# **ПЕРШИЙ РОЗДІЛ**

## **1.1 Веб-застосунок**

Веб-застосунок — розподілений [застосунок](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), в якому [клієнтом](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0) виступає [браузер](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80), а [сервером](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80) — [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). Браузер може бути реалізацією так званих [тонких клієнтів](https://uk.wikipedia.org/wiki/%D0%A2%D0%BE%D0%BD%D0%BA%D0%B8%D0%B9_%D0%BA%D0%BB%D1%96%D1%94%D0%BD%D1%82)— логіка застосунку зосереджується на сервері, а функція браузера полягає переважно у відображенні інформації, завантаженої мережею з сервера, і передачі назад даних користувача. Однією з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому веб-застосунки є міжплатформовими сервісами.

Веб-застосунок отримує запит від клієнта і виконує обчислення, після цього формує веб-сторінку і відправляє її клієнтові мережею з використанням протоколу HTTP. Саме веб-застосунок може бути клієнтом інших служб, наприклад, [бази даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) або стороннього веб-застосунку, розташованого на іншому сервері.

Для побудування веб-застосунку будемо оперувати такими поняттями: операційна система(ОС), база даних, web-сервер, протоколи передачі даних.

## **Операційна система**

### 1.2.1 Означення

Операційна система, скорочено ОС ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) operating system, OS)  — це базовий комплекс [програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), що виконує управління [апаратним забезпеченням](https://uk.wikipedia.org/wiki/%D0%90%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [комп'ютера](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) або [віртуальної машини](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0); забезпечує керування обчислювальним процесом і організовує взаємодію з користувачем.

Операційна система звичайно складається з [ядра операційної системи](https://uk.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%BE%D1%97_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8) та базового набору [прикладного програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

### 1.2.2 Головні функції

Виконання на вимогу програм користувача тих елементарних (низькорівневих) дій, які є спільними для більшості програмного забезпечення і часто зустрічаються майже у всіх програмах (введення та виведення даних, запуск і зупинка інших програм, виділення та вивільнення додаткової пам'яті тощо).

* Стандартизований доступ до [периферійних пристроїв](https://uk.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B8%D1%84%D0%B5%D1%80%D1%96%D0%B9%D0%BD%D1%96_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97) ([пристрої введення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D1%81%D1%82%D1%80%D1%96%D0%B9_%D0%B2%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%BD%D1%8F)-[виведення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D1%81%D1%82%D1%80%D1%96%D0%B9_%D0%B2%D0%B8%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%BD%D1%8F)).
* Завантаження програм у [оперативну пам'ять](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0_%D0%BF%D0%B0%D0%BC%27%D1%8F%D1%82%D1%8C) і їх виконання.
* Керування оперативною пам'яттю (розподіл між [процесами](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), організація [віртуальної пам'яті](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BF%D0%B0%D0%BC%27%D1%8F%D1%82%D1%8C)).
* Керування доступом до даних енергонезалежних носіїв ([твердий диск](https://uk.wikipedia.org/wiki/%D0%A2%D0%B2%D0%B5%D1%80%D0%B4%D0%B8%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA), [оптичні диски](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D1%87%D0%BD%D0%B8%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA) тощо), організованим у тій чи іншій [файловій системі](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0).
* Забезпечення [користувацького інтерфейсу](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%86%D1%8C%D0%BA%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81).
* Мережеві операції, підтримка стеку [мережевих протоколів](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D1%96_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D0%B8).

### 1.2.3 Склад операційної системи

* [ядро операційної системи](https://uk.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%BE%D1%97_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8), що забезпечує розподіл та управління ресурсами обчислювальної системи;
* базовий набір прикладного програмного забезпечення, системні [бібліотеки](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) та програми обслуговування.

Ядро системи — це набір функцій, структур даних та окремих програмних модулів, які завантажуються в пам'ять комп'ютера при [завантаженні операційної системи](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D0%B2%D0%B0%D0%BD%D1%82%D0%B0%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%BE%D1%97_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8) та забезпечують три типи системних сервісів:

* управління введенням-виведенням інформації (підсистема вводу-виводу ядра ОС);
* управління оперативною пам'яттю (підсистема управління оперативною пам'яттю ядра ОС);
* управління [процесами](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) (підсистема управління процесами ядра ОС).

Кожна з цих підсистем представлена відповідними функціями ядра системи.

[Багатозадачні](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%BD%D1%96%D1%81%D1%82%D1%8C) операційні системи також включають ще одну обов'язкову складову — механізм підтримки [багатозадачності](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%BD%D1%96%D1%81%D1%82%D1%8C). Ця складова не надається в якості системного сервісу і тому не може бути віднесена до жодної з підсистем.

Існує три основних механізми забезпечення багатозадачності ([планування задач](https://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%BD%D1%83%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%BA_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%BE%D1%97_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8)):

1. шляхом надання [процесора](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80) окремій задачі на квант часу, який визначається самою задачею (кооперативна багатозадачність; останнім часом практично не використовується або область використання значно обмежена всередині процесів);
2. шляхом надання процесора окремій задачі на квант часу, який визначається обладнанням обчислювальної системи — інтервальним таймером;
3. виділення під окрему задачу окремого процесора в багатопроцесорних системах.

У перших двох випадках на кожному з процесорів в окремо взятий момент часу обраховується лише одна задача, але за рахунок достатньо малого кванту часу (в межах мілісекунд), що почергово надається кожній з задач, виникає ілюзія одночасного виконання в системі багатьох задач.

В сучасних системах, як правило комбінується методи 2 і 3.

В більшості сучасних ОС реалізовано підтримку стеку протоколів [TCP/IP](https://uk.wikipedia.org/wiki/TCP/IP). Це означає що вони можуть взаємодіяти в [мережі](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0), доступаючись до ресурсів одне одного.

Багато ОС також підтримують один чи кілька специфічних протоколів, як наприклад SNA на системах від [IBM](https://uk.wikipedia.org/wiki/IBM), DECnet на системах від Digital Equipment Corporation, та[Microsoft](https://uk.wikipedia.org/wiki/Microsoft)-специфічні протоколи для Windows. Для певних задач використовуються специфічні протоколи, як наприклад [NFS](https://uk.wikipedia.org/wiki/NFS) для роботи з файлами через мережу.

За метою використання операційні системи можна умовно поділити на:

* серверні
* користувацькі

### 1.2.4 Серверні операційні системи

Серверні операційні системи використовують для роботи серверів, оскільки набір їх можливостей з точки зору адміністрування є набагато ширшим ніж у звичайних користувацьких ОС. Призначенням серверної ОС є керування прикладними програмами та сервісами, що обслуговують користувачів локальної мережі та мережі Інтернет. До таких програм належать сучасні системи управління базами даних, засоби управління мережею, служби каталогів, засоби обміну повідомленнями, web-сервери, поштові сервери, мережеві екрани, та інші сервіси.  
Стабільна робота серверних ОС забезпечує роботу всієї комп’ютерної мережі організації, де її використовують, а тому вимоги до продуктивності та надійності таких ОС дуже високі. До таких вимого належать: підтримка кластерів (набір ряду однотипних комп'ютерів мережі, що виконують одне і те ж завдання одночасно розподіляючи між собою навантаження), можливість резервного копіювання даних, зміна конфігурації програмного забезпечення без перезавантаження операційної системи. При виборі серверної ОС необхідно в першу чергу враховувати її вимоги до апаратного забезпечення, так як потужність апаратного забезпечення безпосередньо впливатиме на продуктивність та надійність роботи сервера.

