні обмеження на сайти, тобто коректно сайт відображався тільки у певному браузері. З метою подолання цього явища, відомого під назвою «війна браузерів», у 1994 р. Тім Бернерс-Ліколі заснував у Массачусетському технологічному інституті Консорціум Всесвітньої павутини (W3C). Мета W3C полягала в тому, щоб стандартизувати протоколи і технології, які використовуються для створення Вебу, що, своєю чергою, дозволило би зробити контент максимально доступним для жителів усього світу.

Протягом наступних декількох років W3C опублікував низку специфікацій, зокрема HTML 4.0, формат картинок PNG і версії каскадних стильових таблиць CSS1 і CSS2 [2]. Консорціум виконав величезну роботу, випустивши більш ніж 80 специфікацій та рекомендацій, які отримали загальну назву «веб-стандарти» [136]. Відтоді у професійному співтоваристві веб-розробників правилом хорошого тону стало проходження веб-стандартів.

Веб-стандарти – це відкриті, не захищені будь-якими патентами специфікації та рекомендації W3C, організації, що розробляє і впроваджує технологічні стандарти для мережі Інтернет. Вони визначають відкриту веб-платформу для розроблення застосунків, що дозволяє створювати інтерактивні потужні системи, які взаємодіють із величезними сховищами даних і доступні на будь-якому пристрої.

Дотримання веб-стандартів дає змогу створювати семантичні, доступні, відмовостійкі сайти. Розгляньмо базові веб-стандарти: специфікація HTML (або XHTML), каскадні таблиці стилів CSS та вбудовані скрипти Javascript – три основні технології побудови веб-сторінок [12; 15].

**HTML** – мова розмітки гіпертекстових документів, що є стандартною мовою розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомоги мови HTML. Документ HTML обробляється браузером і відтворюється на екрані у звичному для людини вигляді.

HTML впроваджує такі засоби [12]:

- створення структурованого документа через позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та ін.;
- отримання інформації із Всесвітньої мережі через гіперпосилання;
- створення інтерактивних форм;
- включення зображень, звуку, відео та інших об'єктів до тексту.

HTML використовується для розділення документа, визначення його змісту і структури та визначення значення кожної з частин. HTML-документ містить увесь текст, що відтворюється на веб-сторінках. HTML використовує елементи для визначення різних компонентів сторінки. Цей основний офіційно затверджений W3C стандарт – специфікація, яка визначає мову розмітки гіпертексту HTML. Специфікація містить багато різноманітної та корисної інформації, стислий історичний огляд розвитку HTML, важливі рекомендації та зауваження щодо створення HTML-документів, синтаксичні правила мови розмітки гіпертексту і визначення семантики його елементів. W3C стандарт визначає такі базові поняття і вимоги до HTML-документів: – валідність, яка визначає властивість HTML-документа, що передбачає відповідність його коду синтаксичним правилам мови розмітки гіпертексту, завдяки чому створений документ буде коректно оброблено і відображено більшістю браузерів; – семантика полягає у структурній розмітці, що визначає таку структуру документа, в якому розташування і зв'язок його частин

складають єдине ціле; мова розмітки гіпертексту HTML дозволяє за допомоги спеціальних керуючих конструкцій (тегів) розмітити структуру документа; – конформність означає повну відповідність HTML-документів веб-стандартам. XHTML.

З метою розширення HTML і підвищення сумісності з іншими форматами даних було розроблено розширювану мову розмітки гіпертексту XHTML. Це – мова розмітки, що має таку саму виразну силу, як і HTML, але відповідає синтаксичним правилам розширюваної мови XML. W3C запропонував XML як стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема через Інтернет.

XML-документ складається з текстових знаків. Людина може читати і досить легко зрозуміти такий документ. Стандарт XML визначає набір базових лексичних і синтаксичних правил для побудови мови опису інформації через застосування простих тегів. Цей формат достатньо гнучкий для застосування в різних галузях. Запропонований стандарт визначає метамову, на основі якої через запровадження обмежень на структуру та зміст документів визначаються специфічні, предметно орієнтовані мови розмітки даних. Саме такою, орієнтованою на розмітку сайтів в Інтернеті, є мова XHTML. Оскільки XHTML-документи мають бути коректними XML-документами, їх оброблення можна здійснювати стандартними інструментами оброблення XML-документів на відміну від HTML, який вимагає порівняно складніших і повільніших синтаксичних аналізаторів.

XHTML можна розглядати як численні перетинання HTML і XML. У документі Рекомендацій XHTML 1.0, який W3C опублікував у серпні 2002 р., зазначалося, що сімейство XHTML є подальшим кроком в еволюції Інтернету. У спосіб переходу на XHTML сьогодні розробники контенту можуть увійти у світ XML з усіма супутніми перевагами, залишаючись упевненими у зворотній та майбутній сумісності їхнього контенту [19].

**HTML5.** Це наступна версія мови HTML. Чернетку п'ятої версії мови гіпертекстової розмітки HTML було представлено 22 січня 2008 р. Робота над HTML5 почалася в кінці 2003 р. як доказ концепції, що можна розширити форми HTML, не вимагаючи змін, не сумісних із наявними веб-сторінками. Відповідно до плану розвитку стандарту HTML5, представленого у вересні консорціумом W3C [32], фінальні специфікації HTML 5.0 буде опубліковано в четвертому кварталі 2014 р., після чого почнеться робота над стандартом HTML 5.1, який планується випустити в кінці 2016 р. У HTML 5.0 з'являться специфікації, що будуть стабілізовані й погоджені; інші специфікації, з яких у процесі підготовки HTML 5.0 залишаються невирішені проблеми, буде відкладено і долучено до складу стандарту HTML 5.1. Так до стандарту HTML5 поступово, мірою готовності, додаватимуться нові специфікації та розширення.

Поточна версія HTML5 підтримує такі можливості:

- нові теги, а деякі вилучено; в цілому з нової версії мови розмітки пропонується прибрати близько 15 тегів;
- теги та , скориставшись якими, дуже просто додати до сторінки відео-чи аудіодоріжку;
- векторну графіку (тег ); за допомоги цього тегу розробники зможуть створювати двомірні зображення та анімацію засобами JavaScript;
- можливість вставки SVG графіки просто в тіло HTML документа;
- збереження даних на стороні клієнта, що значно зменшить обсяг інформації, яка передається між веб-браузером і сервером через Web 2.0 застосунки.

Каскадні таблиці стилів CSS [13] не є мовою програмування, як, наприклад, JavaScript, і не є мовою розмітки типу HTML. Насправді це унікальний тип технології, тому немає аналогів, з якими її можна порівнювати. Технології, які визначали інтерфейси до розвитку Web, завжди змішували представлення і структуру документа, що малоприйнятно в такому мінливому середовищі, яким є Web. Засобом, що вирішив цю проблему, став CSS. CSS

(каскадна чи блочна верстка) заступила табличну верстку вебсторінок. Головна перевага блочної верстки полягає в розділенні змісту сторінки (даних) та їхньої візуальної презентації.

CSS дають розробникові сайту контроль над форматуванням і розміткою документа: дозволяють визначити, як елемент повинен відображатися; використовуючи стильові декларації, легко змінити, наприклад, висоту рядка для всіх абзаців або позначити всі заголовки другого рівня певним кольором, змінювати або додавати кольори, фон, розміри і стиль тексту, розміщувати елементи на веб-сторінці в різних місцях.

У той час як HTML структурує документ і повідомляє браузеру, яку функцію має певний елемент (чи буде це посилання на іншу сторінку або заголовок), CSS видає браузеру інструкції про те, як вивести певний елемент – оформлення, розміщення пропусків позиціонування.

Якщо HTML є колодами і цеглинами, які складають структуру будинку, CSS є штукатуркою і фарбуванням для його оброблення. Це реалізується за допомоги системи правил, що визначають, які елементи HTML повинні бути додатково оформлені, і в кожному правилі описуються властивості (наприклад, колір, розмір, шрифт і т. ін.) цих елементів HTML, якими вони будуть маніпулювати, і які значення буде для них задано.

Одна з головних переваг CSS полягає в можливості розділити зміст сторінки (HTML або XML контент, наповнення, або подібна мова розмітки) від вигляду документа (що описується в CSS). Таке розділення може покращити сприйняття і доступність контенту, забезпечити більшу гнучкість і контроль за відображенням контенту в різних умовах, зробити контент більш структурованим і простим, прибрати повтори тощо.

CSS також дозволяє адаптувати контент до різних умов відображення (на екрані монітора, мобільного пристрою, у роздрукованому вигляді, на екрані телевізора, пристроях з підтримкою шрифту Брайля або голосових браузерах тощо).

Один і той самий HTML або XML-документ може бути відображений по-різному залежно від використаного CSS. Підтримуються такі стилі для відображення сторінки:

- стилі автора: інформацію надано автором сторінки;
- зовнішні таблиці стилів (англ. stylesheet), найчастіше окремий файл або файли .css;
- внутрішні таблиці стилів, включені як частина документа чи блоку; – стилі для окремого елемента;
- стилі користувача: локальний .css-файл, вказаний користувачем для використання на сторінках і вказаний в налаштуваннях браузера (наприклад, Opera);
- стилі браузера: стандартний стиль переглядача, наприклад, стандартні стилі для елементів, визначені браузером, використовуються, коли немає інформації про стиль елемента або вона є неповною.

Стандарт CSS визначає порядок та діапазон, послідовність та елементи застосування стилів. Так використовується принцип каскадності, коли для елементів вказується лише та інформація про стилі, що змінилася або не визначена загальнішими стилями.

У результаті застосування CSS сайт має такі переваги:

- інформація про стиль для всього сайту або його частин може міститися в одному .css-файлі, що дозволяє швидко робити зміни в дизайні та презентації сторінок;
- різна інформація про стилі для різних типів користувачів: наприклад, великий розмір шрифту для користувачів із послабленим зором, стилі для виводу сторінки на принтер, стиль для мобільних пристроїв;
- сторінки зменшуються в обсязі та стають більш структурованими, оскільки інформація про стилі відділена від тексту і має певні правила застосування;



- прискорення завантаження сторінок і зменшення обсягів інформації, що передається, навантаження на сервер і канал передавання, оскільки сучасні браузері здатні кешувати (запам'ятовувати) інформацію про стилі та використовувати для всіх сторінок, а не завантажувати для кожної.

Дотримання стандартів HTML і CSS у створенні сайтів забезпечує такі важливі властивості:

- ефективність коду;
- легкість підтримки;
- доступність;
- сумісність із пристроями;
- потрапляння в коло пошукових систем або павуків.

**Вбудовані скрипти JavaScript.** Інструментом створення вбудованих скриптів є скриптова мова, розроблена для запису «сценаріїв», послідовностей операцій, що їх користувач може виконувати на комп'ютері. Сценарії зазвичай інтерпретуються, а не компілюються. Сценарій (скрипт) є програмою, що автоматизує певне завдання, яке без сценарію користувач робив би вручну, використовуючи інтерфейс програми.

**JavaScript** є скриптовою мовою, яка використовується, щоби додати поведінку (behaviour) на веб-сторінку. Вона може використовуватися для перевірки даних, введених у форми, і повідомляти, якщо дані введені неправильно, надає функціональність перетягування (drag and drop functionality), дозволяє міняти стилі на льоту, анімувати такі елементи сторінки, як меню, додавати функціональність кнопки і т. д. Багато скриптів JavaScript працюють через доступ до цільового HTML-елементу і здійснення якоїсь дії над ним. Нагадує CSS, але принцип роботи і синтаксис відрізняються.

Хоча найпоширеніше і найвідоміше застосування JavaScript стосується написання сценаріїв для веб-сторінок, ця мова використовується і для впровадження сценаріїв керування об'єктами, вбудованими в інші програми.

Наразі JavaScript є однією з найпопулярніших мов програмування в Інтернеті. Важливим чинником розповсюдження JavaScript став підхід до побудови інтерфейсів користувача у веб-застосуванні AJAX (Asynchronous JavaScript And XML), завдяки якому веб-сторінка, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти довантажує потрібні користувачу дані. Це суттєво змінило ситуацію та привернуло увагу професійної спільноти до JavaScript. У результаті було розроблено та вдосконалено багато практик використання (зокрема тестування й налагодження), створено бібліотеки та фреймворки, що спричинило використання JavaScript поза браузером [95].

На сьогоднішній день JavaScript підтримується більшістю браузерів. Текст програми вводиться безпосередньо в HTML-документ та інтерпретується самим браузером (точніше, вбудованим у браузер рушієм JavaScript). Застосовується в основному для часткової автоматизації оброблення і маніпуляції даними, що їх використовує сторінка. У розробленні великих і нетривіальних веб-застосунків із використанням JavaScript критично важливим є доступ до інструментів налагодження (debugging). Оскільки браузери від різних виробників дещо відрізняються у поведінці JavaScript і реалізації об'єктної моделі документа, треба мати в руках засоби налагодження для кожного браузера, якщо веб-застосунок орієнтується на нього.

Поточні версії браузерів – Internet Explorer, Firefox, Opera, Google Chrome та Safari мають свої засоби налагодження. Оскільки JavaScript є інтерпретатором без строгої типізації та може виконуватись у різних середовищах зі своїми особливостями сумісності, програміст має бути дуже уважним і повинен перевіряти, що його код виконується так, як очікується в різноманітних конфігураціях. Дуже часто скрипт працює в одному середовищі, проте видає некоректні результати в іншому.

Кожен блок сценарію інтерпретатор розбирає окремо. На веб-сторінках, коли треба комбінувати блоки JavaScript та HTML, синтаксичні помилки

знайти легше, якщо функції сценарію розміщувати в окремому блоці коду або використовувати короткі зв'язані .js файли. За такого підходу синтаксична помилка не спричинятиме «падіння» цілої сторінки.

У цілому JavaScript забезпечує динамічні функції сайту. Невеликі програми на мові JavaScript можуть працювати на клієнтському комп'ютері, не вимагаючи, щоб на сервері було встановлено якесь спеціальне програмне забезпечення; JavaScript дозволяє додавати на сайт певну базову функціональність та інтерактивність, але ця мова має свої обмеження. Складний функціонал потребує програмування на стороні сервера і підтримки динамічних веб-сторінок.

## **2.2. Методологія побудови інформаційних систем на основі веб-технологій**

Методологія створення інформаційних систем полягає в організації процесу побудови ІС та забезпеченні управління цим процесом для того, щоб гарантувати виконання вимог як до самої системи, так і до характеристик процесу розроблення.

Розроблення ІС на основі веб-технологій ґрунтується на загальній методологічній базі проектування інформаційних систем і типових проектних рішеннях із використанням архітектури, складниками якої є веб-сервери, веб-портали, веб-застосунки.

З огляду на це методологію створення ІС на основі веб-технологій представлено як сукупність загальних методологічних засад побудови ІС та проектних рішень, що забезпечують функціонування і доступність системи в мережі Інтернет. 2.1.