## **База даних**

### 1.3.1 Основні поняття

База даних ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) database) – це сукупність [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96_(%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%82%D0%B5%D1%85%D0%BD%D1%96%D0%BA%D0%B0)), організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування (за стандартом ISO/IEC 2382:2015). В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти. Дані у базі організовують відповідно до моделі організації [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96). Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки.

В загальному випадку базою даних можна вважати будь-який впорядкований набір даних. Наприклад, паперову картотеку з формулярами про працівників підприємства у відділі кадрів.

База даних — сховище даних, та [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) — засоби для роботи з базою даних. [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з інформаційної системи може бути видалена, але база даних продовжить існувати. І навпаки: [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) може функціонувати без жодної бази даних.

В загальному базу даних неможливо просто перемістити з однієї [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) до іншої. Але [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) використовують стандарти ([SQL](https://uk.wikipedia.org/wiki/SQL), [ODBC](https://uk.wikipedia.org/wiki/ODBC), [JDBC](https://uk.wikipedia.org/wiki/JDBC)), які уніфікують ряд операцій по роботі з даними і дозволяють різним [застосункам](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) працювати з базами даних різних [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) часто класифікують за моделлю організації даних. Найважливіші [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) використовують реляційну модель, у якій дані подають у виді таблиць. Для кінцевого користувача (та прикладних програм) робота з базою даних напряму неможлива. Всі маніпуляції над даними здійснюють через спеціальні запити, які надсилають до [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) опрацьовує їх і повертає результат. Безпосередньо з базою даних працює виключно [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

Сучасні [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) забезпечують функції щодо керування даними, які можна поділити на такі групи:

* Оголошення даних — створення, зміна та видалення визначень, які описують організацію даних.
* Модифікація даних — додавання даних, їх редагування та видалення.
* Отримання даних — надання даних за запитом застосунку у формі, яка дозволяє їх безпосереднє використання. Дані можуть надаватись або у формі, в якій вони зберігаються у базі даних, або в іншій формі (наприклад, через поєднання різних даних).
* Адміністрування даних — реєстрування та відслідковування дій користувачів, дотримання безпеки роботи з даними, забезпечення надійності та цілісності даних, моніторинг продуктивності, резервне копіювання та відновлення даних тощо.

### 1.3.2 Архітектура «файл-сервер»

Ця архітектура передбачає виділення однієї з машин мережі в якості головної (сервер). На такій машині зберігається спільна централізована БД. Усі інші машини мережі виконують функції робочих станцій, за допомогою яких підтримується доступ користувацької системи до бази даних. Файли бази даних відповідно до призначених для користувача запитів передаються на робочі станції, де в основному і проводиться обробка даних. При великій інтенсивності доступу до одних і тих же даних продуктивність інформаційної системи різко падає. Користувачі також можуть створювати на робочих станціях локальні БД, які використовуються ними монопольно.

### 1.3.3 Архітектура «клієнт-сервер»

В сучасних мережевих інформаційних системах для роботи із загальною базою даних використовують архітектуру [«клієнт-сервер»](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). При цьому в мережі розміщують сервер баз даних. Ним виступає [комп'ютер](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) (або комп'ютери), який містить бази даних, [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) та пов'язане з ними програмне забезпечення, і налаштований для надання користувачам інформаційної системи доступу до бази даних. Клієнти, які працюють із даними (вони можуть бути розташовані на різних комп'ютерах мережі), надсилають відповідні запити серверу. Сервер їх отримує, опрацьовує, та надсилає відповідь клієнту. Сучасні [СКДБ](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) ([MySQL](https://uk.wikipedia.org/wiki/MySQL), [PostgreSQL](https://uk.wikipedia.org/wiki/PostgreSQL), [Microsoft SQL Server](https://uk.wikipedia.org/wiki/Microsoft_SQL_Server) та інші) працюють відповідно до цієї архітектури. Сервер баз даних, як правило, є достатньо потужною багатопроцесорною системою, яка використовує масиви дисків [RAID](https://uk.wikipedia.org/wiki/RAID) для підвищення надійності зберігання даних. Використання дискових масивів [RAID](https://uk.wikipedia.org/wiki/RAID) дозволяє відновити дані, навіть якщо один з дисків вийшов з ладу.

### 1.3.4 Класифікація баз даних

Бази даних класифікують за різними критеріями. За моделлю організації [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96_(%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%82%D0%B5%D1%85%D0%BD%D1%96%D0%BA%D0%B0)) розрізняють такі бази даних:

* Ієрархічна. Ієрархічна база даних може бути представлена як дерево, що складається з об'єктів різних рівнів. Між об'єктами існують зв'язки типу «предок-нащадок». При цьому можлива ситуація, коли об'єкт не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок.
* Мережна. Така база даних подібна до ієрархічної, за винятком того, що кожен об'єкт може мати більше одного предка.
* [Реляційна](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). [Реляційна база даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) зберігає дані у вигляді таблиць. Найвживаніші [СКБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) використовують [реляційну модель даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).
* Об'єктно-орієнтована. У базі даних цього виду дані оформляють у вигляді моделей об'єктів.

За розміщенням даних виділяють такі види баз:

* Локальна, або централізована. Така база даних підтримується на одному комп'ютері.
* Розподілена. Частини такої бази даних розміщують на різних комп'ютерах мережі.

За структурованістю:

Структуровані БД використовують [структури даних](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), тобто структурований опис типу фактів за допомогою схеми даних, більш відомої як [модель даних](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). Модель даних описує об'єкти та взаємовідношення між ними. Існує декілька моделей (чи типів) баз даних, основні: [ієрархічна](https://uk.wikipedia.org/w/index.php?title=%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_%D1%96%D1%94%D1%80%D0%B0%D1%80%D1%85%D1%96%D1%87%D0%BD%D0%B0&action=edit&redlink=1), [мережна](https://uk.wikipedia.org/w/index.php?title=%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%BD%D0%B0&action=edit&redlink=1) та [реляційна](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

До неструктурованих БД належать повнотекстові бази даних, які містять неструктуровані тексти статей чи книг у формі, що дозволяє здійснювати швидкий пошук.

Найбільш ходові і популярні способи – це об'єктна й реляційна моделі даних.Реляційна модель використовує представлення даних як сукупність двовимірних таблиць особливого виду, відомого в математичній сфері як відношення. Таким чином, перша [модель](http://school.xvatit.com/index.php?title=%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%B5) являє собою набір таблиць, зв'язаних між собою. Кожний рядок і стовпчик в таблиці відповідає дійсності реального світу (наприклад, це інформація про людину).

Об'єктна модель орієнтується на об'єктному [програмуванні](http://school.xvatit.com/index.php?title=%D0%9E_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%BC_%D0%9F%D0%9E_%D0%B8_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%D1%85_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F). Тобто, у такій моделі дані являють собою набір об'єктів и властивостей, зв'язаних між собою якими-небудь подібностями. Робота з об'єктами здійснюється за допомогою схованих у них методів.

## **Web-сервер**

Веб-сервер ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Web Server) — це [сервер](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80), що приймає [HTTP](https://uk.wikipedia.org/wiki/HTTP)-запити від [клієнтів](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), зазвичай [веб-браузерів](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80), видає їм [HTTP](https://uk.wikipedia.org/wiki/HTTP)-відповіді, зазвичай разом з [HTML](https://uk.wikipedia.org/wiki/HTML)-сторінкою, зображенням, [файлом](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB), медіа-потоком або іншими даними. Веб-сервер — основа [Всесвітньої павутини](https://uk.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%8F_%D0%BF%D0%B0%D0%B2%D1%83%D1%82%D0%B8%D0%BD%D0%B0).