Методологія створення інформаційних систем  
Методологія створення інформаційних систем повинна забезпечувати за допомоги відповідного набору інструментальних засобів виконання завдань із дотриманням таких умов:

- створення інформаційних систем, що відповідають цілям, завданням установи (підприємства) та визначеним вимогам щодо автоматизації ділових процесів;
- гарантію створення системи із визначеними параметрами протягом заданого часу в рамках обумовленого бюджету;
- простоту супроводу, модифікації та розширення системи з метою забезпечення її відповідності у мінливих умовах роботи установи (підприємства);
- забезпечення створення інформаційних систем, що відповідають вимогам відкритості, переносимості та масштабованості;
- можливість використання в системі розроблених раніше і застосованих на підприємстві засобів інформаційних технологій (програмного забезпечення, баз даних, засобів обчислювальної техніки, телекомунікацій).

Існують методології, що враховують усі стадії життєвого циклу (далі – ЖЦ) програмного забезпечення. ЖЦ, який описує сукупність окремих етапів робіт, що проводяться в заданому порядку протягом періоду часу, починається з вирішення питання про розроблення програмного забезпечення і закінчується припиненням використання програмного забезпечення [95]. У загальному випадку ЖЦ визначається моделлю та описується у формі методології (методу). Модель або парадигма ЖЦ визначає загальну організацію, основні його фази та принципи переходу між ними. Вона представляє структуру, що складається з процесів, робіт і задач, що забезпечують проектування, експлуатацію і супровід програмного продукту.

Методологія (метод) визначає комплекс робіт, їхній детальний зміст і рольову відповідальність спеціалістів на всіх етапах вибраної моделі. Саме методологія визначає, які мови і системи будуть застосовуватися для розроблення програмного забезпечення, і часто рекомендує, який технологічний підхід буде використано.

Розгляньмо дві провідні методології проектування, в яких інструментально підтримуються всі етапи життєвого циклу розроблення ПЗ, та методологію моделювання бізнес-процесів в інформаційних системах.

Швидке розроблення застосунків, RAD (англ. Rapid application development) – методологія створення засобів розроблення застосунків, програмних продуктів, яка приділяє особливу увагу швидкості та зручності програмування, створенню технологічного процесу, що дозволяє максимально швидко створювати комп'ютерні програми [12].

Методологія RAD пов'язана з концепцією візуального програмування, оскільки охоплює комплекс спеціальних інструментальних засобів швидкого розроблення прикладних інформаційних систем, що дозволяють оперувати з певним набором графічних об'єктів для функціонального відображення окремих інформаційних компонент у застосунках.

RAD охоплює процес розроблення інформаційних систем, що має такі властивості:

- невелика команда програмістів (зазвичай від 2 до 10 осіб);
- ретельно опрацьований виробничий графік робіт, що розраховується на порівняно короткий термін розроблення (від 2 до 6 міс.);
- використовує ітераційну (спіральну) модель розроблення;
- повне завершення робіт на кожному з етапів ЖЦ не є обов'язковим.

RAD визначає такі фази розроблення:

- планування (Requirements Planning Phase): сукупність вимог, отриманих за системного планування та аналізу процедури розроблення ЖЦ; на цьому етапі користувачі, менеджери та ІТ-фахівці обговорюють завдання проекту, його обсяг, системні вимоги, а також складнощі, які можуть виникнути в розробленні; фаза завершується узгодженням ключових моментів із RAD-

групою та отриманням від керівників проекту дозволу на продовження;

- користувальницьке проектування (User Design Phase): протягом цього етапу користувачі, взаємодіючи з системними аналітиками, розробляють моделі й прототипи, які мають усі необхідні системні функції; для перекладу користувальницьких прототипів у робочі моделі RAD-група зазвичай використовує спеціальну техніку розроблення застосунків; користувальницьке проектування виявляється тривалим ітеративним процесом, який дозволяє користувачам зрозуміти, змінити і в підсумку вибрати робочу модель, що відповідає їхнім вимогам;
- конструювання (англ. Construction Phase) – етап, основна задача якого полягає в розробленні програм і застосунків, написанні коду, інтеграції модулів і системному тестуванні;
- перемикання (англ. Cutover phase ) – етап, що охоплює операції з конверсії даних, тестування, перехід на нову систему і тренування користувачів.

**Методологія RUP** (англ. Rational Unified Process). RUP є ітеративним процесом розроблення програмного забезпечення, створеним Rational Software – підрозділом IBM 2003 р. Ця методологія спирається на перевірені практикою методи аналізу, проектування і розроблення ПЗ, методи управління проектами, а також забезпечує прозорість і керованість процесу, дозволяє створювати ПЗ відповідно до вимог замовника на момент здачі ПЗ із урахуванням можливостей інструментальних засобів підтримки розроблення [1].

В основі методології RUP лежить покроковий підхід. Він визначає етапи життєвого циклу, контрольні точки, правила робіт для кожного етапу, впорядковуючи цим проектування і розроблення ПЗ.

Для кожного етапу життєвого циклу використовуються такі будівельні блоки RUP:

- роботи (склад і послідовність робіт, а також правила їх виконання);
- ролі (розподіл повноважень серед учасників проекту);
- робочі продукти (склад і шаблони формованих проміжних і підсумкових документів, а також порядок контролю та перевірки якості).

Методологія RUP дозволяє об'єднати проектну команду, надаючи в її розпорядження перевірені світовою практикою кращі підходи до розроблення ІС. До них належать такі процеси створення ПЗ, як управління проектами, бізнес-моделювання, управління вимогами, аналіз і проектування, тестування і контроль змін.

Впровадження RUP сприяє виробленню якісних внутрішньокорпоративних стандартів і підвищенню загальної культури розроблення. Основою RUP є ітеративний процес розроблення. RUP визначає життєвий цикл проекту, що складається з чотирьох фаз:

- початкова фаза;
- фаза уточнення;
- фаза конструювання;
- фаза впровадження.

У RUP описано шість найкращих практик, які є парадигмою програмної інженерії щодо шести ідей, яких варто дотримуватись у конструюванні будь-якого проекту, щоб мінімізувати провали та збільшити продуктивність.

1. Ітеративне розроблення.
2. Управління вимогами.
3. Використання компонент.
4. Візуальне моделювання.
5. Перевірка якості.
6. Контроль змін.

**Методологія UML.** Представлені вище методології широко застосовують візуальне моделювання, що забезпечує ефективний механізм

визначення і формалізації вимог до ІС. Уніфікована мова моделювання UML є інструментом такої візуалізації у вигляді діаграми, що представляє основні компоненти бізнес-процесів, користувачів та їхньої взаємодії в ІС.

UML – мова графічного опису для об'єктного моделювання у розробленні програмного забезпечення. UML є мовою широкого профілю, це – відкритий стандарт, який використовує графічні позначення для створення абстрактної моделі системи, званої UML-моделлю.

Мову UML було створено для визначення, візуалізації, проектування та документування переважно програмних систем. Мова UML може бути застосована на всіх етапах життєвого циклу аналізу інформаційних систем і розроблення прикладних програм для їх функціонування. Семантика мови UML визначається для двох видів об'єктних моделей: структурних моделей та моделей поведінки. Структурні (статичні) моделі описують структуру сутностей або компонентів системи, включаючи їхні класи, інтерфейси, атрибути і зв'язки.

Моделі поведінки (динамічні моделі) описують поведінку або функціонування об'єктів системи, зокрема їхні методи, взаємодію (співробітництво) між ними, а також процес зміни станів окремих компонентів і системи в цілому. Словник мови UML містить сутності – абстракції, що є основними елементами моделі, та відношення – основні сполучні будівельні блоки. Сутності та відношення за певними правилами поєднуються в конструкції – діаграми.

В UML визначено чотири типи сутностей [3]:

- структурні сутності;
- сутності поведінки;
- групувальні сутності;
- анотаційні сутності.

Кожен елемент нотації UML має унікальне графічне позначення та специфікацію – текстове представлення синтаксису і змістовної семантики відповідного будівельного блоку.



У рамках мови UML всі подання про моделі складної системи фіксуються у вигляді спеціальних графічних конструкцій, що одержали назву діаграм [48]. Діаграма в UML – це графічне представлення набору елементів, що зображується у вигляді зв'язного графу з вершинами (сутностями) і ребрами (відношеннями). Основна мета діаграм – візуальне моделювання розроблюваної системи чи її архітектури.

Типовий набір містить такі види UML-діаграм: діаграма класів (англ. Class diagram), діаграма компонентів (англ. Component diagram), діаграма композитної / складової структури (англ. Composite structure diagram), діаграма розгортання (англ. Deployment diagram), діаграма об'єктів (англ. Object diagram), діаграма пакетів (англ. Package diagram), діаграма діяльності (англ. Activity diagram), діаграма автомата, діаграма кінцевого автомата, діаграма станів (англ. State Machine diagram), діаграма, на якій представлений кінцевий автомат з простими станами, переходами і композитними станами; діаграма прецедентів (англ. Use case diagram), діаграма послідовності (англ. Sequence diagram), діаграма комунікації (англ. Communication diagram), діаграма співпраці, діаграма огляду взаємодії (англ. Interaction overview diagram), діаграма синхронізації (англ. Timing diagram), діаграма станів (англ. Statechart diagram). Кожна із цих діаграм деталізує та конкретизує різні подання про моделі складної системи в термінах мови UML, а інтегрована модель складної системи в нотації UML представляється у вигляді сукупності зазначених вище діаграм.

### **2.3. Проектні рішення побудови інформаційних систем на основі веб-технологій**

Будь-яку ІС спрямовано на виконання двох загальних завдань: забезпечення певної функціональності та взаємодії з користувачем. Для ІС, побудованих на основі веб-технологій, характерним є те, що на рівні архітектури системи виокремлюються дві частини: частина забезпечення функціональності, яку називають back end, і частина забезпечення взаємодії з користувачем під назвою front end [5]. Це обумовлюється розподіленістю веб-

системи (серверів, клієнтів) і відповідає положенню програмної інженерії, де терміни front end і back end відображають розділення задач між рівнем представлення і рівнем доступу до даних. Виходячи з такого підходу проектні рішення щодо побудови ІС на основі веб-технологій будемо розглядати на рівні front end, який представляється у вигляді порталу або сайту ІС, та рівні back end, що охоплює сервери та прикладні застосунки ІС.

### **Принципи побудови і функціонування порталів і сайтів (front end).**

Відповідно до одного з найбільш універсальних означень, портал є захищеною точкою взаємодії з різноманітною інформацією, бізнеспроцесами і людьми, відповідно до вимог та обов'язків кожного користувача. Взаємодія з користувачами відбувається через веб-інтерфейс.

Портал – це вхід (або вихід) у глобальний інформаційний простір [77]. Портал характеризується існуванням розвиненої системи інформаційних ресурсів і активною взаємодією з користувачами через різні сервіси.

Портал має централізований вхід і спеціальні засоби для зручної навігації по інформаційних ресурсах. Це – веб-сайт, призначено для певної аудиторії користувачів, що здійснює аналіз, оброблення і доправлення контенту, надає доступ до сервісів і застосунків на основі персоналізації для конкретного користувача.

Портал забезпечує такі функції:

- інтеграцію та агрегацію великого обсягу різнотипних даних;
- гнучкий пошук;
- персоналізацію змісту порталу для певного користувача.

Сучасні портали характеризуються розвиненим інструментарієм [94]. До складу сервісу і служб порталу входять:

- сумісні базова служба та сервіс, характерні для порталів всіх типів;
- спеціалізовані служби, що забезпечують можливості адекватного доступу до різних електронних ресурсів.

Базовий сервіс містить:

- сервіс навігації та пошуку інформації по ресурсах порталу;

- інформаційний сервіс;
- сервіс інтерактивного спілкування користувачів порталу;
- сервіс персоніфікації порталу;
- сервіс моніторингу і статистики.

Базова служба містить:

- службу персонального порталу користувача та її сервіс;
- службу аутентифікації та авторизації доступу до змісту порталу.

Сервіс навігації та пошуку інформації по ресурсах порталу містить:

- каталог ресурсів, призначений для реєстрації та подальшої каталогізації всіх інформаційних ресурсів, що входять до порталу, а також забезпечення інформаційної та функціональної бази для ефективного пошуку серед зареєстрованих ресурсів, зокрема з використанням банку інформаційних об'єктів;
- карту порталу, призначену для відображення основних змістовних розділів порталу, яка пропонує користувачеві огляд основних функціональних можливостей порталу;
- метапошукову систему, що здійснює пошук інформації серед інформаційних джерел порталу, а також із використанням зовнішніх інформаційних джерел.

**Програмно-технологічною платформою** для побудови і підтримки системи порталів є програмно-апаратний комплекс, який дозволяє будувати і підтримувати портали різного призначення й архітектури та забезпечувати виконання такого набору функцій: підтримка комунікацій, персоналізація, профілювання, пошук, забезпечення безпеки, стандартний веб-доступ до порталу, виконання застосунків, можливість спільної роботи, керування вмістом, керування користувачами, контроль і керування продуктивністю, керування знаннями [94].

Для мінімізації витрат на побудову порталу програмно-технологічна платформа повинна містити і забезпечувати певний базовий набір сервісів, склад якого наведено на рис. 2.1.

Служби пакування і середовище портлетів	Адаптація контенту і навігації	Визначення пристроїв та мережні служби	Компонент служб представлення
Служби персоналізації		Служби безпеки	Компонент користувальницьких служб
Служби публікації	Служби підписки	Служби підтримки співробітництва	
Служби доступу та пошукова система	Служби доставки	Служби керування документообігом	
Сховище спільної інформації	Інформаційний каталог	Каталог правил	
Менеджер розбиття за категоріями	Менеджер об'єднання контенту	Менеджер подій	Компонент керування інформацією
Адаптери БД і файлів	Адаптери інструментів знань	Адаптери керування контентом	Компонент адаптерів порталу
Адаптери ПЗ підтримки співробітництва та офісних застосунків	Інструменти розроблення адаптерів	Адаптери інтеграції застосунків	
Засоби розробки		Засоби керування продуктивністю та адміністрування	Компонент Web-інфраструктури
Сервер Web-застосунків			

Рис. 2.1. Компоненти базового набору сервісів порталу

Успішне функціонування порталу великою мірою залежить від правильності вибору програмної платформи, яка, своєю чергою визначає первинні вимоги до апаратної платформи порталу. Вирізняються загальні (інваріантні до програмної платформи) вимоги до апаратної платформи порталу:

- відповідність міжнародним стандартам відкритих систем, зокрема стандартам розроблення, супроводу й документування;
- відповідність міжнародним стандартам в галузі керування якістю ISO 9000;

- інтегрованість, що забезпечує існування розвинених технологічних засобів інтеграції з іншими прикладними системами та базами даних;
- адаптованість, що означає наявність засобів налаштування порталу під функціональні вимоги конкретної організаційної системи освітнього закладу, а також підтримує технології перенесення рішень з одної платформи до іншої;
- багатоплатформність, яка забезпечує здатність системи працювати під керуванням різних операційних систем;
- розподіленість, що дозволяє будувати портал (системи порталів) на деяких, зокрема географічно віддалених, серверах; можливість створення дзеркальних серверів;
- масштабованість, властивість платформи, що характеризується кількістю користувачів; обсягом даних, що зберігаються; інтенсивності обміну даними; швидкості оброблення запитів і даних; набору послуг, що надаються; способами забезпечення доступу тощо;
- надійність системи повинна підтримуватися на рівні не менше ніж 99,7% (процентне співвідношення часу безперебійної роботи до часу роботи системи);
- система повинна характеризуватися надмірністю блоків живлення;
- надійність системи повинна підтримуватися на рівні не менше ніж 99,7% (процентне співвідношення часу безперебійної роботи до часу роботи системи);
- система повинна характеризуватися надмірністю блоків живлення;
- платформа повинна підтримувати динамічну реконфігурацію на рівні мікроядра та ядра операційної системи;

- система повинна забезпечувати оброблення ситуацій, пов'язаних зі збоєм окремих компонент з подальшим автоматичним їх вилученням із конфігурації після перезавантаження.

**Функціонал програмно-технологічної платформи повинен задовольняти таким вимогам:**

- виконання застосунків: дозволяти легко розробляти, розгортати і керувати різними застосунками;
- можливість спільної роботи: дозволяти окремим користувачам і великим організаціям об'єднати свої ресурси і працювати разом через Інтернет;
- керування вмістом: надавати гнучкість виробництву і керуванню окремими веб-вузлами, а кінцевому користувачеві – налаштований під нього (персоніфікований) вміст порталу;
- керування користувачами: організація керування користувачами, ресурсами та безпекою всередині і зовні системи мережного захисту;
- контроль і керування продуктивністю: керування трафіком, динамічне кешування даних, кешування мережі;
- керування контентом: об'єднання внутрішньої та зовнішньої інформації та надання інформації, заснованої на контекстній концепції;
- підтримка комунікацій: система електронної пошти й повідомлень;
- підтримка персоналізації інформації, що базується на аналізі даних про користувача в режимі реального часу;
- підтримка профілювання: відстеження, аналіз та передбачення дій користувачів порталу з урахуванням поведінки клієнта через його уподобання;
- підтримка функції пошуку;

- виконання функцій безпеки: внутрішній та зовнішній захист для запобігання несанкціонованого доступу до мережі (Firewall), унікальність реєстрації (SSO –Single Sign-On);
- стандартний www-доступ до порталу для контент адміністрування.

**Вимоги до апаратної частини системи:**

- продуктивність;
- збереження даних (дискова підсистема, гаряча заміна);
- розширюваність (масштабованість);
- підтримка необхідної конективності (зв'язаність компонентів системи, можливість з'єднання, наприклад, комп'ютерів між собою та здатність до взаємодії, наприклад, програм між собою);
- надійність;
- гарантія та підтримка виробника.

**До вимог до програмної частини системи відносять:**

- відповідність стандартам (міжнародним) відкритих систем, зокрема стандартам розроблення, супроводу і документування;
- відкритий API і засоби для розроблень (інструментарій, документація);
- переносимість (портування) програмного забезпечення.

Головним завданням у портуванні є збереження звичних користувачеві інтерфейсу і способів роботи з пакетом і його властивостями. Додавання нових або видалення частини наявних властивостей у портуванні програмних продуктів не допускається.

**Вимоги до веб-застосунків.** Якість застосунку, що розробляється для виконання окремих задач на порталі, визначається тим, наскільки він відповідає вимогам, закладеним на стадії проектування системи. Всі вимоги до застосунків поділяють на функціональні та не функціональні. Функціональні вимоги визначають ту функціональність системи, яку

розробники повинні побудувати для того, щоб користувачі змогли виконати свої задачі в рамках своїх бізнес-процесів.

Основні нефункціональні вимоги до веб-застосунків:

– *Надійність*. Формально надійність полягає у властивості зберігати в часі та встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах і умовах застосування (ГОСТ 27.002-89). Тобто, вимоги до надійності застосунку визначаються умовами його функціонування (параметри сервера, максимальна кількість користувачів застосунку), а також граничними показниками якості роботи системи за цих умов (час обробки запиту користувача до системи, кількість відмов системи). Отже, надійний веб-застосунок має забезпечувати доступ до всіх функцій для користувача за будь-яких умов (усі можливі умови для цього застосунку повинні бути розглянуті та враховані під час проектування системи).

– *Швидкодія*. Швидкодія застосунку визначається як середній час обробки запиту користувача до системи. Максимальним прийнятним часом відповіді для веб-застосунків вважається 5 секунд.

– *Безпека*. Вимога безпеки веб-застосунку передбачає: розмежування прав доступу до функцій і даних кожного компонента вебзастосунку, контроль рівня доступу компонентів та/або користувачів, авторизацію та верифікацію користувачів.

– *Масштабованість*. Це – здатність системи збільшувати свою продуктивність за умов підвищеного навантаження і додавання ресурсів. Для користувача масштабованого веб-застосунку має залишатися непомітним момент (тобто час відповіді систем на запити користувача не повинен помітно змінюватися), коли зросте навантаження (наприклад, до застосунку отримують доступ одночасно ще декілька користувачів), і зі зміною конфігурації застосунку (наприклад, якщо на рівень бізнес-логіки буде додано додатковий компонент обробки даних).



**Засоби інформаційного наповнення сайтів і порталів.** Засоби керування інформаційним наповненням сайтів і порталів (англ. Content Management Systems, CMS) дозволяють оперативно і своєчасно керувати як наповненням сайтів, так і інформацією в цілому. Ці засоби дозволяють здійснювати централізоване керування накопичуваними даними, відокремлення змісту від представлення (тобто від дизайну сайту чи застосування, що виступає як клієнт CMS-рішення), автоматизацію керування життєвим циклом інформаційного наповнення, використання інформаційного наповнення різними користувачами для різних задач.

CMS представляє систему для забезпечення та організації спільного процесу створення, редагування й керування текстовими і мультимедіа документами (контентом). Зазвичай цей контент розглядається як неструктуровані дані предметного завдання на противагу структурованим даним під керуванням СКБД.

CMS дозволяють керувати текстовим і графічним наповненням веб-сайту, надаючи користувачеві зручні інструменти зберігання й публікації інформації.

CMS виконує такі основні завдання:

- збирання в єдиному цілому та об'єднання на основі ролей і завдань усіх різнотипних джерел знань та інформації, доступних як усередині організації, так і за її межами;
- забезпечення взаємодії співробітників, робочих груп і проектів зі створеними ними базами знань, інформацією й даними так, щоб їх легко можна було знайти, витягти й повторно використовувати звичним для користувача способом.

Більша частина сучасних систем керування вмістом реалізується за допомоги візуального (WYSIWYG) редактора – програми, яка створює HTML-код зі спеціальною спрощеною розміткою, що дозволяє користувачеві простіше формувати текст.

**Принципи побудови функціоналу ІС (back end).** Основою архітектури сучасної ІС є багаторівнева клієнт-серверна модель, в якій виокремлюються такі рівні:

- рівень клієнтів;
- рівень доступу до ресурсів ІС;
- рівень застосунків та бізнес-логіки;
- рівень даних системи.

Рівень клієнтів та рівень доступу до ресурсів визначає портали (front end ІС), а рівень застосувань та бізнес-логіки (back end ІС) визначає веб-сервери та сервери застосунків, на яких розгортаються компоненти прикладних застосунків і сервісів. На рівні back end через застосунки (прикладні системи, підсистеми, окремі програми та компоненти) реалізується безпосередня функціональність системи, алгоритми обчислень та обробки даних, задачі та функції системи.

Сучасний погляд на архітектуру застосунків ґрунтується на поняттях рівнів і шарів архітектури. Загалом визначено такі архітектурні рівні:

- рівень подання: призначення цього рівня полягає у підготуванні, формуванні та відображенні результатів роботи застосунку для кінцевого користувача, а також забезпечення інтерфейсу для взаємодії з прикладним застосунком, зокрема на цьому рівні перебуває графічний інтерфейс користувача (GUI);

- рівень логіки застосунку: на цьому рівні знаходиться основна функціональність застосунку, визначаються бізнес-процеси та алгоритми обробки даних;

- рівень джерел даних: цей рівень визначає зв'язок з інформаційними ресурсами застосунку; кожне джерело розглядається як компонента з уніфікованим інтерфейсом, яка інкапсулює внутрішні подробиці реалізації та зберігання даних;

- рівень інформаційних ресурсів: на цьому рівні безпосередньо перебувають інформаційні ресурси – бази даних, файлові структури, окремі

документи та ін.; самі ресурси можуть входити до складу застосунку або належати до інших складових інформаційної системи: зв'язок із ресурсом відбувається за допомоги відповідного джерела даних, яке по суті є менеджером доступу до ресурсу.

На рис. 2.2 відображено зв'язок між наведеними поняттями.

Рівні архітектури	Шари рівнів архітектури	Розміщення
Подання	Відображення (клієнтська частина)	Клієнт
	Зовнішні служби	Веб-сервер
	Контролер застосунків (серверна частина)	
Логіка застосування	Прикладні служби	Сервер застосунків
	Бізнес-логіка, модель домену	
Джерела даних	Перетворення, відображення	Сервер баз даних, файлова система
Інформаційні ресурси	моделей даних	
	Менеджери інформаційних ресурсів, конектори	
	Логіка керування ресурсом	
	Дані, інформація	

Рис. 2.2. Схема розташування рівнів і шарів архітектури застосунків ІС

Рівень подання прикладного застосунку складається з трьох шарів: відображення, яке реалізується на клієнтській компоненті, зовнішніх веб-служб і контролера застосунків, які розміщуються на веб-сервері.

Рівень логіки застосування поділяється на два шари: прикладних служб та бізнес-логіки. Ці архітектурні шари розміщуються на сервері застосунків. За обмежених системних вимог до цих шарів вони можуть замість сервера застосунків розташовуватися на Webсервері.

Рівень джерел даних прикладного застосунку складається з двох шарів: відображення моделей даних і менеджерів інформаційних ресурсів. Ці архітектурні шари розміщуються на сервері застосунків. За обмежених системних вимог до цих шарів вони можуть замість сервера застосунків розташовуватись на Web-сервері [5].

Рівень інформаційних ресурсів складається з двох шарів: логіки керування ресурсом та безпосередніх даних. До першого шару належать прикладні компоненти, які є складниками самого інформаційного ресурсу, наприклад, процедури, що зберігаються в базах даних. До другого шару належать інформаційні об'єкти ресурсу, наприклад, бази даних.

Для визначення проектних рішень back end ІС важливою умовою є визначення базових платформ.

### **Розгляньмо типові базові платформи.**

Операційними середовищами для серверів ІС здебільша є операційні системи Windows та Linux. Платформами для прикладного програмування компонентів ІС є засоби та мови програмування від фірми Microsoft (.Net, ASP, VB, C++, C# тощо), а також засоби й технології із застосуванням мови програмування Java.

Базовою платформою для клієнтів ІС є операційна система Windows (зокрема Windows 10), з браузером Google Chrome та офісним пакетом MS Office.

Останнім часом як базові платформи клієнта підтримуються операційні системи мобільних пристроїв і планшетних комп'ютерів, (IOS, Android тощо). Відповідно до вибору базової платформи існують відповідні методи, моделі, засоби розробки прикладних застосунків. У поширеному типовому рішенні щодо серверів застосунків як операційного середовища вибирається ОС Windows. Застосунки, що мають порівняльно просту логіку та не потребують спеціальних системних вимог щодо організації транзакцій, асинхронної взаємодії та ін., можуть бути реалізовані в середовищі Web-серверу.

Для реалізації прикладних серверних компонентів із застосуванням Javateхнологій використовується сервер застосунків JBoss. Для реалізації прикладних серверних компонентів із застосуванням технологій та засобів від Microsoft застосовується IIS з відповідними моделями і технологіями [19].

Типовим рішенням для реалізації функціоналу ІС є побудова прикладних застосунків на основі шаблонів проектування [11]. Шаблон проектування описує типові рішення для певного класу проблем чи функцій,

які властиві прикладним застосункам. До таких рішень належать рішення, що безпосередньо не пов'язані з основною функціональністю, але характерні майже для кожного застосунку, наприклад, реалізація системних функцій, організація взаємодії компонентів, доступу до даних тощо.

Для кожного шаблону проектування існують його модель, визначення, функціональність, структура та правила взаємодії. Для багатьох шаблонів проектування існують реалізації, які можуть використовуватись у різних застосунках.

Сукупність шаблонів проектування та їх реалізацій формують каркас (англ. framework) застосунку. Такий каркас є основою для прикладного програмування, тобто інтеграція його з бізнес-об'єктами забезпечує реалізацію застосунку. При цьому каркас є елементом, який може повторно використовуватись для різних застосунків, які відповідають принципам і правилам визначеної архітектури.

### **РОЗДІЛ 3. РОЗРОБКА ДОВІДНИКА СПІВРОБІТНИКІВ НАВЧАЛЬНО- НАУКОВОГО ІНСТИТУТУ ФІЗИКИ, МАТЕМАТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НА ЗАСАДАХ WEB-ТЕХНОЛОГІЙ**

#### **3.1. Структура та опис документообігу навчально-наукового інституту фізики, математики та інформаційних технологій**

Дослідження підприємства направлено на виявлення можливостей підвищення ефективності управління підприємством на базі використання засобів обчислювальної техніки і сучасних економіко-математичних методів, а так само на збільшення обсягу та підвищення якості послуг, що надаються при існуючих технічних засобах [1].

Будь-яка організація вирішує певний комплекс поставлених перед ним завдань. Кожне завдання розбивається між відділами організації для її вирішення. Саме тому сучасні організації складаються з декількох відділів, які взаємодіють між собою в процесі виконання роботи. Навчально-науковий інститут фізики, математики та інформаційних технологій, як організація, так само складається з підрозділів, що відповідають за виконання певного завдання.

Виходячи з цього, можна виділити організаційну структуру, яка включає директорат і кафедри.

Директорат надає великий спектр послуг. В ННІФМІТ здійснюється організація навчально-виховного процесу, складання штатного і навчального розкладу. Послуги, що надаються директоратом інституту повинні відповідати потребам співробітників кафедри.

#### **Опис документообігу**

Рішення задач управління передбачає інтенсивний обмін інформацією між усіма учасниками системи, як між людьми, так і між людьми і персональними комп'ютерами (ПК). Можливі два види обміну інформацією - документований і не документований.