Веб-сервером називають як [програмне забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), що виконує функції веб-сервера, так і комп'ютер, на якому це програмне забезпечення працює.

Клієнти дістаються веб-сервера за [URL](https://uk.wikipedia.org/wiki/URL)-адресою потрібної їм [веб-сторінки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) або іншого ресурсу.

Клієнти для звернення до веб-серверів можуть використовуватися абсолютно різні пристрої:

* [Веб-браузер](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) — найпоширеніший спосіб
* Спеціальне [програмне забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) може самостійно звертатися до веб-серверів для отримання оновлень або іншої інформації
* [Мобільний телефон](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B1%D1%96%D0%BB%D1%8C%D0%BD%D0%B0_%D1%82%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD%D1%96%D1%8F) може дістатися до ресурсів веб-сервера за допомогою протоколу [WAP](https://uk.wikipedia.org/wiki/WAP) або HTTP
* Інші [інтелектуальні](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82) пристрої або [побутова техніка](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B1%D1%83%D1%82%D0%BE%D0%B2%D0%B0_%D1%82%D0%B5%D1%85%D0%BD%D1%96%D0%BA%D0%B0)

## **1.5 Протоколи передавання даних**

### 1.5.1 Поняття протоколів

Протокол передавання даних — набір угод [інтерфейсу](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) логічного рівня, які визначають обмін даними між різними [програмами](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F). Ці угоди задають однаковий спосіб передачі повідомлень і обробки помилок при взаємодії програмного забезпечення рознесеного на просторі апаратної платформи, з'єднаної тим чи іншим інтерфейсом.

[Стандартизований](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82) протокол передачі даних також дозволяє розробляти інтерфейси (вже на фізичному рівні), не прив'язані до конкретної [апаратної платформи](https://uk.wikipedia.org/w/index.php?title=%D0%90%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%B0_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%B0&action=edit&redlink=1) і виробнику (наприклад, [USB](https://uk.wikipedia.org/wiki/USB), [Bluetooth](https://uk.wikipedia.org/wiki/Bluetooth)).

Мережевий протокол — набір правил, що дозволяє здійснювати з'єднання і обмін даними між двома і більше включеними в [мережі](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0) пристроями. Різні протоколи, найчастіше, описують лише різні сторони одного типу [зв'язку](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D0%BA%D0%BE%D0%BC%D1%83%D0%BD%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%97); взяті разом, вони утворюють стек протоколів. Назви «протокол» і «стек протоколів» також вказують на [програмне забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), яким реалізується протокол. Нові протоколи для [Інтернет](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) а визначаються [IETF](https://uk.wikipedia.org/wiki/IETF), а інші протоколи — [IEEE](https://uk.wikipedia.org/wiki/IEEE) або [ISO](https://uk.wikipedia.org/wiki/ISO). [ITU-T](https://uk.wikipedia.org/wiki/ITU-T) займається телекомунікаційними протоколами та форматами.

Найпоширенішою системою класифікації мережних протоколів є так звана [модель OSI](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), відповідно до якої протоколи поділяються на 7 рівнів за своїм призначенням — від [фізичного](https://uk.wikipedia.org/wiki/%D0%A4%D1%96%D0%B7%D0%B8%D1%87%D0%BD%D0%B8%D0%B9_%D1%80%D1%96%D0%B2%D0%B5%D0%BD%D1%8C) (формування і розпізнавання електричних або інших сигналів) до [прикладного](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%B8%D0%B9_%D1%80%D1%96%D0%B2%D0%B5%D0%BD%D1%8C)([інтерфейс програмування додатків](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B4%D0%BE%D0%B4%D0%B0%D1%82%D0%BA%D1%96%D0%B2) для передачі інформації додатками).

Мережеві протоколи вимагають правила роботи комп'ютерам, які підключені до мережі. Вони будуються за багаторівневим принципом. Протокол деякого рівня визначає одне з технічних правил зв'язку. В наш часдля мережевих протоколів використовується модель [OSI](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI) (Open System Interconnection — взаємодія відкритих систем, ВОС).

### 1.5.2 Модель OSI

Модель OSI — це 7-рівнева логічна модель роботи мережі. Модель OSI реалізується групою протоколів і правил зв'язку, організованих в кілька рівнів:

* На фізичному рівні визначаються фізичні (механічні, електричні, оптичні) характеристики ліній зв'язку;
* На канальному рівні визначаються правила використання фізичного рівня вузлами мережі;
* Мережевий рівень відповідає за адресацію і доставку повідомлень;
* Транспортний рівень контролює черговість проходження компонентів повідомлення;
* Завдання сеансового рівня — координація зв'язку між двома прикладними програмами, що працюють на різних робочих станціях;
* Рівень представлення служить для перетворення даних із внутрішнього формату комп'ютера у формат передачі;
* Прикладний рівень є прикордонним між прикладною програмою і іншими рівнями — забезпечує зручний інтерфейс зв'язку мережевих програм користувача.

### 1.5.3 Види протоколів

Стек протоколів [TCP / IP](https://uk.wikipedia.org/wiki/TCP/IP) — це два протоколи нижнього рівня, що є основою зв'язку в мережі Інтернет. Протокол TCP розбиває передану інформацію на порції і нумерує всі порції. За допомогою протоколу IP всі частини передаються одержувачу. Далі за допомогою протоколу TCP перевіряється, чи всі частини отримані. При отриманні всіх порцій TCP розміщує їх в потрібному порядку і збирає в єдине ціле.

TCP (англійською Transmission Control Protocol — протокол керування переда­ванням) відповідає за організацію сеансу зв'язку між двома комп'ютерами у мережі.  
  
IP (англійською Internet Protocol — міжмережний протокол) відповідає за маршрутизацію, тобто за те, щоб пакет було доставлено за певною адресою. За допомогою протоколу TCP ПК перевіряє, чи всі частини отримано. При отриманні всіх порцій TCP розміщує їх в потрібному порядку і збирає в одне ціле.

Найвідоміші протоколи, використовувані в мережі Інтернет:

* [HTTP](https://uk.wikipedia.org/wiki/HTTP) (Hyper Text Transfer Protocol) — це протокол передачі [гіпертексту](https://uk.wikipedia.org/wiki/%D0%93%D1%96%D0%BF%D0%B5%D1%80%D1%82%D0%B5%D0%BA%D1%81%D1%82). Протокол HTTP використовується при пересиланні Web-сторінок з одного комп'ютера на інший.
* [FTP](https://uk.wikipedia.org/wiki/FTP) (File Transfer Protocol) — це протокол передачі файлів зі спеціального файлового сервера на комп'ютер користувача. FTP дає можливість абоненту обмінюватися двійковими і текстовими файлами з будь-яким комп'ютером мережі. Встановивши зв'язок з віддаленим комп'ютером, користувач може скопіювати файл із віддаленого комп'ютера на свій або скопіювати файл зі свого комп'ютера на віддалений.
* [POP](https://uk.wikipedia.org/wiki/POP) (Post Office Protocol) — це стандартний протокол поштового з'єднання. Сервери POP обробляють вхідну пошту, а протокол POP призначений для обробки запитів на отримання пошти від клієнтських поштових програм.
* [SMTP](https://uk.wikipedia.org/wiki/SMTP) (Simple Mail Transfer Protocol) — протокол, який задає набір правил для передачі пошти. Сервер SMTP повертає або підтвердження про прийом, або повідомлення про помилку, або запитує додаткову інформацію.
* [Uucp](https://uk.wikipedia.org/w/index.php?title=Uucp&action=edit&redlink=1) (Unix to Unix Copy Protocol) — це нині застарілий, але все ще застосовується протокол передачі даних, у тому числі для електронної пошти. Цей протокол передбачає використання пакетного способу передачі інформації, при якому спочатку встановлюється з'єднання клієнт-сервер і передається пакет даних, а потім автономно відбувається його обробка, перегляд або підготовка листів.
* [Telnet](https://uk.wikipedia.org/wiki/Telnet) — це протокол віддаленого доступу. TELNET дає можливість абоненту працювати на будь-який ЕОМ мережі Інтернет, як на своїй власній, тобто запускати програми, змінювати режим роботи і так далі. На практиці можливості лімітуються тим рівнем доступу, який заданий адміністратором віддаленої машини.
* [DTN](https://uk.wikipedia.org/wiki/DTN) — протокол, призначений для забезпечення наддалекої космічного зв'язку.