Документований - це обмін документами, головним чином на папері, підготовленими і заповненими людьми, або ПК у вигляді роздруківок.

Недокументований обмін інформацією може бути використаний для: негайного отримання відомостей від особи, яка є їх джерелом; усній передачі розпоряджень; підтвердження виконання отриманих наказів або команд; повідомлення про ситуації, які потребують невідкладного втручання тощо. Його перевагою є швидкість отримання інформації, відсутність обмежень на зміст питань і відповідей, можливість введення будь-яких уточнень і деталізації. Основним недоліком недокументованих повідомлень є відсутність правової та юридичної відповідальності джерела інформації за достовірність і точність виданих відомостей, неможливість подальшої перевірки змісту раніше надісланого повідомлення. Тому у випадках, пов'язаних з матеріальною, фінансовою або іншою підвищеною відповідальністю, використовують тільки документовані повідомлення [1].

Основні типи документів з точки зору відділів їх ведення класифікуються на: вхідні, вихідні та внутрішні. До внутрішніх можна віднести організаційно-розпорядчу документацію. В системі організаційно-розпорядчої документації виділені три групи документів: організаційна, в яку входять статuti, інструкції, правила; розпорядча - постанови, розпорядження, накази, рішення; довідково-інформаційна - листи, довідки, протоколи, акти.

В процесі виконання об'єднання потоків вхідних і вихідних документів формуються типові групи документів: договірні документи, бухгалтерські документи, документи підтримки і впровадження нових технологічних процесів, звітні документи.

В сучасних системах управління є програмне забезпечення, що дозволяє автоматизувати створення форми документа за мінімальною інформацією, що повідомляється користувачем.

Елементами документообігу є угоди на працевлаштування співробітника ННІФМІТ і бухгалтерська документація, до якої можна віднести відомості про заробітну плату співробітників ННІФМІТ з урахуванням науково-педагогічного стажу, доповідні, службові записки, заяви, супровідні документи на матеріали і т.п.

Угода на працевлаштування співробітника ННІФМІТ укладається між навчальним закладом і співробітником, що працює на кафедрі, на певний термін з виконанням перелічених в ній вимог.

Угода містить такі дані:

- ✓ дані про ННІФМІТ;
- ✓ прізвище, ім'я, по батькові директора ННІФМІТ;
- ✓ дата укладення договору;
- ✓ прізвище, ім'я, по батькові співробітника ННІФМІТ;
- ✓ адреса і телефон співробітника інституту;
- ✓ паспортні дані співробітника (серія, номер паспорта);
- ✓ термін дії договору.

Договір також містить інформацію про обов'язки ННІФМІТ, обов'язки співробітника ННІФМІТ, про відповідальність сторін, що укладають договір.

У документообіг включаються різні внутрішні документи, такі як розпорядження, накази, інструкції, правила, протоколи, довідки і т.д.

Бухгалтерські звіти і звіти про діяльність ННІФМІТ, що подаються вищим організаціям, є вихідними документами.

Всі відділи взаємодіють між собою: документація формується на кафедрах, в директораті, а потім стікається до відділу управління якістю освіти і бухгалтерію університету. Виробнича документація складається співробітниками ННІФМІТ, обробляється керівниками, і потім передається керівній ланці, що знаходиться вище за рівнем управління. Документація являє собою оформлену на папері інформацію, необхідну для складання звітів про діяльність кафедр, директорату та інституту в цілому. Ці звіти можуть бути використані директором інституту для прийняття управлінських рішень.

### **Інформаційні потреби користувачів**

Основними користувачами програмного комплексу, призначеного організувати оперативне отримання інформації про співробітників навчально-наукового інституту фізики, математики та інформаційних технологій, є директорат, завідувачі, співробітники кафедр.



Дані, необхідні для формування потоків часто використовуваних запитів: відомості про структурні підрозділи інституту, відомості про співробітників кафедр і директорату.

Регламентовані запити:

- ✓ пошук фізичної особи за різними критеріями;
- ✓ пошук співробітника кафедри за різними критеріями;
- ✓ пошук співробітника директорату за різними критеріями;
- ✓ список співробітників кафедри;
- ✓ кількісний склад кафедр;
- ✓ науковий склад кафедри.

Інші запити до бази даних є непередбачуваними.

### **3.2. Основні об'єкти предметної області**

Виходячи з аналізу предметної області Навчально-наукового інституту фізики, математики та інформаційних технологій та потреб користувачів, можна визначити основні об'єкти предметної області: фізичні особи, співробітники кафедр, директорат, які представлені в таблиці 3.1, і їх атрибути, наведені в таблиці 3.2.

Основний потік запитів буде припадати на ці дані. Об'єкти предметної області мають властивості (атрибути) - характеристики даного поняття предметної області.

Таблиця 3.1.

#### **Список основных об`ектов предметной области**

<b>Найменування об'єкту</b>	<b>Короткий опис</b>
1 Фізичні особи	Дані про фізичні особи
2 Співробітники кафедр	Дані про співробітників кафедри
3 Директорат	Дані про співробітників директората

Таблиця 3.2.

**Атрибути основних об'єктів предметної області**

<b>Найменування об'єкту</b>	<b>Найменування атрибуту</b>	<b>Короткий опис</b>
1 Фізичні особи	1 Код фізичної особи	Унікальний код фізичної особи
	2 Прізвище	Прізвище фізичної особи
	3 Ім'я	Ім'я фізичної особи
	4 По батькові	По батькові фізичної особи
	5 Дата народження	Число, місяць, рік народження фізичної особи
	6 Адреса	Адреса проживання фізичної особи
	7 Телефон домашній	Номер домашнього телефону фізичної особи
	8 Телефон мобільний	Номер мобільного телефону фізичної особи
2 Співробітники кафедр	1 Код співробітника	Унікальний код співробітника кафедри
	2 Аудиторія	Відомості про аудиторію
	3 Телефон внутрішній	Номер внутрішнього телефону співробітника кафедри
	4 Телефон міський	Номер міського телефону співробітника кафедри

	5 Додаткові функціональні обов'язки	Додаткові функціональні обов'язки співробітника кафедри
3 Директорат	1 Код співробітника	Унікальний код співробітника
	2 Аудиторія	Відомості про аудиторію
	3 Телефон внутрішній	Номер внутрішнього телефону
	4 Телефон міський	Номер міського телефону

На основі аналізу інформаційних запитів виявляються зв'язки між основними об'єктами, охарактеризовані і представлені в таблиці 3.3.

Таблиця 3.3.

#### Зв'язки між основними об'єктами предметної області

Найменування зв'язку	Об'єкти, які беруть участь в зв'язку	Короткий опис
1 Працюють в	Директорат и Фізичні особи	Дані про фізичних осіб, які відносяться до співробітників директорату
2 Є	Співробітники кафедр та Фізичні особи	Дані про фізичних осіб, які відносяться до співробітників кафедр

Ухвалення рішення про створення реляційної бази даних, а також аналіз визначених вище об'єктів і атрибутів, вимагають виконання нормалізації даних.

### 3.3. Нормалізація даних

Реляційна база даних повинна бути піддана процедурі нормалізації. Нормалізована база даних не схильна до дефектів поновлення, додавання, видалення. Процес нормалізації має на меті усунення надмірності даних і полягає у приведенні до третьої нормальної форми Бойса-Кодда (3НФ) [7].

Так, нормалізація - це розбиття таблиці на дві або більше, яким властиві кращі властивості при введенні, зміні і видалення даних. Остаточна мета нормалізації зводиться до отримання такого проекту бази даних, в якому кожен факт з'являється лише в одному місці, тобто виключена надмірність інформації. Це робиться не стільки з метою економії пам'яті, скільки для виключення можливої суперечливості збережених даних.

При проектуванні бази даних адресної книги співробітників Навчально-наукового інституту фізики, математики та інформаційних технологій були дотримані принципи нормалізації:

- будь-яка не ключове поле кожної таблиці функціонально залежить тільки від первинного ключа, а не від його частини (якщо ключ складовою);
- немає функціональної залежності не ключового поля від іншого не ключового поля.

У процесі нормалізації було виділено 7 відносин (таблиць бази даних адресної книги співробітників), які відповідають ЗНФ: «Фізичні особи», «Кафедри», «Посади кафедр», «Наукові звання», «Співробітники кафедр», «Директорат», «Посади директорату».

Однак проектування реляційної бази даних тільки в термінах відносин на основі розглянутого механізму нормалізації, створення вручну графічної структури системи, перевірка її на повноту і несуперечність, відстеження версій і виконання модифікації часто представляє для проектувальника складний і незручний процес. Створення концептуальної і фізичної моделей, а потім використання для створення і модифікації структури бази даних за допомогою сучасних CASE-засобів дозволяє подолати обмеження реляційної моделі і забезпечити потребу проектувальників баз даних в більш зручних і потужних засобах моделювання предметної області [4].

### **3.4. Розробка концептуальної моделі структури даних**

Проектування баз даних зазвичай виконується не в термінах реляційної моделі, а з використанням концептуального моделювання баз даних на основі

семантичних моделей, що відображають значення реальних сутностей і відносин.

Мета концептуального моделювання - забезпечення найбільш природних для людини способів збору і представлення тієї інформації, яку передбачається зберігати в створюваній базі даних. Тому концептуальну модель даних намагаються будувати за аналогією з природною мовою.

Основними конструктивними елементами семантичних моделей є сутності, зв'язки між ними і їх властивості (атрибути). Однією з найбільш популярних концептуальних семантичних моделей є модель «сутність-зв'язок» («Entity-Relation») - ER-модель.

Сутність - будь-який значимий об'єкт предметної області, інформацію про який необхідно зберігати в базі даних.

Атрибут - поійменована характеристика сутності. Його ім'я має бути унікальним для конкретного типу сутності, але може бути однаковим для різного типу сутностей. Атрибути використовуються для визначення того, яка інформація повинна бути зібрана про сутність.

Для моделювання структури даних використовуються ER-діаграми, які в наочній формі представляють зв'язку між сутностями. Відповідно до цього ER-діаграми набули поширення в CASE-системах, що підтримують автоматизоване проектування реляційних баз даних.

Одними з найбільш популярних програмних продуктів в цій області є ERwin компанії Platinum і PowerDesigner компанії Sybase.

З урахуванням проведеного аналізу предметної області та нормалізації даних засобами програми PowerDesigner побудована концептуальна модель у формі діаграми "сутність-зв'язок", представлена в додатку А.

Сутність «Фізичні особи» (атрибути: Код особи, Прізвище, Ім'я, По батькові, Дата народження, адресу, телефон домашній, Телефон мобільний) містить відомості про людей, які працюють в навчально-науковому інституті фізики, математики та інформаційних технологій.

Сутність «Кафедри» (атрибути: код кафедри, назва кафедри, скорочена назва кафедри, аудиторний фонд, випускаюєміє спеціальності) призначена для зберігання інформації про кафедри, що входять до складу навчально-наукового інституту фізики, математики та інформаційних технологій.

Сутність «Посади кафедр» (атрибути: код посади кафедри, назва посади, примітка) призначена для зберігання інформації про посади на кафедрах.

Сутність «Наукові звання» (атрибути: код звання, назва звання) містить інформацію про наукові звання.

Сутність «Співробітники кафедр» (атрибути: код співробітника, аудиторія, телефон внутрішній, телефон міський, додаткові функціональні обов'язки) призначена для зберігання інформації про співробітників кафедри.

Сутність «Посади директорату» (атрибути: код посади директорату, назва посади, примітка) містить інформацію про посади директорату.

Сутність «Директорат» (атрибути: код директорату, аудиторія, телефон внутрішній, телефон міський) призначена для зберігання інформації про директорат інституту.

Атрибути сутностей можуть бути обов'язковими для заповнення інформацією і необов'язковими.

Крім самих сутностей в концептуальній моделі відображають зв'язки між сутностями. Кожна зв'язок між сутностями характеризується ім'ям, типом (вказано в дужках) і класом приналежності.

Тип зв'язку (М: 1) "багато-до-одного" - кілька примірників об'єктів одного класу пов'язано з одним екземпляром іншого класу.

У розробленій концептуальній моделі предметної області всі зв'язки характеризуються типом (М: 1).

Клас приналежності визначає обов'язковість входження кожного примірника об'єктів в зв'язок. Розрізняють обов'язковий і необов'язковий клас приналежності. У першому випадку кожен екземпляр класу об'єктів обов'язково бере участь в зв'язку, в другому - допускаються екземпляри об'єктів, які не беруть участі в зв'язку. Обов'язковий клас приналежності на

моделі позначається вертикальною рисою, необов'язковий - порожнім колом (див. Додаток. 1).

У концептуальній моделі предметної області представлені наступні зв'язки між сутностями:

- «Працюють в»- «Директорат» і «Фізичні особи» (М: 1);
- «є»- «Співробітники кафедр» і «Фізичні особи»(М: 1);
- «Займають»- «Директорат» і «Посади директорату»(М: 1);
- «Працюють»- «Співробітники кафедр» і «Кафедри»(М: 1);
- «Вибираються на» - «Співробітники кафедр» і «Посади кафедри»(М: 1);
- «Мають» - «Співробітники кафедр» і «Наукові звання»(М: 1).

Таким чином, концептуальна модель предметної області є схемою з 7 сутностей і зв'язків між ними.

### **3.5. Архітектура клієнт-сервер**

По суті створюваний програмний комплекс для обліку рейсів і пасажирів, що купили квитки на рейси є інформаційною системою. Для її реалізації необхідно використати архітектуру клієнт-сервер, оскільки передбачається, що база даних зберігатиметься на сервері. Усі користувачі, запустивши програму-клієнта, працюватимуть з однією і тією ж базою даних, звертаючись за інформацією до неї по локальній комп'ютерній мережі.

Особливістю архітектури клієнт-сервер являється використання виділених серверів баз даних, що розуміють запити на мові структурованих запитів SQL (Structured Query Language).

Відмінна риса серверів БД – наявність довідників даних, в якому записана структура БД, обмеження цілісності даних, формати і навіть серверні процедури обробки даних за викликом або по подіях в програмі. Об'єктами розробки в таких додатках, окрім діалогу і логіки обробки являються, реляційна модель даних і набір SQL операторів для типових запитів до БД.

Більшість конфігурацій клієнт-сервер використовує двох вирівняну модель, в якій клієнт звертається до послуг сервера. Передбачається, що

діалогові компоненти розміщується на клієнтові, що дозволяє забезпечити графічний інтерфейс. Компоненти управління даними розміщуються на сервері. Дворівневе визначення архітектури клієнт-сервер використовує саме цей варіант: додаток працює у клієнта, а система управління базою даних(СУБД) - на сервері.

Така організація пред'являє найменші вимоги до сервера, тому вона краще масштабується, чим інші. Складні додатки, що викликають велику взаємодію з базою даних, можуть жорстко завантажити як клієнта, так і мережу. Результати SQL-запиту повинні повернутися клієнтові для обробки, оскільки там знаходиться логіка ухвалення рішення.

Для скорочення навантаження на мережу і спрощення адміністрування додатків на сервері, логіка рішень оформляється у вигляді процедур, що зберігаються, і поповнюється на сервері баз даних.

Процедура, що зберігається, – процедура з операторами SQL для доступу до бази даних, що викликається по імені з передачею необхідних параметрів і виконується на сервері.

Перевантаження процедур, що зберігаються, логікою може перевантажити сервер, що приведе до втрати продуктивності. Тому частину логіки додатка часто розміщують на стороні сервера, і частина - на стороні клієнта. Такі клієнт-серверні системи називаються системами з розділеною логікою, це дозволяє отримати збалансоване завантаження клієнтів і сервера, але затрудняє супровід додатка.

### **3.6. Вибір та обґрунтування СУБД для реалізації бази даних**

Багато відомих великих компаній такі незалежно від напрямку бізнесу, мають одно загальне: вони вибрали MySQL в якості ключового компонента їх інформаційних систем.

MySQL однаково добре застосовується і для управління ракетними системами, збору даних для аерокосмічних досліджень, зберігання і обробки даних біржі.



Додатки подібного роду мають багато загальних вимог: легкість використання і управління; продуктивність; масштабованість; переносимість; використання ресурсів; відновлення після збою.

MySQL розроблений з метою задовольняти усім цим вимогам. Перераховані характеристики MySQL також дуже добре підходять для робочих груп, відділів підприємства.

Здійснимо порівняльний аналіз MySQL з SQL-серверами Microsoft SQL Server і Sybase щодо архітектури і особливостей цих SQL-серверів.

Спочатку треба зробити одно зауваження: до випуску Microsoft SQL Server 6.0, Sybase SQL Server і Microsoft SQL Server були одним і тим же продуктом.

Тому у більшості випадків вони поведуться абсолютно однаково. З цієї причини, термін "SQL Server" відноситься і до Microsoft SQL Server і до Sybase SQL Server. Там, де ці продукти відрізняються, згадуються їх відповідні повні імена.

MySQL був створений групою співробітників Digital Equipment Corporation DEC, що бажали утілити в RDBMS ряд унікальних технологічних рішень, що забезпечують великі переваги в порівнянні з існуючими серверами баз даних.

MySQL продемонстрував цілий ряд технологічних нововведень. Серед них множинні покоління записів, автоматичне двофазне підтвердження транзакцій, зеркалювання бази даних, великі двійкові об'єкти [BLOBs], бітові індекси, багатовимірні масиви і повідомлення про події.

Більшість існуючих RDBMS не змогли відтворити або створити еквівалентні технології. Архітектура SQL Server використовує комбінацію сторінкових, індексних і табличних блокувань для забезпечення конкурентного доступу до даних і обмеження цілісності. SQL Server також підтримує двофазне підтвердження транзакцій, проте вимагає великої кількості коду для реалізації підтвердження або відкату. SQL Server забезпечує

зберігання даних типу BLOB, але у більше обмеженому і менш швидкодіючому варіанті, ніж це реалізовано в Sybase.

Для того, щоб гарантувати цілісність даних, архітектура SQL Server використовує механізм блокувань сторінок даних. Найбільш часта причина блокування - додавання запису або її модифікація. Крім того, при перегляді "читач" запису блокує сторінку, на якій цей запис знаходиться, і її суміжні сторінки, може привести до проблем продуктивності для користувачів, які потребують доступу до даних на заблокованих сторінках.

Розробник вимушений описувати складну обробку можливого виникнення конфліктів блокувань. Це збільшує складність додатка, і підвищує вартість розробки і супроводу.

Індекси SQL Server блокуються точно так, як і сторінки цих відповідних таблиць. У додатках, працюючих з великими об'ємами даних, це може сильно понизити продуктивність системи.

Для забезпечення цілісного представлення даних в Sybase або Microsoft SQL Server розробник повинен використати блокування таблиць. Блокування таблиці викликає повне блокування, що розділяється, для оновлення або виняткову.

У Microsoft SQL Server 6.5 механізм блокувань поліпшений в порівнянні з версією 6.0 і Sybase SQL Server підтримкою блокувань на рівні записів при вставці. Це збільшує продуктивність вставки записів, але ніяк не вирішує інші проблеми із сторінковими, індексними або табличними блокуваннями. Тому, незалежно від версії, оновлення даних в архітектурі SQL Server все одно вимагає табличних або сторінкових блокувань для забезпечення цілісності даних.

MySQL забезпечує оптимістичні блокування за допомогою Архітектури Многоверсионности Записів(Multi - Generational Architecture - [MGA]). Цей механізм створює оптимізовані версії для нових, видалених або оновлюваних записів, які видно тільки в контексті конкретної транзакції, що змінює дані. Реально, MySQL версионировать тільки змінювані стовпці (поля) шляхом

створення deltas. Це забезпечує максимальну продуктивність і мінімальні вимоги до дискового простору.

Замість того щоб писати код обробки сторінкових, індексних і табличних блокувань, розробник при використанні MySQL повинен обробляти тільки конфлікти оновлення з іншими транзакціями. Це означає значно менші витрати при розробці і супроводі для корпорацій, використовуючи MySQL.

Архітектура багатOVERсійності записів MySQL гарантує доступність даних на читання для будь-яких користувачів і у будь-який час. Клієнтський додаток ніколи не чекає доступності таблиць, записів або індексів, незалежно від числа користувачів в системі або тривалості і складності якої-небудь транзакції. Розробники, використовуючи MySQL, автоматично отримують максимум продуктивності додатків, безвідносно складності обробки даних.

Як і сімейство СУБД SQL Server, MySQL підтримує типи як CHAR так і VARCHAR. З точки зору клієнтського додатку вони виглядають так само, як і в SQL Server. Це забезпечує сумісність додатків. Усередині, MySQL забезпечує зберігання цих типів даних інакше, ніж SQL Server. У MySQL, дані CHAR і VARCHAR зберігаються однаково - кінцеві пропуски обрізуються, і тільки рядок фактичної довжини зберігається у базі даних. У разі VARCHAR, коли дані просяться з сервера, клієнтському додатку повертається значення змінної довжини.

У разі CHAR, MySQL доповнює рядок пропусками до довжини, вказаної в структурі таблиці, і повертає дані як рядок з фіксованою довжиною. MySQL використовує алгоритм стискування (RLE) для економії місця, займаного даними на диску, як для типу CHAR так і для VARCHAR.

Максимальна довжина поля типу VARCHAR в MySQL дорівнює 32K (таке ж обмеження довжини має і CHAR). Розробник може використати усю вигоду від VARCHAR без обмеження в 255 символів. Така можливість має велике значення для розробників, які хочуть здійснювати пошук або маніпулювати великими потоками тексту. Якщо розмір даних MEMO може

перевищити 32K, то тільки MySQL дозволяє ефективно використати тип BLOB з визначуваним розміром сегменту.

Оскільки спосіб зберігання CHAR і VARCHAR в MySQL ідентичний, розробник ніколи не піклується про вибір типу даних з урахуванням продуктивності. Замість цього розробник може заснувати свій вибір на тому, як видається відповідний тип даних в клієнтському додатку. Розробник також може не турбуватися про оптимальне зберігання даних і про витрати дискового простору, оскільки MySQL використовує алгоритми стискування строкових даних.

MySQL забезпечує унікальний тип даних Багаторозмірний Масив (Multi Dimensional Array [MDA]). Тип MDA не реалізований ні в одній іншій РСУБД. Тип MDA дозволяє розробникові зберігати масиви будь-якої довжини і до 16 вимірів.

Висока продуктивність і багате представлення даних, що забезпечуються багатовимірними масивами, дозволяють розробникам створювати рішення, неможливі при використанні інших СУБД.

Установка MySQL дуже проста. MySQL автоматично і динамічно розподіляє простір для установки. Це означає, що немає необхідності ні в попередньому розподілі дискового простору, ні в подальшому при активній роботі з базою даних. Окрім цього, завдяки механізму мноверсионности записів, в MySQL немає файлів протоколів транзакцій. Оскільки MySQL не вимагає модифікації ядра ОС, він захищений від проблем сумісності при оновленні ядра ОС.

Це дозволяє розробникові супроводжувати операційну систему не оглядаючись на працездатність СУБД. MySQL автоматично конфігурується і настроюється, і не вимагає ніякого втручання адміністратора в налаштування. Це максимально полегшує управління і супровід.

Відновлення бази даних MySQL відбувається автоматично без втручання адміністратора БД. При запуску MySQL він перевіряє БД на

наявність непідтверджених записів. При існуванні таких вони переводяться в скасований стан. Цей процес займає декілька секунд.

Ядро MySQL використовує менш 2Мб пам'яті. При установці на диску вимагається біля 8Мб, MySQL не вимагає пам'яті більше, ніж базова пам'ять для операційної системи. Він динамічно використовує ресурси диска і пам'яті без втручання адміністратора БД.

В результаті проведеного аналізу для реалізації БД була вибрана СУБД - MySQL [5].

### 3.7. Фізична модель бази даних

Концептуальна модель дозволяє зрозуміти суть створюваної інформаційної системи, але вона не підходить для створення безпосередньо структури бази даних. Для генерації структури бази даних необхідно перетворити концептуальну модель в фізичну [4].

Фізична модель БД визначає спосіб розміщення даних на носіях (пристроях зовнішньої пам'яті), а також спосіб і засоби організації ефективного доступу до них. Фізичне проектування бази даних нерозривно пов'язане з конкретною СУБД, для якої створюється модель. Фізична модель бази даних інституту інформаційних технологій, реалізована в термінах СУБД Interbase засобами Power Designer, представлена в додатку Б.

Таблиця FizLitsa містить відомості про співробітників ІІТ.

Таблиця 3.4.

**FizLitsa фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
FKod	INTEGER	4	Унікальне	Унікальний код фізичної особи
FFam	VARCHAR	51	Not null	Прізвище фізичної особи
FUmya	VARCHAR	31	Not null	Ім'я фізичної особи
FOtch	VARCHAR	31		По батькові фізичної особи

FDataRojd	DATE	8	Not null	Число, місяць, рік народження фізичної особи
FAdres	VARCHAR	101	Not null	Адреса проживання фізичної особи
FTelDom	VARCHAR	31		Номер домашнього телефону фізичної особи
FTelMob	VARCHAR	31	Not null	Номер мобільного телефону фізичної особи

Первинним ключем в таблиці FizLitsa є поле FKod - номер, який однозначно ідентифікує фізичну особу.

Таблиця 3.5.

#### **DoljnostuDurectorata фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
DDKod	INTEGER	4	Унікальне	Код посади директорату
DDNazvanue	VARCHAR	11	Not null	Назва посади
DDPrumechanue	VARCHAR	101		Примітки

Первинним ключем в таблиці DoljnostuDurectorata є поле DDKod - номер, який однозначно ідентифікує посаду директорату.

Таблиця 3.6.

#### **Durektorat фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
DKod	INTEGER	4	Унікальне	Унікальний номер співробітника
DDKod	INTEGER	4	Not null	Код посади директорату
FKod	INTEGER	4	Not null	Код фіз. особи
DAudutoruya	VARCHAR	21	Not null	Відомості про аудиторію

DTelVnytr	VARCHAR	31	Not null	Номер внутрішнього
DTelGorod	VARCHAR	31		Номер міського телефону

Первинним ключем в таблиці Durektorat є поле DKod - номер, який однозначно ідентифікує кожного співробітника директорату. Поля DDKod і FKod є зовнішніми ключами, за якими здійснюється зв'язок таблиці Durektorat з таблицями DoljnostuDurectorata і FizLitsa.

Таблиця 3.7.

#### **Kafedru фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
KKod	INTEGER	4	Унікальне	Унікальний код кафедри
KNazvanue	VARCHAR	101	Not null	назва кафедри
KNazvSokrash	VARCHAR	31		скорочена назва кафедри
KAydutFond	VARCHAR	31		аудиторний фонд
KVupSpets	VARCHAR	101		випускаються спеціальності

Первинним ключем в таблиці Kafedru є поле KKod - номер, який однозначно ідентифікує кафедру.

Таблиця 3.8.

#### **DoljnostuKafedr фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
DKKod	INTEGER	4	Унікальне	Унікальний код посади кафедри
DDNazvanue	VARCHAR	11	Not null	Назва посади
DDPrumechanue	VARCHAR	101		Примітка

Первинним ключем в таблиці DoljnostuKafedr є поле DKKod - номер, який однозначно ідентифікує посаду на кафедрі.

Таблиця 3.9.

**NaychZvaniya фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
NKod	INTEGER	4	Унікальне	Унікальний код звання
NNazvanue	VARCHAR	101	Not null	Назва звання

Первинним ключем в таблиці NaychZvaniya є поле NKod - номер, який однозначно ідентифікує наукове звання.

Таблиця 3.10.

**SotrKafedr фізичної моделі бази даних**

Найменування	Тип	Розмір	Обмеження	Коментарій
SKod	INTEGER	4	Унікальне	Унікальний код співробітника кафедри
FKod	INTEGER	4	Not null	Код фізичної особи
NKod	INTEGER	4		Код звання
KKod	INTEGER	4	Not null	Код кафедри
DKKod	INTEGER	4	Not null	Код посади кафедри
SAydtoruya	VARCHAR	31	Not null	Відомості про аудиторію
STelVnytr	VARCHAR	31		Номер внутрішнього телефону співробітника кафедри
STelGorod	VARCHAR	31		Номер міського телефону співробітника кафедри



SDopFynkObya z	VARCHAR	101		Додаткові функціональні обов'язки співробітника кафедри
-------------------	---------	-----	--	---

Первинним ключем в таблиці SotrKafedr є поле SKod - номер, який однозначно ідентифікує співробітника кафедри. Поля FKod, NKod, KKod, DKKod є зовнішніми ключами, за якими здійснюється зв'язок таблиці SotrKafedr з таблицями FizLitsa, NaychZvaniya, Kafedru, DoljnostuKafedr.

Кінцевим етапом проектування структури клієнт-серверної бази даних за допомогою автоматизованих засобів є створення структури БД у вигляді сценарію, текст якого містить команди, написані на мові структурованих запитів обраної СУБД. Так, засобами CASE-системи для генерації структури бази даних необхідно перетворити концептуальну модель в фізичну, а потім створити сценарій, який в подальшому запускається на виконання засобами сервера бази даних [4, 193-198].

Для автоматизації створення структури бази даних в дипломній роботі засобами програми Power Designer після створення фізичної моделі бази даних адресної книги співробітників ІТ отриманий сценарій бази даних, представлений в додатку В. В подальшому сценарій (скрипт БД на мові SQL) виконаний засобами СУБД InterBase в програмі управління роботою сервера IBConsole, в результаті чого отримано файл бази даних.