Розглянемо детальніше протоколи прикладного рівня, які будуть фігурувати у дипломному проекті.

### 1.5.4 HTTP

HTTP — [протокол](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) передачі даних, що використовується в [комп'ютерних](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) мережах. Назва скорочена від Hyper Text TransferProtocol, протокол передачі [гіпер-текстових документів](https://uk.wikipedia.org/wiki/%D0%93%D1%96%D0%BF%D0%B5%D1%80%D1%82%D0%B5%D0%BA%D1%81%D1%82)

HTTP належить до протоколів моделі [OSI](https://uk.wikipedia.org/wiki/OSI) 7-го [прикладного рівня](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%B8%D0%B9_%D1%80%D1%96%D0%B2%D0%B5%D0%BD%D1%8C).

Основним призначенням протоколу HTTP є передача [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) ([текстових файлів](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB) з розміткою [HTML](https://uk.wikipedia.org/wiki/HTML)), хоча за допомогою нього успішно передаються і інші [файли](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB), які пов'язані з веб-сторінками ([зображення](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B5_%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F) і [застосунки](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BA%D0%B8)), так і не пов'язані з ними (у цьому HTTP конкурує з складнішим [FTP](https://uk.wikipedia.org/wiki/FTP)).

HTTP припускає, що клієнтська програма — [веб-браузер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) — здатна відображати гіпертекстові веб-сторінки та файли інших типів у зручній для користувача формі. Для правильного відображення HTTP дозволяє клієнтові дізнатися мову та [кодування](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) веб-сторінки й/або запитати версію сторінки в потрібних мові/кодуванні, використовуючи позначення із стандарту [MIME](https://uk.wikipedia.org/wiki/MIME).

HTTP — протокол прикладного рівня, схожими на нього є [FTP](https://uk.wikipedia.org/wiki/FTP) і [SMTP](https://uk.wikipedia.org/wiki/SMTP). Обмін повідомленнями йде за звичайною схемою «запит-відповідь». Для ідентифікації ресурсів HTTP використовує глобальні [URI](https://uk.wikipedia.org/wiki/URI). На відміну від багатьох інших протоколів, HTTP не зберігає свого стану. Це означає відсутність збереження проміжного стану між парами «запит-відповідь». Компоненти, що використовують HTTP, можуть самостійно здійснювати збереження інформації про стан, пов'язаний з останніми запитами та відповідями. [Браузер](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80), котрий посилає запити, може відстежувати затримки відповідей. [Сервер](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80) може зберігати [IP-адреси](https://uk.wikipedia.org/wiki/IP-%D0%B0%D0%B4%D1%80%D0%B5%D1%81) та заголовки запитів останніх клієнтів. Проте, згідно з протоколом, клієнт та сервер не мають бути обізнаними з попередніми запитами та відповідями, у протоколі не передбачена внутрішня підтримка стану й він не ставить таких вимог до клієнта та сервера.

Кожен запит/відповідь складається з трьох частин:

1. стартовий рядок;
2. заголовки;
3. тіло повідомлення, що містить дані запиту, запитаний ресурс або опис проблеми, якщо запит не виконано.

Обов'язковим мінімумом запиту є стартовий рядок. Починаючи з HTTP/1.1 обов'язковим став заголовок Host: (щоб розрізнити кілька доменів, які мають одну й ту ж IP-адресу).  
Доменне ім'я — це послідовність розділених крапками слів, яка зіставляється з певною ІР-адресою.

Стартові рядки розрізняються для запиту й відповіді. Рядок запиту виглядає так:

<Метод> <URL> HTTP/ <Версія>, де ‹Метод› може бути:

OPTIONS

Повертає методи HTTP, які підтримуються сервером. Цей метод може служити для визначення можливостей веб-сервера.

GET

Запрошує вміст вказаного ресурсу. Запитаний ресурс може приймати параметри (наприклад, пошукова система може приймати як параметр шуканий рядок). Вони передаються в рядку URI (наприклад: <http://www.example.net/resource?param1=value1&param2=value2>). Згідно зі стандартом HTTP, запити типу GET вважаються ідемпотентними — багатократне повторення одного і того ж запиту GET повинне приводити до однакових результатів (за умови, що сам ресурс не змінився за час між запитами). Це дозволяє кешувати відповіді на запити GET.

HEAD

Аналогічний методу GET, за винятком того, що у відповіді сервера відсутнє тіло. Це корисно для витягання мета-інформації, заданої в заголовках відповіді, без пересилання всього вмісту. Зокрема, [клієнт](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82) чи [проксі](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96), перевіривши заголовок Last-Modified: (останній час модифікації), таким чином може переконатися, що сторінка на сервері не змінилася від часу попереднього запиту.

POST

Передає призначені для користувача дані (наприклад, з HTML-форми) заданому ресурсу. Наприклад, в [блогах](https://uk.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%B3) відвідувачі зазвичай можуть вводити свої коментарі до записів в HTML-форму, після чого вони передаються серверу методом POST, і він поміщає їх на сторінку. При цьому передані дані (у прикладі з блогами — текст коментаря) включаються в тіло запиту. На відміну від методу GET, метод POST не вважається ідемпотентним, тобто багатократне повторення одних і тих же запитів POST може повертати різні результати (наприклад, після кожного відправлення коментаря з'являтиметься одна копія цього коментаря).

PUT

Завантажує вказаний ресурс на сервер.

PATCH

Завантажує певну частину ресурсу на сервер.

DELETE

Видаляє вказаний ресурс.

TRACE

Повертає отриманий запит так, що [клієнт](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82) може побачити, що проміжні [сервери](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80) додають або змінюють в запиті.

CONNECT

Для використання разом з [проксі-серверами](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), які можуть динамічно перемикатися в тунельний режим [SSL](https://uk.wikipedia.org/wiki/SSL).

В основному використовуються методи GET і POST.

Відповідь сервера буде виглядати так (перший рядок):

HTTP/ <Версія> <Код статусу> <Опис статусу>

### 1.5.5 HTTPS

HTTPS — схема [URI](https://uk.wikipedia.org/wiki/URI), що синтаксично ідентична http: схемі, яка зазвичай використовується для доступу до ресурсів [Інтернет](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82). Використання https: URL вказує, що протокол [HTTP](https://uk.wikipedia.org/wiki/HTTP) має використовуватися, але з іншим портом за замовчуванням (443) і додатковим шаром [шифрування](https://uk.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)/[автентифікації](https://uk.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F) між HTTP і [TCP](https://uk.wikipedia.org/wiki/TCP). Ця схема була винайдена у компанії [Netscape Communications Corporation](https://uk.wikipedia.org/wiki/Netscape_Communications_Corporation) для забезпечення автентифікації та шифрування комунікацій і широко використовується в [Інтернеті](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) у програмному забезпеченні, в якому важлива безпека комунікацій, наприклад, у платіжних системах та корпоративних логінах.