### **3.7. Обмеження цілісності даних**

Обмеження використовуються для підтримки цілісності даних при функціонуванні системи. СУБД повинна забезпечувати несуперечність даних заданим обмеженням при перекладі БД з одного стану в інший. Використання обмежень пов'язане також з адекватністю відображення предметної області за допомогою даних, які зберігаються в БД [5].

Для завдання посилальної і смислової цілісності бази даних в СУБД InterBase визначаються:

1. Ставлення підпорядкованості між таблицями БД шляхом визначення первинних ключів у батьківських і зовнішніх ключів у дочірніх таблиць.
2. Обмеження на значення окремих стовпців.
3. Тригери - підпрограми, які автоматично виконуються сервером до або (і) після події зміни запису в таблиці БД.
4. Генератори для створення і використання унікальних значень потрібних полів.
5. Збережені процедури, що є підпрограмами, які приймають і повертають параметри, а також здатні виконувати запити в БД.

Первинний (PRIMARY KEY) і зовнішній (FOREIGN KEY) ключі будуються для забезпечення посилювальної цілісності реляційно пов'язаних таблиць. Первинний ключ, крім цього, виконує функції підтримки унікальності своїх значень, що зумовлено його основним призначенням - однозначно характеризувати запис в таблиці [13].

У базі даних "Адресна книга співробітників інституту інформаційних технологій" кожна таблиця має первинний ключ. Для зовнішніх ключів в зв'язаних таблицях задано каскадне оновлення значень пов'язаних полів і накладено обмеження NOT NULL. Крім первинних і зовнішніх ключів, таблиці бази даних включають поля, обов'язкові для заповнення (обмеження NOT NULL) і необов'язкові (обмеження NULL).

### **3.8. Опис створення бази даних**

Опишемо основні дії по створенню бази даних в MySQL за допомогою phpMyAdmin. На першому кроці заходимо до phpMyAdmin, для входу використовуємо ім'я користувача root та порожній пароль (рис. 3.1)

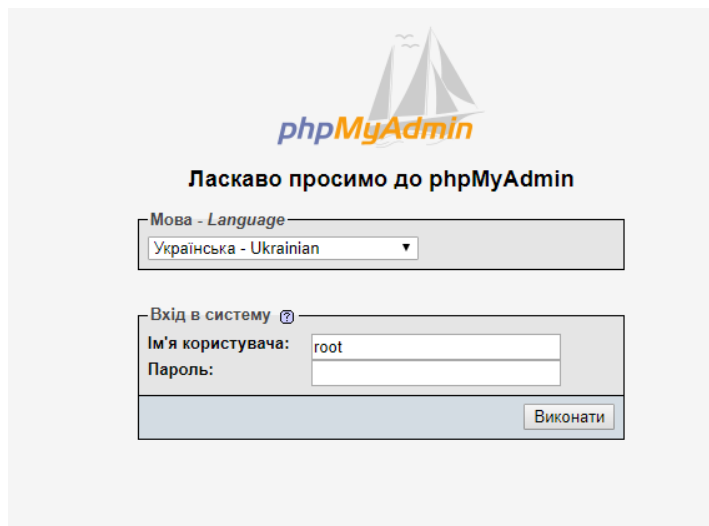


Рис. 3.1. Вхід в систему керування даними

Далі створюємо нову базу даних з іменем adressKniga (рис. 3.2)

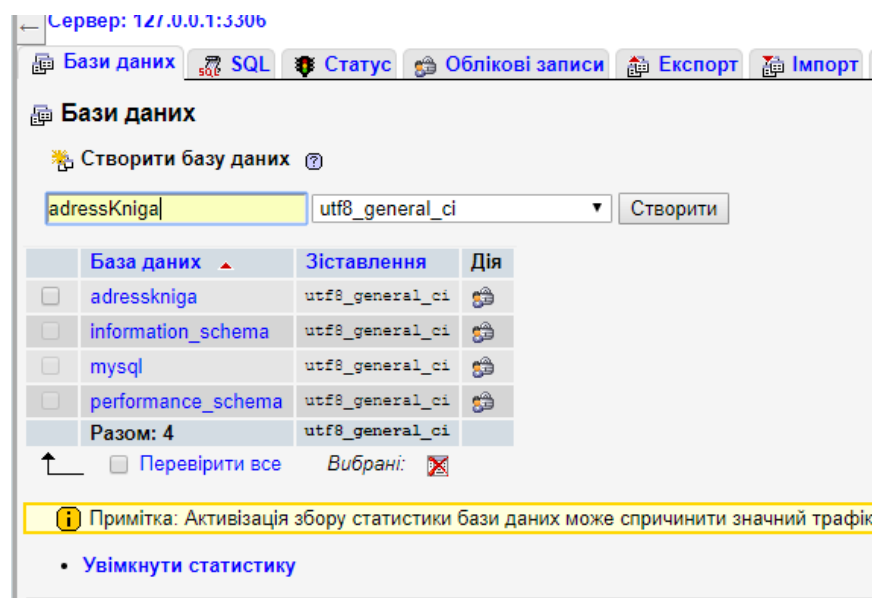


Рис. 3.2. Вхід в систему керування даними.

Розпочинаємо створення з довідників. Перша таблиця NaychZvaniya має два поля.

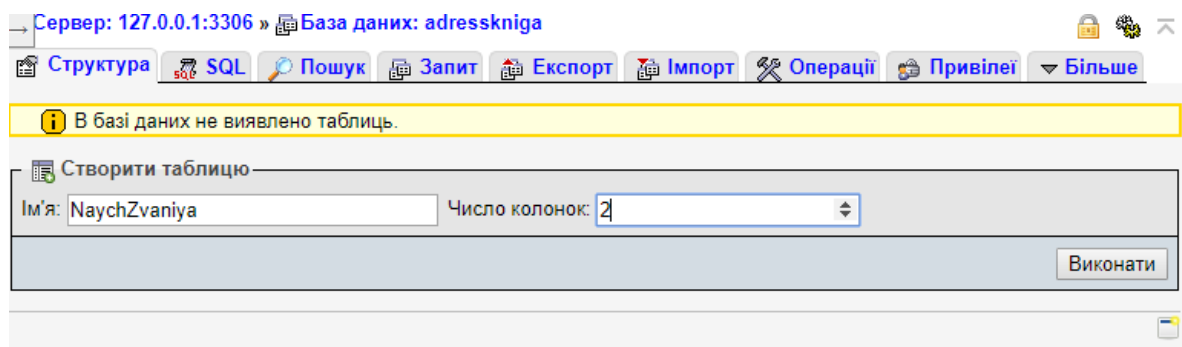


Рис. 3.3. Створення таблиці

Згідно фізичної моделі бази даних задаємо параметри кожного стовпця (рис. 3.4)

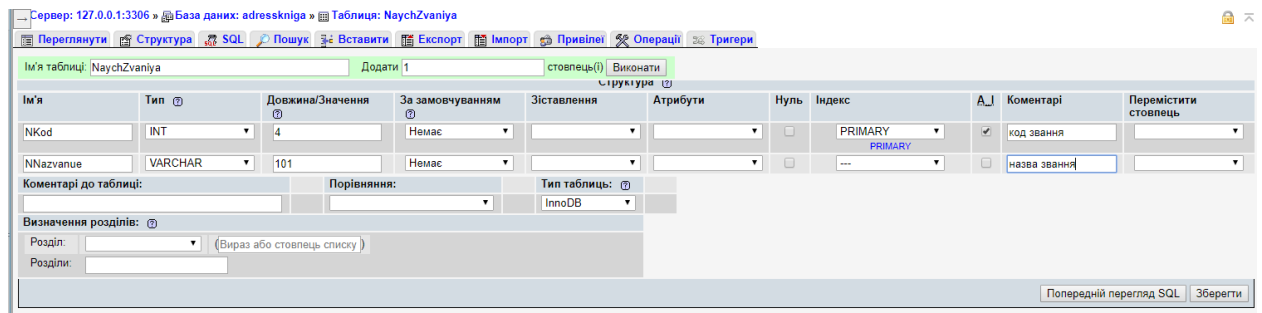


Рис. 3.4. Задання параметрів стовпців

Відповідний код на мові MySQL

```
CREATE TABLE `adresskniga`.`NaychZvaniya` ( `NKod` INT(4) NOT NULL AUTO_INCREMENT COMMENT 'код звання', `NNazvanue` VARCHAR(101) NOT NULL COMMENT 'назва звання', PRIMARY KEY (`NKod`)) ENGINE = InnoDB;
```

Додаємо основні дані в таблицю за допомогою команд MySQL

```
INSERT INTO `NaychZvaniya` (`NKod`, `NNazvanue`) VALUES (NULL, 'немає');
```

І так далі.

```
INSERT INTO `NaychZvaniya` (`NKod`, `NNazvanue`) VALUES (NULL, 'кандидат наук'), (NULL, 'професор');
```

Отримуємо дані з таблиці.

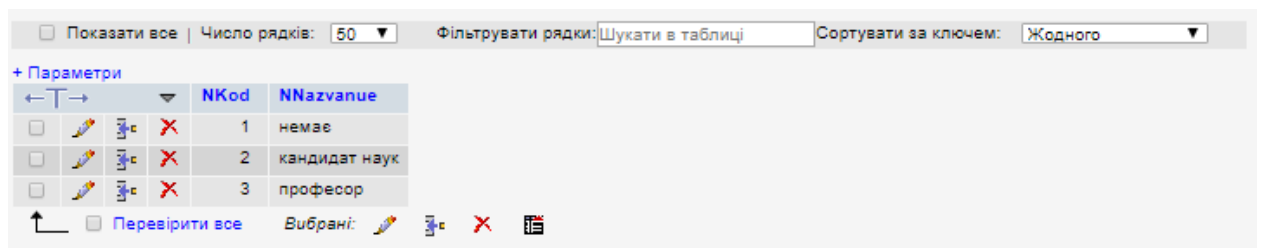


Рис.3.5. Дані з таблиці NaychZvaniya

Аналогічним чином створюємо DoljnostuKafedr, Kafedru, DoljnostuDurectorata, DoljnostuDurectorata.

Далі переходимо до основного довідника FizLitsa

Ім'я таблиці: FizLitsa		Додати 1		стовпець(і)		Виконати			
Структура									
Ім'я	Тип	Довжина/Значення	За замовчуванням	Зіставлення	Атрибути	Нуль	Індекс	A.I	Коментарі
FKod	INT	4	Немає			<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	Унікальний код фіз о
FFam	VARCHAR	51	Немає			<input type="checkbox"/>	---	<input type="checkbox"/>	Прізвище
FUmya	VARCHAR	31	Немає			<input type="checkbox"/>	---	<input type="checkbox"/>	Ім'я
FOtch	VARCHAR	31	Немає			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	Побатькові
FDataRojd	DATE		Немає			<input type="checkbox"/>	---	<input type="checkbox"/>	народження
FAdres	VARCHAR	101	Немає			<input type="checkbox"/>	---	<input type="checkbox"/>	адреса
FTelDom	VARCHAR	31	Немає			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	дом телефон
FTelMob	VARCHAR	31	Немає			<input type="checkbox"/>	---	<input type="checkbox"/>	моб телефон

Рис. 3.6. Створення таблиці FizLitsa

Вводимо відомості, щодо основних фізичних осіб університету.

Сервер: 127.0.0.1:3306 » База даних: adresskniga » Таблиця: fizlitsa									
Переглянути Структура SQL Пошук Вставити Експорт Імпорт Привілей Операції Тригери									
Показано рядки 0 - 27 (всього 28, Запит виконувався 0.0004 секунди.)									
SELECT * FROM `fizlitsa`									
Профілювання [Порядкове реда]									
Показати все Число рядків: 50 Фільтрувати рядки: Шукати в таблиці Сортувати за ключем: Жодного									
+ Параметри									
		FKod	FFam	FUmya	FOtch	FDataRojd	FAdres	FTelDom	FTelMob
		унікальний код фіз особи	Прізвище	Ім'я	Побатькові	народження	адреса	дом телефон	моб телефон
<input type="checkbox"/>		1	Донченко	В	Ю	0000-00-00	ifmit.s.2015@gmail.com	642	+380 50428-95-81
<input type="checkbox"/>		2	Жукова	В	М	0000-00-00	v.zhukova.lnu@gmail.com	642	+380 66771-11-26
<input type="checkbox"/>		3	Кіреєв	І	Ю	0000-00-00	igkireev@gmail.com	642	+380 66121-22-00
<input type="checkbox"/>		4	Козуб	Г	О	0000-00-00	galina14kga@gmail.com	642	+380 50673-46-73
<input type="checkbox"/>		5	Могильний	Г	А	0000-00-00	g.mogilniy@gmail.com	642	+380 66173-38-50
<input type="checkbox"/>		6	Тихонов	Ю	Л	0000-00-00	t2003i@mail.ru	642	+380 66262-39-74
<input type="checkbox"/>		7	Переяславская	С	О	0000-00-00	pereyaslav9@gmail.com	642	+380 50213-56-46
<input type="checkbox"/>		8	Смагіна	О	О	0000-00-00	olga_smagina@mail.ru	642	+380 99162-06-42
<input type="checkbox"/>		9	Семенов	М	А	0000-00-00	semenoff.lnu@gmail.com	642	+380 95760-87-64
<input type="checkbox"/>		10	Жучок	А	В	0000-00-00	zhuchok_a@mail.ru	642	+380 95307-62-96

Рис. 3.7. Дані введені в таблицю FizLitsa

Аналогічним чином створюємо таблицю SotrKafedr, а після цього створюємо зв'язок між таблицями. Наприклад FKod зв'язок із таблицею FizLitsa рис. 3.8.

Сервер: 127.0.0.1:3306 » База даних: adresskniga » Таблиця: sotrkafedr									
<div>Переглянути</div> <div>Структура</div> <div>SQL</div> <div>Пошук</div> <div>Вставити</div> <div>Експорт</div> <div>Імпорт</div> <div>Привілей</div> <div>Операції</div> <div>Тригери</div>									
<div>Структура таблиці</div> <div>Вид відносин</div>									
Обмеження зовнішнього ключа									
<div>Дія</div> <div>Обмеження властивостей</div> <div>Стовпець</div> <div>Обмеження зовнішнього ключа (INNODB)</div>									
<div>Обмеження зовнішнього</div> <div>ON DELETE RESTRICT</div> <div>ON UPDATE CASCADE</div> <div>FKod</div> <div>База даних</div> <div>Таблиця</div> <div>Стовпець</div> <div>adresskniga</div> <div>fizlitsa</div> <div>FKod</div>									
<div>+ Додати обмеження</div>									
<div>Попередній перегляд SQL</div> <div>Зберегти</div>									

Рис. 3.8. Створення зв'язку між таблицями

Або на мові SQL приклад створення зв'язку із таблицею NaychZvaniya.

ALTER TABLE `sotrkafedr` ADD FOREIGN KEY (`KKod`) REFERENCES `naychzvaniya`(`NKod`) ON DELETE RESTRICT ON UPDATE RESTRICT;

На рис. 3.9 ми бачимо результат створення зв'язків між таблицями.

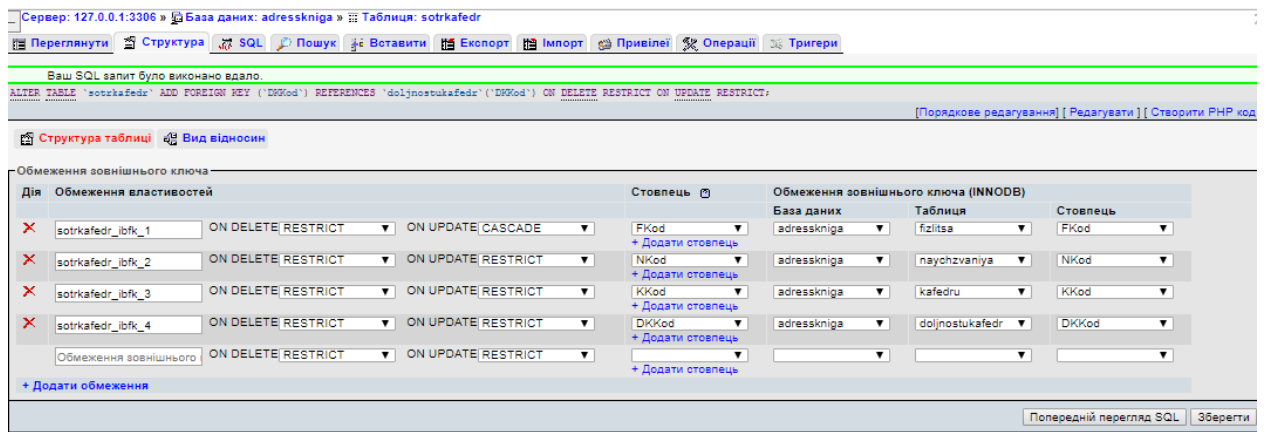


Рис. 3.9 Результат створення зв'язків між таблицями

Аналогічним чином створюємо зв'язки між іншими таблицями.

### 3.9. Опис створення програмного додатку на мові PHP

Після створення та завантаження бази даних переходимо до створення програмного засобу на мові PHP.

Головний файл через який ми проводимо доступ до будь якої частини функціональності це index.php (Додаток 3 Лістинг файлу index.php) з початку в ньому ми завантажуюємо та створюємо глобальні бібліотеки model.php в яких зберігається function open\_database\_connection() в якій відкривається зв'язок із базою даних adresskniga та зберігаються функції, що виконують запити на зчитування даних, а саме get\_all() дані про всіх працівників, get\_all\_by\_kafedra(\$id) видає відомості про всіх працівників певної кафедри, get\_all\_kafedrus() список всіх кафедр інституту.

controllers.php містить контролери, які описують дії додатку в залежності від запитів, які надійшли на його адресу.

Також в файлі реалізована система роутингу сайту - в залежності від того який запит надійшов на сайт, викликається той, чи інший контролер.

Опишемо більш детально роботу сайту.

При заході на адресу index.php система роунтингу, викличе функцію index\_action(), ця функція описана в файлі controllers.php

```
function index_action()
```

```
{
```

```
    $title = "Адресна книга НН ІФМФІТ";
```

```

$content = " <h1>Адресна книга НН ІФМІТ</h1>
<ul>
<li><a href='/index.php/kafedras'>Працівники по кафедрам</a></li>
<li><a href='/index.php/all'>Усі працівники</a> </li>
</ul>";
require 'templates/layout.php';
}

```

В цій функції ми задаємо дві змінні, які будуть використовуватися у подальшому \$title, яка задає заголовок сторінки, які відображаються у заголовку сторінки; \$content в якому зберігається основний контент сторінки.

А далі викликається файл templates/layout.php, який задає зовнішній вигляд будь якої сторінки. (Додаток 4. Лістинг layout.php). В ньому відображається стандартна розмітка html-сторінки в тезі <head> включається title із сформований у відповідному контролері змінної \$title, підключається таблиця стилей design.css.

В тезі body формується заголовок сторінки <header> та нижня частина <footer>, вони є незмінними для будь якої сторінки, а між ними виводиться основний контент, який зберігається у змінній \$content.

На рисунку 3.10 зображено зовнішній вид головної сторінки.

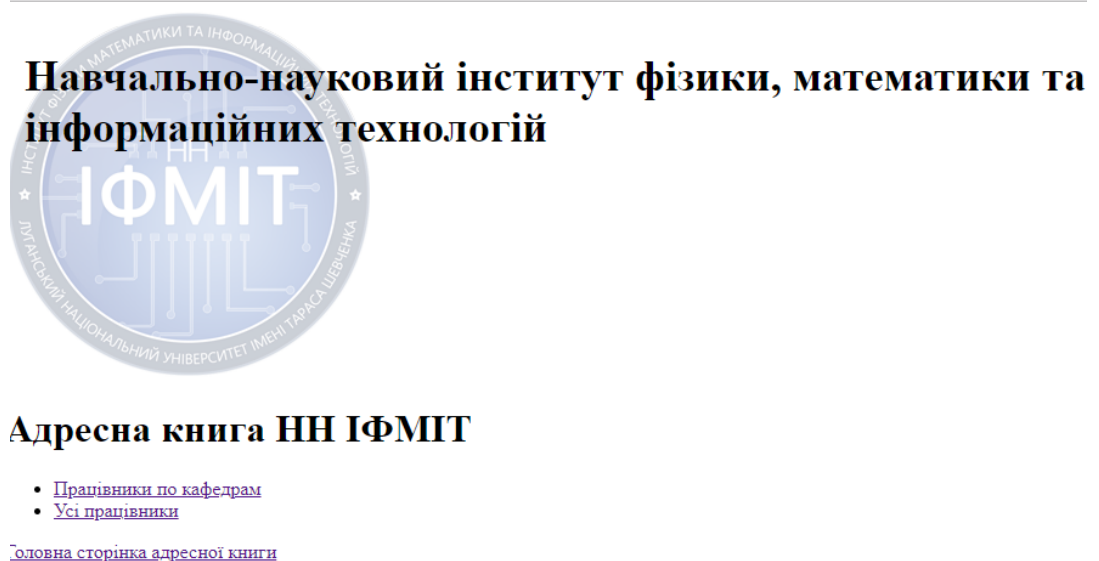


Рис. 3.10. Головна сторінка адресної книги НН ІФМІТ

Якщо ми klikнемо по пункту «Працівники по кафедрам» внутрішній шлях `/index.php/kafedras`, то згідно системи роутингу ми зробимо виклик `kafedras_action()`, опис цієї функції записано у файлі `controllers.php` і складається з одного рядка `require 'templates/kafedras.php'`;

Лістинг цього файлу приведемо у додатку 5 Лістинг файлу `templates/kafedras.php`. З початку визивається функція `$posts = get_all_kafedrusr()`, яка зберігається у файлі `model.php`. Розглянемо детально кожен рядок цієї функції

Створюємо зв'язок з базою даних

```
$link = open_database_connection();
```

Викликаємо SQL код над базою даних вибираючи із таблиці `kafedru` елементи `KKod`, `KNazvanue`, `KNazvSokrash`, елементи якої ми записуємо у змінну `$result`

```
$result = $link->query('SELECT KKod, KNazvanue, KNazvSokrash  
FROM kafedru');
```

Створюємо масив, та записуємо в нього результати виконання вибірки із бази даних

```
$kafedrusr = array();  
while ($row = $result->fetch(PDO::FETCH_ASSOC)) {  
    $kafedrusr[] = $row; }  
закриваємо з'єднання із базою даних
```

```
close_database_connection($link);
```

повертаємося у базу даних

```
return $kafedrusr;
```

Далі ми повертаємося до файлу `kafedras.php` в ньому ми задаємо змінну `$title`, та переходимо до створення `$content`, ми називаємо `ob_start()`, щоб запустити вихідний буфер. В ньому ми формуємо за допомогою конструкції `foreach` нумерований список, який представляє собою набір посилань. Зовнішній вигляд сторінки представлено на рис. 3.11.





## Навчально-науковий інститут фізики, математики та інформаційних технологій

### Список кафедр НН ІФМІТ

- [Кафедра інформаційних технологій та систем коротка назва \(ІТС\).](#)
- [Кафедра фізико-технічних систем та інформатики коротка назва \(ФТІ\).](#)
- [Кафедра документознавства та інформаційної діяльності коротка назва \(ДІД\).](#)
- [Кафедра алгебри та системного аналізу коротка назва \(АСА\).](#)

[Головна сторінка адресної книги](#)

Рис. 3.11 Сторінка за адресом index.php/kafedras

Далі якщо користувач клікне по кафедрі ІТС, тобто він перейде за адресом index.php/kafedra?id=1, то згідно системи роунтингу elseif ('/index.php/kafedra' === \$uri && isset(\$\_GET['id'])), ми зробимо виклик функції kafedraid\_action(\$id), яка в свою чергу виконає файл listKafedra.php дивимось додаток 6 Лістинг файлу templates/listKafedra.php. Головна робота по отриманню інформації з бази даних виконується функцією get\_all\_by\_kafedra(\$id) із файлу model.php в ній створюється зв'язок із базою даних, та виконується запит:

```
SELECT t2.FFam,t2.FUmya,t2.FOtch,t2.FTelMob,t3.KNazvanue FROM  
sotrkafedr as t1 join fizlitsa as t2 on t1.FKod = t2.FKod join kafedru as t3 on  
t1.KKod = t3.KKod WHERE t1.KKod =:id;
```

В ньому із таблиць sotrkafedr, fizlitsa, kafedru ми вибираємо дані, причому :id ми замінюємо на обраний користувачем за допомогою конструкції \$statement->bindValue(':id', \$id, PDO::PARAM\_INT);, а інші конструкції аналогічні функції get\_all\_kafedrus(), яка розглянута вище. Вся інформація повертається до функції listKafedar.php, в якій формується таблиця з відомостями про співробітників відповідної кафедри. Вигляд сторінки на рисунку 3.12.



**Навчально-науковий інститут фізики, математики та інформаційних технологій**

**Кафедра інформаційних технологій та систем список працівників**

Прізвище	Імя	Побатькові	Мобільний телефон
Матівський	В	В	+380 99353-81-36
Могильний	Г	А	+380 66173-38-50
Донченко	В	Ю	+380 50428-95-81
Переяславская	С	О	+380 50213-56-46
Семенов	М	А	+380 95760-87-64
Смагина	О	О	+380 99162-06-42
Тихонов	Ю	Л	+380 66262-39-74

[Головна сторінка адресної книги](#)

Рис. 3.12. Вигляд сторінки за адресом `index.php/kafedra?id=1`

## ВИСНОВКИ

У магістерській роботі було розглянуто дослідження технологій проектування інформаційних систем та розробка довідника співробітників навчально-наукового інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

У першому розділі в результаті огляду літературних джерел проаналізовано поняття та можливості інформаційних систем. Розглянуто принципи створення інформаційних систем. Виявлено особливості веб-орієнтованих інформаційних систем.

В другому розділі представлено загальну характеристику веб-технологій, а саме: подано базові поняття, розглянуто основні тенденції розвитку, проаналізовано веб-стандарти та вимоги до програмних платформ і веб-застосунків. Для розгляду особливостей побудови ІС на основі веб-технологій спершу аналізується загальна методологія створення ІС, зокрема методології RAD, RUP, UML. Далі визначаються проектні рішення щодо побудови ІС як інтернет/інтранет об'єкта.

У третьому розділі в процесі аналізу функціонування навчально-наукового інституту фізики, математики та інформаційних технологій були розглянуті предметна область, інформаційні потреби користувачів довідника. Визначено дані, що формують потоки часто виконуваних запитів, які необхідні співробітникам директорату, кафедрам інституту, а також виділені регламентовані запити до бази. Виконана нормалізація таблиць бази даних для адресної книги, спроектована інфологічна модель структури даних. Обґрунтовано вибір СУБД MySQL архітектури клієнт-сервер для реалізації бази даних створеної інформаційної системи. Проведено доталогочне проектування БД. Задано обмеження цілісності в СУБД MySQL на оновлення і видалення даних в зв'язаних таблицях бази даних. За допомогою CASE-системи Power Designer розроблені концептуальна і фізична моделі. Розроблено довідник співробітників інституту фізики, математики та інформаційних технологій на засадах Веб-технологій.

Додатки містять розроблені концептуальну та фізичну моделі, засобами CASE-системи PowerDesigner, а також лістинги найбільш важливих процедур програми з вказівкою всіх використовуваних компонентів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1) DimDim Software: Проектирование и разработка автоматизированных, информационных и аналитических систем. проектирование [Электронный ресурс]. – Режим доступа : <http://www.info-system.ru/>.
- 2) UML2.ru. Сообщество системных аналитиков [Электронный ресурс]. – Режим доступа : <http://www.uml2.ru/>.
- 3) [www.library-odeku.16mb.com](http://www.library-odeku.16mb.com)
- 4) Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. М: «Финансы и статистика», 1998.
- 5) Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М: «Финансы и статистика», 2006.
- 6) Гвоздева В. А. Основы построения автоматизированных информационных систем учебник М.: БИНОМ. Лаборатория знаний, 2008.
- 7) Глушаков С.В., Жакин И.А., Хачиров Т.С. Программирование Web-страниц. М.: Фолио, 2005. – 390 с.
- 8) Грекул В.И. и др. Проектирование информационных систем. Курс лекций. Интернет-Университет Информационных Технологий, М.2005.
- 9) Грищенко В. Н. Підхід до створення WEB-застосунків на компонентній основі // Проблеми програмування. – 2004. – № 4. – С. 13–24.
- 10) Грищенко В. Н. Типовые элементы компонентно-ориентированной разработки WEB-приложений // Кибернетика и системный анализ. – 2005. – №. – С. 163–175.
- 11) Гудман Д. JavaScript и DHTML: сборник рецептов. Для профессионалов. СПб.: Питер, 2004. – 528 с.