Власне кажучи, HTTPS це не окремий протокол, а звичайний HTTP через [SSL](https://uk.wikipedia.org/wiki/SSL) або [TLS](https://uk.wikipedia.org/wiki/TLS). Це гарантує помірний захист від підслуховування та від нападу [«людина посередині»](https://uk.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_%C2%AB%D0%BB%D1%8E%D0%B4%D0%B8%D0%BD%D0%B0_%D0%BF%D0%BE%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%B8%D0%BD%D1%96%C2%BB) (man-in-the-middle) (якщо це здійснюється належним чином і уповноважені на видачу [сертифікатів](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D1%81%D0%B5%D1%80%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%82) верхнього рівня роблять свою роботу належним чином).

TCP портом за замовчуванням для HTTPS є 443 (для HTTP — 80).

Щоб підготувати веб-сервер для прийняття https транзакцій адміністратор повинен створити сертифікат з відкритим ключем для веб-сервера. Ці сертифікати можуть бути створені на [UNIX](https://uk.wikipedia.org/wiki/UNIX) сервері такими програмами, як наприклад [OpenSSL](https://uk.wikipedia.org/wiki/OpenSSL). Цей сертифікат повинен бути [підписаний](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D0%BF%D1%96%D0%B4%D0%BF%D0%B8%D1%81) уповноваженим на видачу сертифікатів ([certificate authority](https://uk.wikipedia.org/wiki/Certificate_authority)), який засвідчує, що отримувач сертифікату — саме той, про кого йдеться у сертифікаті. [Браузери](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) розповсюджуються з сертифікатами уповноважених на видачу сертифікатів верхнього рівня організацій, таким чином браузери можуть перевірити сертифікати підписані ними.

Організації можуть також мати власних уповноважених на видачу сертифікатів, особливо якщо вони відповідальні за конфігурацію браузерів, що мають доступ до їх власних сайтів (наприклад, сайти на внутрішній мережі компанії), оскільки вони можуть тривіально додати свій власний сертифікат до браузера.

Деякі сайти використовують самостійно підписані сертифікати. Їх використання забезпечує захист проти підслуховування, але є ризик нападу «людина-посередині». Для запобігання нападу необхідна перевірка сертифікату іншим методом (наприклад подзвонити власнику сертифіката задля перевірки контрольної суми сертифіката).

Система може також використовуватися для клієнтської автентифікації, щоб обмежити доступ до веб-сервера тільки зареєстрованими користувачами. Для цього адміністратор сайту створює сертифікати для кожного користувача, які завантажуються в їх браузер. Ці сертифікати зазвичай містять ім'я і [електронну пошту](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BF%D0%BE%D1%88%D1%82%D0%B0) зареєстрованого користувача та автоматично перевіряються сервером при кожному повторному підключенні. Повторне введення паролю не потрібне.

Рівень захисту залежить від коректності запровадження браузерного і серверного програмного забезпечення та підтримуваних криптографічних алгоритмів.

### 1.5.6 Pop3

Post Office Protocol (Поштовий Офісний Протокол) — це [протокол](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB), що використовується клієнтом для доступу до повідомленнь[електронної пошти](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BF%D0%BE%D1%88%D1%82%D0%B0) на [сервері](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%88%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80).

Остання версія протоколу — третя. POP3 дозволяє [клієнтові](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82) мати вибірковий доступ до повідомлень на [сервері](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80). За своєю функціональністю POP є набагато біднішим за [IMAP](https://uk.wikipedia.org/wiki/IMAP) протокол, не надаючи клієнту [інтерфейсу](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) з маніпулювання [папками](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D1%80%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D1%96%D1%8F_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97)) на сервері, вибірковим отриманням частин повідомлення чи можливості завантаження заголовків листів.

POP3 протокол, за замовчуванням працює на 110 порту TCP. Шифрований Secure POP3 (SSL-POP) працює на 995 порту TCP.

POP підтримує прості вимоги «завантаж-і-видали» для доступу до віддалених поштових скриньках. Хоча більша частина POP-клієнтів надає можливість залишити пошту на сервері після завантаження, що використовують POP клієнти зазвичай з'єднуються, витягають всі листи, зберігають їх на комп'ютері користувача як нові повідомлення, видаляють їх з сервера, після чого роз'єднуються.

Інші протоколи, зокрема IMAP, надають більш повний і комплексний віддалений доступ до типових операцій з поштовою скринькою. Багато клієнтів електронної пошти підтримують як POP, так і IMAP; однак, набагато менше інтернет-провайдерів підтримують IMAP.

У протоколі POP3 передбачено 3 стану сеансу:

* Авторизація - клієнт проходить процедуру аутентифікації.
* Транзакція - клієнт отримує інформацію про стан поштової скриньки, приймає і видаляє пошту.
* Оновлення - сервер видаляє обрані листи і закриває з'єднання.

### 1.5.7 IMAP

IMAP ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Internet Message Access Protocol — «Протокол доступу до інтернет-повідомлень») — [мережевий протокол](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB)прикладного рівня для доступу до [електронної пошти](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BF%D0%BE%D1%88%D1%82%D0%B0).

Аналогічно [POP3](https://uk.wikipedia.org/wiki/POP3), служить для роботи з вхідними листами, однак забезпечує додаткові функції, зокрема, можливість пошуку за ключовим словом без збереження пошти в локальній пам'яті.

IMAP надає користувачеві великі можливості для роботи з поштовими скриньками, розташованими на [центральному сервері](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%88%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80).[Поштовий клієнт](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%88%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BB%D1%96%D1%94%D0%BD%D1%82), що використовує цей протокол, отримує доступ до сховища кореспонденції на сервер так, начебто ця кореспонденція розташована на [комп'ютері](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) одержувача. Електронними листами можна маніпулювати з комп'ютера користувача (клієнта) без постійного пересилання з сервера і назад файлів з повним змістом листів. Для відправки листів використовується протокол [SMTP](https://uk.wikipedia.org/wiki/SMTP).

Більшість команд можна використовувати лише в певних станах.

* Без аутентифікації - клієнт повинен надати ім'я і пароль, перш ніж йому стане доступна більшість команд. Перехід в цей стан виробляється при встановленні з'єднання без попередньої аутентифікації.
* Аутентифікація - клієнт ідентифікований і повинен вибрати поштову скриньку, після чого йому стануть доступні команди для роботи з повідомленнями. Перехід в цей стан відбувається при встановленні з'єднання з попередньою аутентифікацією, коли видані всі необхідні ідентифікаційні дані або при помилковому виборі поштової скриньки.
* Вибір - система потрапляє, коли успішно здійснений вибір поштової скриньки.
* Вихід - система потрапляє при перериванні з'єднання в результаті запиту клієнта або внаслідок незалежного рішення сервера.

IMAP був розроблений для заміни простішого протоколу POP3 і має такі переваги в порівнянні з останнім:

* Листи зберігаються на [сервері](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%88%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), а не на машині клієнта. Можливий доступ до одної і тої ж поштової скриньки з різних клієнтів. Підтримується також одночасний доступ декількох клієнтів. У протоколі є механізми, за допомогою яких клієнт може бути проінформований про зміни, зроблені іншими клієнтами.
* Підтримка декількох поштових скриньок (або тек). Клієнт може створювати, вилучати і перейменовувати поштові скриньки на сервері, а також переміщати листи з одної поштової скриньки в інші.
* Можливе створення спільних папок, до яких можуть мати доступ декілька користувачів.
* Інформація про стан листів зберігається на сервері і доступна всім клієнтам. Листи можуть бути позначені як прочитані, важливі тощо
* Підтримка пошуку на сервері. Немає необхідності завантажувати з сервера безліч повідомлень для того, щоб знайти одне потрібне.
* Підтримка онлайн-роботи. Клієнт може підтримувати з сервером постійне з'єднання, при цьому сервер у реальному часі інформує клієнта про зміни в поштових скриньках, у тому числі про нові листи.
* Передбачено механізм розширення можливостей протоколу.

Поточна версія протоколу має позначення IMAP4rev1 (IMAP, версія 4, ревізія 1). Протокол підтримує передачу пароля користувача в зашифрованому вигляді. Крім того, IMAP-трафік можна зашифрувати за допомогою [SSL](https://uk.wikipedia.org/wiki/SSL).

# **ДРУГИЙ РОЗДІЛ**

## **2.1 Існуючі рішення проблеми**

Метою даного дипломного проекту є створення web-додатку для адміністрування бази медичних карт. А саме заходи забезпечення захищеного документообігу між клієнтом та сервером, під час взаємодії користувача або адміністратора з медичною карткою пацієнта, за допомогою веб-застосунку.

Загалом, проблему адміністрування будь-якої бази даних у масштабах великої системи із значним документообігом вирішують різними способами, які безпосередньо залежать як від необхідного рівня безпеки збереження та маніпулювання даними, так і ресурсів, які можна витратити на забезпечення системи і її подальше покращення.

До низького рівня можна віднести структури, де бази даних організовані в табличних процесах початкового рівня та журналах. А документообіг здійснюється за однією з найгірших стратегій. У кожного підрозділу є свій журнал із своїми записами, та власною нумерацією документів.

Наприклад один підрозділ відправив документ, який зареєстрований у його журналі умовним номером «А», на центральне управління. Але у журналі управління отриманий документ вирішили зареєструвати (згідно з власною нумерацією) умовним номером «Б».

Таке розподілення не буде впливати на роботу лише доти, доки не з’явиться потреби повторно отримати ці документи. При такому запиті можуть виникнути серйозні проблеми, адже умовні номери підрозділу і управління не співпадають, а записи про зміну позначення можуть бути зашкоджені, загублені або і зовсім не впроваджені. До того ж, забезпечити збереження паперових документів на достатньому рівні, дуже важко, адже завжди існує людський фактор, та вірогідність знищення документів за умов аварійної ситуації, пов’язаної з техногенним фактором.

До середнього рівня можна віднести системи на зразок достатньо розвиненого медичного закладу. Вони пропонують таке рішення проблеми. Все налагоджено за «клієнт-серверною» архітектурою.

Існує єдиний сервер, для котрого прописані адреси всіх комп’ютерів, які можуть мати до нього доступ. Він організований на окремій системі, для підключення потрібно мати комп’ютер з емулятором цієї системи зі спільним паролем, заходячи до котрої, отримуємо доступ до всієї бази пацієнтів. Далі, в залежності від вибраного ім’я лікаря, видається вся клієнтська база поточного користувача. Окрім цього, з’являються форми реєстрації пацієнтів, в залежності від галузі професії користувача. Такі форми зручні у використанні при занесенні інформації до бази даних одразу під час консультації пацієнта. Представлена система доволі зручна у використанні, також виключається потреба обміну документами поміж підрозділами, тому що кожен, хто потребує певної інформації о пацієнті та результатів його попередньої консультації у іншого лікаря, може просто вибрати потрібного користувача у відповідному секторі програми.

Але така система має певний ряд недоліків. Перш за все, система розроблена доволі давно, тому на майбутнє виключається можливість використання нового обладнання, а взаємодія клієнта з сервером обмежена його (клієнта) потребою у емуляторі.

Не дивлячись на зручність документообігу представленої системи, неможливо не помітити, що, по-перше, безпечність її дуже сумнівна. Адже будь-хто у клініці, знаючи пароль, має доступ до всієї бази даних пацієнтів. По-друге, надіслати документ людині (не вилучаючи його із системи і не формуючи окремий звіт), яка не має доступу до підключеного у подібну систему ПК, і знаходиться, наприклад, у відпустці або у відрядженні, буде виникати потреба додаткових засобів передачі конфіденціальної інформації відкритими каналами зв’язку, наприклад e-mail. На мою думку, найбільший недолік такої системи - прив’язка до локальної бази даних.

До високого рівня можна віднести системи великих корпорацій або державних установ. У якості приклада приведемо систему пенсійного фонду. Організація виглядає таким чином. Існує головний сервер з базою даних, центральне управління з адміністратором, який займається її обслуговуванням та підрозділи, які заносять інформацію до спільної бази даних. Усі отримані документи сортуються згідно з датою, підрозділом, тематикою ті іншими критеріями, які можуть бути залучені у пошуковому процесі. Підрозділи захищені від вільного доступу. Щоб отримати інформацію, або документацію з іншого підрозділу треба звернутися до бази даних з відповідним запитом. Така концепція дозволяє інкапсулювати документацію кожного окремого відділку.

Але є проблема, яка істотно впливає на покращення взаємодії користувача з програмою – неможливість внесення змін до коду програми. Програма була розроблена для обробки надходячої інформації з текстових файлів логування операцій. Для логування наше повідомлення (отримане або відправлене) заносилося до наскрізної таблиці, згідно з якою, отримували можливість здійснювати моніторинг обміну інформацією між підрозділами.

Нажаль на зараз ця технологія застаріла, як матеріально, так і морально. Вона не буду працювати з новими операційними системами, а отже і не матиме можливість працювати на новому технічному обладнанні. Це пояснюється рядом чинників. Для забезпечення роботи даної технології використовується зв’язка операційної системи SCO UnixWare і FTP протоколу. Неможливість використання іншого технічного обладнання обумовлено саме використанням операційної системи SCO (розширене пояснення якої, буде представлене у наступному підрозділі). Іншою проблемою стає використання FTP. Дана програма є програмою з зарезервованими підключеннями, тобто для кожного окремого користувача ми маємо організувати окремий доступ. Але організація доступу у програмі з закритим кодом – майже неможлива задача, бо ми не знаємо і не можемо змінити правила підключення по FTP. Тобто, нам представлена класична модель «чорного ящика», де ми можемо маніпулювати із вхідними та вихідними даними, в той час, як правила і реалізація поза зоною нашего досягнення.

Окрім такого базового розподілення, можна винести окремо багато частин з більш або менш розвиненими технологіями. Наприклад, системи, які потребують дуже високої безпечності та майже не обмежуються у фінансовому забезпеченні, можуть використовувати електронні ключі (іноді донгл). Це апаратний засіб, призначений для захисту програмного забезпечення (ПО) і даних від копіювання, нелегального використання і розголошення. Він дозволяє зберігати на собі ключі, за якими здійснюється доступ до документів. Ключі змінюються кожну хвилину (секунду, пів секунди, в залежності від якості ключа), при підключенні виникає звернення до сервера, якщо ключі співпадають – доступ дозволяється. Зазвичай має вигляд флешки. Такий засіб забезпечення безпеки дуже надійний, але вимагає великих витрат. Як можемо бачити, реалізація залежить виключно від потреб конкретного підприємства, можливості його фінансування та подальшого забезпечення системи усіма видами ресурсів.

Наразі набирає все більш популярності підхід застосування веб-застосунків. Де використовується переваги доступу до бази даних без потреби бути прив’язаним до локальної бази даних. Можливість підключатися з будь-якого місця, за наявності браузера та інтернету. До того ж ідея веб-застосунку перш за все – багатоплатформенність. Саме з цього основного ряду чинників, у даному дипломному проекті була обрана саме ця стратегія реалізації.