- 12) Дари К., Бринзаре Б., Черchez-Тоза Ф., Бусика М. AJAX и PHP. Разработка динамических веб-приложений. – СПб.: Символ-плюс, 2006. –336 с.
- 13) Изучаем HTML, XHTML и CSS. - Эрик Фримен, Элизабет Фримен.,2010 – 656 с.
- 14) Ипатова Э.Р. Методологии и технологии системного проектирования информационных систем, учебник М.: Флинта: МПСИ, 2008
- 15) Исаев Г.Н. Проектирование информационных системОмега-Л, 2013.–424
- 16) Коггзолл Дж. PHP5 Полное руководство.: - пер.с англ. – М.: Издательский дом «Вильямс», 2006 – 752 с.
- 17) Колисниченко Д. Н. Самоучитель PHP 5 – СПб.: Наука и Техника, 2007. – 640 с.
- 18) Ларман К. Применение UML и шаблонов проектирования: Введение в объектно-ориентированный анализ и проектирование: Учебное пособие: Пер. с англ. - М.: Вильямс, 2001. – 496 с.
- 19) Маклаков С.С. BPwin и Egwin. CASE- средства разработки информационных систем. М.: ДИАЛОГ-МИФИ., 2000.– 256с.
- 20) Марти Холл, Лэрри Браун. Программирование для Web. М.: Вильямс, 2002. – 1264 с.
- 21) Марченко А.В. Проектування інформаційних систем [електронний ресурс] / А. В. Марченко. – К., 2016. – Режим доступу: [http://elearning.sumdu.edu.ua/free\\_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20151030212747/content-20151030212747.pdf](http://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20151030212747/content-20151030212747.pdf) – Назва з екрану.
- 22) Маслов А.В. Проектирование информационных систем вэкономике: учебное пособие / Маслов А.В. – Томск: Изд-во Томского политехническогоуниверситета, 2008. – 216 с.
- 23) Мацяшек Л. А. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML / Л. А.

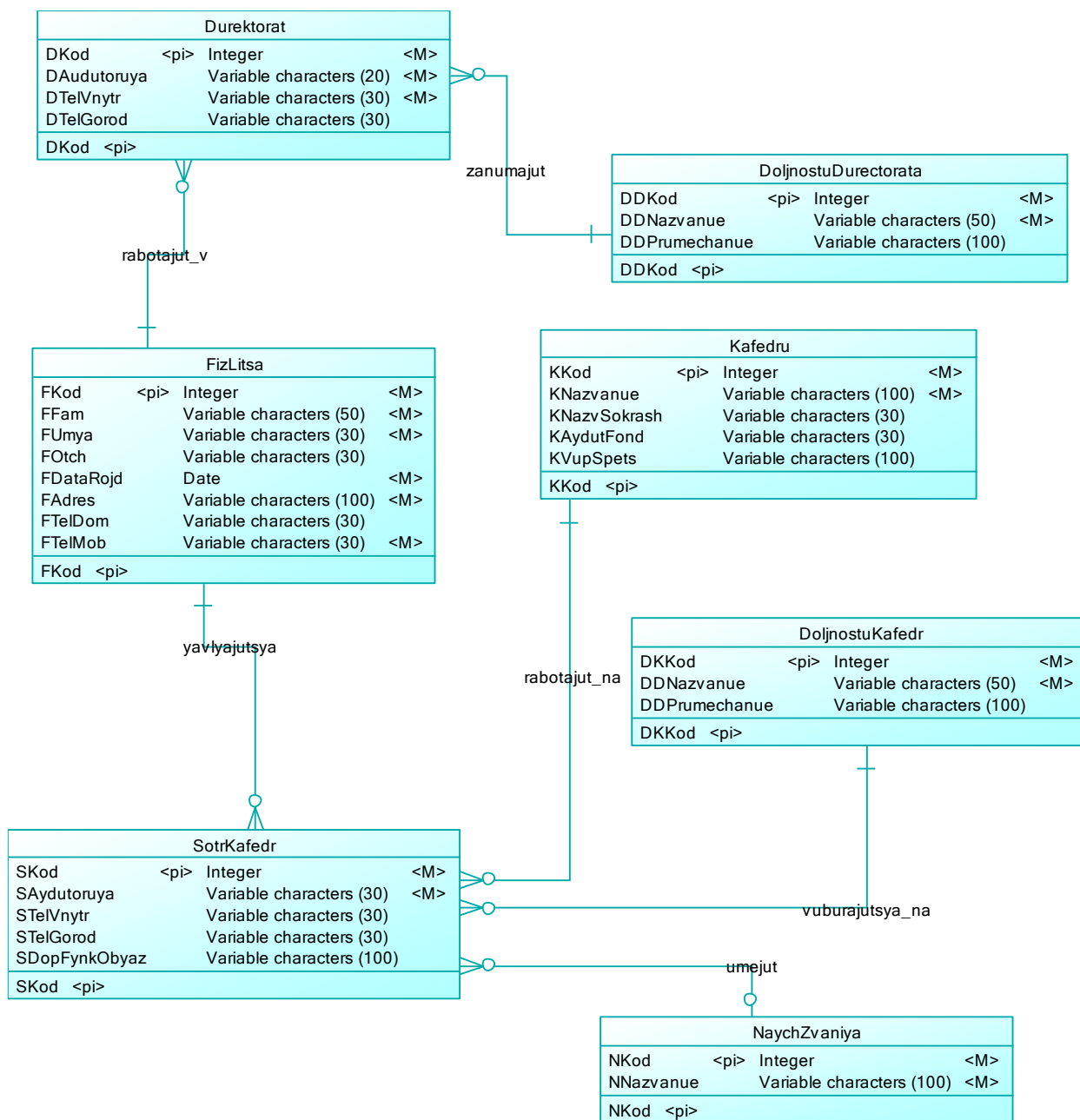
- Мацяшек ; пер. с англ. – М. : Издательский дом "Вильямс", 2002. – 432 с.
- 24) Моделирование бизнеса. Методология ARIS / М. Каменова, А. Громов, М. Ферапонтов, А. Шматолук. – М. : Весть-Мета Технология, 2001. – 328 с.
- 25) Муссиано Ч., Кеннеди Б Изучаем HTML, XHTML и CSS., 2011. – 752 с.
- 26) Недашківський О. Л. Планування та проектування інформаційних систем / О. Л. Недашківський. – К., 2014. – 215 с.
- 27) Объектно-ориентированный анализ и проектирование [Электронный ресурс]. – Режим доступа : <http://ood.asf.ru/>.
- 28) Основи теорії інформаційних систем: Лабораторний практикум для студентів напряму 6.050101 «Комп'ютерні науки» / уклад.: І. Е.Райчев, О.Г.Харченко. – К.: Видав. Нац. авіац. ун-ту «НАУ-друк», 2014. – 48 с.
- 29) Пасічник В.В., Литвин В.В., Шаховська Н.Б. Проектування інформаційних систем. Навчальний посібник (затв. МОН України) Львів: 2013.– 380 с.
- 30) Проектування автоматизованих інформаційних систем. Конспект лекцій для спеціальності 5.05010301 «Розробка програмного забезпечення». Модуль1 / Укл.: Блонський Л.А. – Львів, Вид - во ЛККЕП. 2011. – 27 с.
- 31) Проектування інформаційних систем: Посібник / За редакцією В. С. Пономаренка. – К.: Видавничий центр «Академія», 2002. – 488 с.
- 32) Титенко С. В. Інформаційно-логічні та архітектурні засади універсальних систем керування web-контентом/ С. В. Титенко // XII международная научная конференция имени Т. А. Таран «Интеллектуальный анализ информации ИАИ-2014», Киев, 14-16 мая 2014 г. : сб. тр./ гл. ред. С.В.Сирота. – К. : Просвіта, 2014. – С. 214-220.

- 33) Титенко С. В. Структурные основы онтологически-ориентированной системы управления информационно-учебным Web-контентом / С. В. Титенко // Управляющие системы и машины: информационные технологии: междунар. науч. журн. - 2012. - № 2. - С. 35-42. ISSN 0130-5395
- 34) Титенко, С. В. Моделювання спеціалізованих інформаційних об'єктів в універсальних системах керування Web-контентом / С. В. Титенко // Вісник Східноукраїнського національного університету імені Володимира Даля – 2012. – №8 (179). Ч.2. — С. 235-239.
- 35) Томашевський О.М. Інформаційні технології та моделювання бізнес-процесів: Навчальний посібник. / О.М.Томашевський, Цегелик М.Б., Вітер Г.Г., В.І.Дубук. К.: Центр учбової літератури, 2005. – 296 с.
- 36) Ушакова І. О. Проектування інформаційних систем: практикум / І. О. Ушакова. – Х. : ХНЕУ ім. С. Кузнеця, 2015. – 236 с.
- 37) Шаховська Н. Б., Литвин В. В. Проектування інформаційних систем: навчальний посібник. – Львів: «Магнолія-2006», 2011. – 380 с.

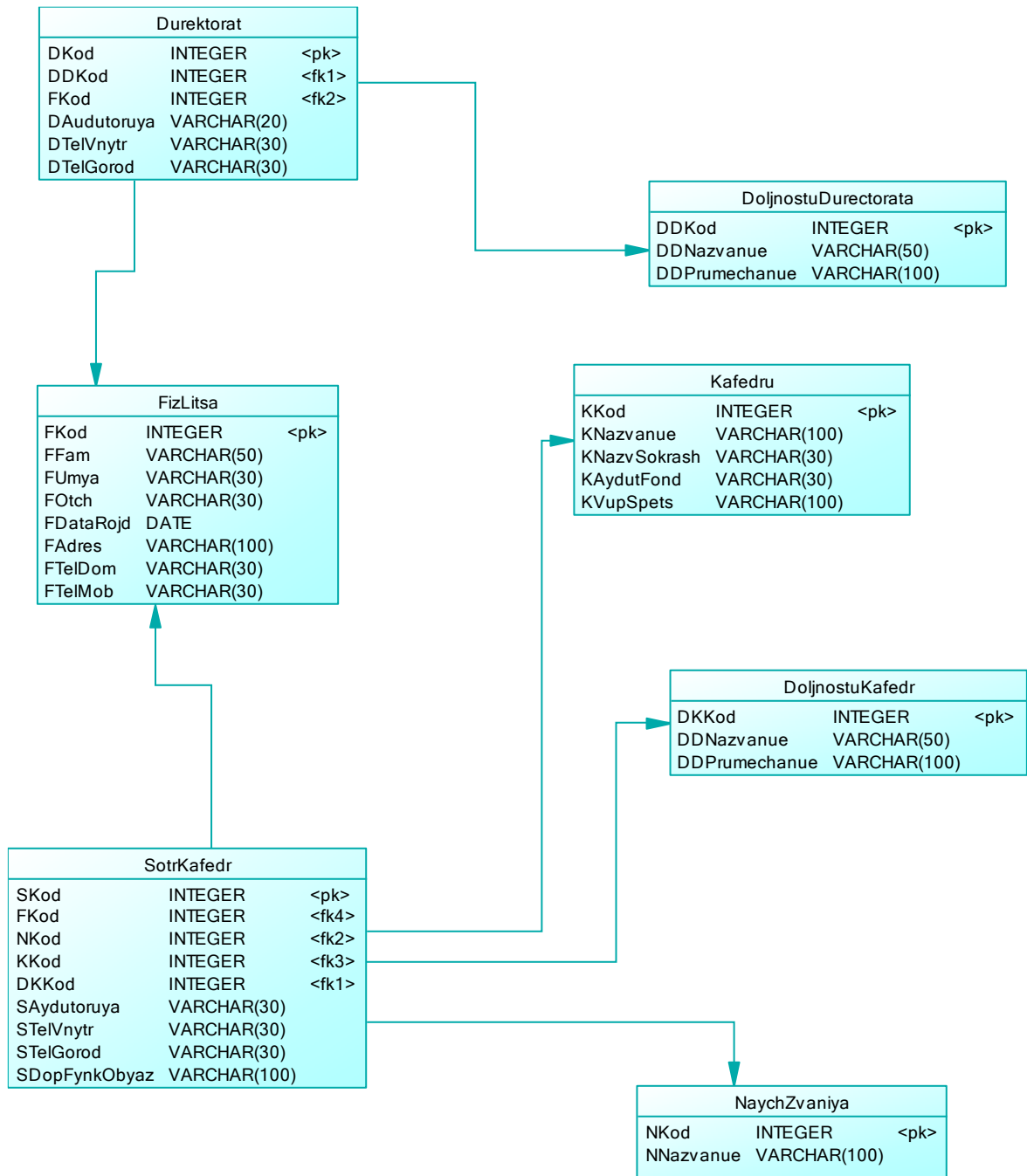


## ДОДАТКИ

### Додаток 1. Концептуальна модель бази даних



## Додаток 2. Фізична модель бази даних



### Додаток 3. Лістинг файлу index.php

```
<?php
//завантаження та ініціалізація глобальних бібліотек
require_once 'model.php';// для зв'язку з базою даних
require_once 'controllers.php';//для керування відображенням

//система роунтига
$uri = parse_url($_SERVER['REQUEST_URI'], PHP_URL_PATH);
if ('/index.php' === $uri) {
    index_action();
} elseif ('/index.php/kafedra' === $uri && isset($_GET['id'])) {
    kafedraid_action($_GET['id']);
} elseif ('/index.php/kafedras' === $uri ) {
    kafedras_action();
} elseif ('/index.php/all' === $uri ) {
    all_action();
} else {
    none_action();
}
```

#### Додаток 4. Лістинг файлу layout.php

```
<!-- templates/layout.php -->
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title><?= $title ?></title>
    <link rel="stylesheet" href="/templates/design.css">
</head>
<body>
<header class="container">
    
    <div class="topleft"><h1>Навчально-науковий інститут фізики,
математики та інформаційних технологій</h1></div>
</header>

<?= $content ?>

<footer>
    <a href="/index.php">Головна сторінка адресної книги</a>
</footer>
</body>
</html>
```

## Додаток 5. Лістинг файлу templates/kafedras.php

```
<?php
$posts = get_all_kafedrus();
$title = 'Список кафедр';
ob_start();
?>

<h1> Список кафедр НН ІФМІТ</h1>
<ul>
    <?php foreach ($posts as $post):?>
        <li>
            <a href="/index.php/kafedra?id=<?=$post['KKod']?>">
<?=$post['KNazvanue']?> коротка назва (<?=$post['KNazvSokrash']?> ).</a>
        </li>
    <?php endforeach; ?>
</ul>

<?php
$content = ob_get_clean();
include 'layout.php';
```

## Додаток 6. Лістинг файлу templates/listKafedra.php

```
<?php
$posts = get_all_by_kafedra($id);
$title = $posts[0]['KNazvanue'].' список працівників';
ob_start();
//print_r($posts)
?>

<h1> <?=$posts[0]['KNazvanue']?> список працівників </h1>
<table>
    <thead>
        <tr>
            <th> Прізвище</th>
            <th> Ім'я</th>
            <th> Побатькові</th>
            <th> Мобільний телефон</th>
        </tr>
    </thead>
    <?php foreach ($posts as $post):?>
        <tr>
            <td> <?=$post['FFam']?></td>
            <td> <?=$post['FUmya']?></td>
            <td> <?=$post['FOtch']?></td>
            <td> <?=$post['FTelMob']?></td>
        </tr>
    <?php endforeach; ?>
</table>
<?php
$content = ob_get_clean();
include 'layout.php';
```