**2.2 Програмні та технічні продукти для рішення проблеми**

### 2.2.1 Операційні системи

Побудова IT-інфраструктури будь-якого підприємства, як і придбання домашнього комп'ютера, починається з вибору операційних систем. Операційна система є найбільш важливою частиною програмного забезпечення, що керує ресурсами комп'ютера, на якому вона виконується (а іноді і ресурсами інших комп'ютерів мережі), та надає користувачу певні послуги.

Операційні системи для кінцевих користувачів (клієнтські операційні системи) призначені для надання користувачам доступу до певного набору застосувань і для керування ресурсами комп'ютера. В останні роки виробники операційних систем, як правило, випускають такі ОС в двох редакціях — для корпоративних і для домашніх користувачів.

Операційні системи для корпоративних користувачів, зазвичай, орієнтовані на роботу з офісними і бизнес-застосуваннями, для звернення до корпоративних файлових чи поштових серверів, серверів застосувань та інших корпоративних чи зовнішніх ресурсів. Тому такі ОС повинні задовольняти жорстким вимогам стосовно надійності, безпеки, керованості.

Що стосується домашніх ОС, то вимоги до їх безпеки і керованості не є такими високими, але критерії підтримки технологій мультимедіа, що використовуються в іграх, додатках для обробки фото чи відео та в інших подібних застосуваннях, можуть виявитися досить серйозними.

І корпоративні, і домашні користувачі надають перевагу операційним системам з графічним користувацьким інтерфейсом. Вперше перспективність графічного інтерфейсу усвідомили засновники компанії Apple Стівен Джобс і Стівен Возняк, що побачили його в одній з дослідницьких лабораторій Xerox. Їх перший комп'ютер, що випущено в 1983 році, мав графічний користувацький інтерфейс, а роком пізніше з'явився перший Macintosh з операційною системою Mac OS, яка протягом довгого часу була найпопулярнішою операційною системою подібного класу.­­

#### 2.2.1.1 Клієнтські операційні системи

Windows

Сьогодні найбільш популярними операційними системами для кінцевих користувачів є сімейство продуктів Microsoft Windows, які займають основну частку ринку операційних систем. Компанія Microsoft почала роботу над операційною системою з графічним інтерфейсом в 1981 році. Перша версія Windows, випущена в листопаді 1985 року. Вона мала значно менші можливості, ніж Mac OS, і до випуску в 1990-му Windows 3.0 була не дуже відомою.

Ситуація почала змінюватися з виходом версій 3.1 і 3.11, а вдала маркетингова кампанія в середині 90-х років привела до різкого зростання популярності Windows 95 і подальших продуктів цього сімейства як для корпоративних робочих станцій так і для домашніх комп'ютерів.

Найбільш популярними сьогодні є Windows7, Windows8, Windows10.

Linux

Коли в 1991 році фінський студент Лінус Торвальдс починав роботу над Linux, то мало хто чекав такого успіху від чергового клону UNIX. Проте, протягом подальших років тисячі ентузіастів попрацювали над кодом цієї операційної системи, і згодом співробітник Массачусетського технологічного інституту (MIT) Річард Столлмен розробив для Linux проект GNU, який сьогодні перетворився на єдину платформу GNU/Linux і є основою для Open Source (програмування з відкритим кодом).

На даний час Linux має величезну популярність, а вдосконаленням її коду займаються не лише ентузіасти, але і багато крупних виробників комерційного ПЗ, зокрема IBM, Novell, Oracle, Sun Microsystems. Найширше Linux застосовується як серверна платформа, але багато виробників ПО і засобів розробки серйозно розглядають її і як клієнтську операційну систему з двох причин: невисокі вимоги Linux до апаратних ресурсів і відсутність альтернатив Windows на ринку клієнтських операційних систем для найбільш поширеної апаратної платформи.

Спочатку Linux мала лише інтерфейс командного рядка, але зараз для цієї операційної системи існує кілька графічних оболонок, що містять інструменти, які схожі на наявні в Windows (файловий менеджер, засоби для роботи з Інтернетом і електронною поштою, інструменти конфігурації ОС, а також офісні рішення). Це робить її привабливою для кінцевих користувачів.

Сьогодні для Linux створено багато програмного забезпечення для кінцевих користувачів, зокрема, офісні пакети від Sun Microsystems і Corel, графічні пакети, ігри і засоби розробки додатків.

Хоча у початкового коду Linux не існує конкретного правовласника, є ряд компаній, що продають комерційні дистрибутиви операційної системи, які призначені для вирішення конкретних завдань, зокрема, дистрибутиви для домашніх комп’ютерів і для корпоративних робочих станцій. Саме завдяки цим дистрибутивам Linux стала доступною для кінцевих користувачів, які не спеціалізуються в налаштуванні Linux, бо ця процедура раніше була значно складнішою від встановлення і налаштування Windows.

Mac OS і Mac OS X

Mac OS відразу створювалася як операційна система з графічним користувацьким інтерфейсом. Рівень підтримки графіки і мультимедіа в ранніх версіях Mac OS був вищим, ніж у тодішніх версіях Windows, завдяки чому комп'ютери Macintosh досі широко застосовуються у видавничій справі і в суміжних галузях. Саме у Mac OS вперше з'явилася підтримка таких звичних нині пристроїв, як мишка, дисководи для компакт-дисків, технологія Plug and Play, підтримка стандарту під’єднання зовнішніх пристроїв USB, а також прості засоби налаштування мережного доступу.

Проте, список програмного забезпечення і обладнання, що є сумісним з Mac OS, є доволі обмеженим, внаслідок чого вартість експлуатації програмно-апаратних комплексів з Mac OS є набагато вищою, ніж у Windows-сумісних комп'ютерів.

Поява 32-розрядних версій Windows, що мали можливості, які раніше були характерними лише для Mac OS, а також вдала маркетингова політика компанії Microsoft, привела до помітного скорочення частки Mac OS на ринку. Також, в Mac OS у порівнянні з Windows була істотно обмежена підтримка багатозадачності, а це створювало певні незручності в роботі користувачів.

До особливостей останніх версій Mac OS відноситься якісна підтримка роботи з цифровим відео і звуком.

#### 2.2.1.2 Серверні операційні системи

Основним призначенням серверних операційних систем зазвичай є управління застосуваннями, що обслуговують користувачів корпоративної мережі чи зовнішніх користувачів. Це - системи управління базами даних, сервери застосувань, засоби управління мережами, служби каталогів, засоби обміну повідомленнями і групової роботи, веб-сервери, поштові сервери. Вимоги до продуктивності і надійності вказаних операційних систем є дуже високими. Вирішення типових завдань користувача підтримують далеко не всі серверні ОС.

Найбільш дорогі версії серверних операційних систем підтримують кластери (набори однотипних комп'ютерів, між якими поділяється навантаження при виконанні одного завдання), мають засоби резервування та дублювання і здійснюють ці операції без перевантаження операційної системи.

Хоча частка Windows на ринку серверних операційних систем зараз є досить значною, але не такою великою, як на ринку клієнтських ОС. Окрім продукції Microsoft на цьому ринку популярними є серверні операційні системи від IBM, Hewlett-Packard, Novell, Sun Microsystems тощо. Достатньо великою є частка операційних систем з відкритим початковим кодом і самим швидкорослим на ринку серверних операційних систем є Linux-сегмент.

Серверні версії Windows

Windows Server 2003

Створення сімейства Windows Server 2003 стало наступним кроком в розвитку серверних операційних систем Windows 2000. Основними особливостями даного сімейства ОС є наявність в їх складі платформи Microsoft .NET Framework, а також підтримка веб-сервісів XML.

Windows Server 2003 існує в кількох редакціях, що різняться за призначенням, масштабованістю і рівнем доступності, починаючи від редакції для розгортання Web-застосувань і обслуговування невеликих компаній і закінчуючи редакцією з підтримкою симетричної багатопроцесорної обробки з використанням до 32 процесорів для обробки транзакцій в режимі реального часу. Дві старші редакції Windows Server 2003 Enterprise Edition і Windows Server 2003 Datacenter Edition доступні для 32- і 64-розрядних платформ.

Windows Server 2008

Якщо Linux в прагненні наздогнати і перегнати переймає гірші риси Windows, Microsoft активно запозичує все краще з світу UNIX-ів, реалізовуючи їх в своїх серверних системах. Server 2008 є більше схожим на UNIX, ніж останні дистрибутиви Linux.

Проект Server 2008 стартував ще до випуску Server 2003, але перша бета-версія з'явилася лише в липні 2005 року, друга і третя бета вийшла менше ніж через рік - в травні 2006 року і в квітні 2007 року відповідно.

Під управлінням Server 2008 працює багато Інтернет-серверів, які раніше використовували Server 2003/Windows 2000.

Server 2008 засновано на ядрі Vista (яке у свою чергу засноване на коді Server 2003) і тому містить всі основні її функції, зокрема переписаний з нуля TCP/IP-стек, що містить покращену підтримку динамічної пам'яті, файлу підкачки, вводу/виводу, рандомізацію адресного простору і контроль цілісності купи для захисту від переповнених буферів тощо.

Microsoft продовжує підсилювати безпеку і оптимізує систему під роботу з потужним апаратним забезпеченням, що зовсім не йде на користь серверам. В практичному плані кінцеві користувачі вимушені вкладати гроші в апаратуру для отримання тієї ж самої продуктивності, що і на Server 2003/Windows 2000.

До переваг, в першу чергу, можна віднести доопрацювання командного рядка, що дозволяє виконувати 99% операцій з віддаленої машини (тобто, без фізичного доступу до сервера), а також покращені механізми моніторингу, діагностики помилок і відновлення системи після падінь. Нарешті, Server 2008 підтримує технології апаратної віртуалізації Intel/AMD, що дозволяють запускати гостьові операційні системи практично без втрати продуктивності.

У Server 2008 з'явився режим server core, що дозволяє встановлювати систему без графічної оболонки і без бібліотеки .NET Framework, управляючи сервером через командний рядок або видалено через консоль управління (*Microsoft Management Console*). Microsoft, нарешті, відмежувала віконну підсистему від ядра, і подарувала чистий текстовою режим, якого всі від неї чекали.

До складу Server 2008 входить вбудований віртуалізатор, який сконструйовано на базі емулятора Microsoft Virtual РС і який дозволяє запускати кілька операційних систем одночасно! Причому підтримуються не лише ОС від Microsoft, але також UNIX-клони: Linux, xBSD тощо. Природно, на віртуальних машинах можна запускати і саму Windows, що дуже корисно для виявлення вірусів, розгляду підозрілих програм, отриманих з сумнівних джерел, або для різних експериментів з системою, яка у гіршому разі може гуркнути лише віртуальний диск, на якому, окрім неї, немає нічого цінного.

Провайдери можуть використовувати емулятор для надання віртуального хостингу, дозволяючи клієнтам встановлювати свої власні операційні системи, що до цього було небаченою розкішшю. Аналогічним чином відбувається розгортання відмовостійких систем.

Тепер, замість придбання резервного сервера, достатньо запустити кілька віртуальних машин. Якщо впаде одна система, обробку запитів візьме на себе інша. Природно, емулятор страхує лише від програмних збоїв, і тому, в дуже важливих інфраструктурах без апаратного дублювання заліза як і раніше не обійтися, проте подібні інфраструктури займають порівняно невелику частину ринку, та і апаратні відмови трапляються не так вже часто. До головного мінусу віртуалізатора відносять необхідність в заміні старого обладнання.

Microsoft до цих пір не визначилася з політикою ліцензування своїх систем, що запущені на віртуальних машинах, і зараз необхідно платити за кожну інсталяцію, включаючи віртуальну. Очевидно, що апаратна емуляція дозволяє використовувати базову операційну систему як фундамент для решти. Але платити за фундамент, як за повнофункціональну ОС, не використовуючи і 10% її можливостей, смішно.

Набагато вигідно покласти це завдання на FREEBSD або Linux, під які є купа безкоштовних емуляторів, що підтримують апаратну віртуалізацію (наприклад, XEN), і які дозволяють запускати Server 2008 як гостьову операційну систему, внаслідок чого потрібно платити лише за одну інсталяцію, а не за дві.

Server 2008 це останній сервер в лінійці Windows NT, що підтримує x86. Далі будуть розроблятися лише 64-бітові версії.

64-бітові версії побудовано з врахуванням останніх віянь DRM, включаючи обов'язковий підпис всіх драйверів, неможливість модифікації ядра тощо. Тобто, все те, що дозволяє Microsoft витісняти неугодних гравців з ринку, одночасно з цим заграючи з Голівудом і іншими утримувачами авторських прав на медіаконтент, позиціонуючи Windows як систему, захищену від цифрового грабежу.

UNIX та її різновиди

Операційну систему UNIX було створено в кінці 60-х років в Bell Laboratories фірми AT&T. Ядро операційної системи написано на асемблері, а решта частин ОС мовою С. З цієї причини UNIX може переноситися на різні апаратні платформи, що привело до появи як комерційних, так і некомерційних версій даної ОС.

Загальними характеристиками для всіх версій UNIX є багатокористувацький режим із засобами захисту даних від несанкціонованого доступу, реалізація мультипрограмної обробки в режимі розділення часу, використання механізмів віртуальної пам'яті і свопінгу, уніфікація операцій вводу-виводу, ієрархічна файлова система, наявність різноманітних засобів взаємодії процесів, зокрема міжмережного.

Окремо можна визначити таку систему, як SCO UnixWare (зараз SCO OpenServer), яку ми згадували у першій частині другого розділу під час опису систем на прикладі пенсійного фонду. Це версія операційної системи Unix з закритим вихідним кодом, розроблена компанією Santa Cruz Operationruen (SCO) і розповсюджується SCO Group. SCO OpenServer 5, операційна система на основі UNIX System V Release 3, була вперше випущена Santa Cruz Operation в 1992. SCO OpenServer 5 до сих пір продовжує підтримуватися SCO.  
SCO OpenServer 6, операційна система на основі UNIX System V Release 4.2MP, була випущена SCO Group в 2005. У ній були додані підтримка великих файлів, пам'яті, багатопоточність ядра. SCO OpenServer 6 містить ядро ​​UnixWare 7 SVR5, інтегроване з додатками і призначеної для користувача середовищем OpenServer 5.  
SCO придбала права на розповсюдження UnixWare і кодової бази System V Release 4 у Novell в 1995. Частина коду UnixWare була повторно використана в OpenServer.  
2 серпня 2000 SCO оголосила про продаж своїх підрозділів, що займаються розробкою UnixWare і OpenServer компанії Caldera Systems, Inc. Операція була завершена в травні 2001. Частина SCO, Tarantella Division, була перетворена в Tarantella, Inc., яка в 2005 була куплена корпорацією Sun Microsystems. У той же час Caldera поміняла ім'я на Caldera International, а в 2002 на SCO Group.  
SCO Group продовжила розробку і підтримку OpenServer. Останньою версією стала 5.0.7.  
22 червня 2005 було випущено OpenServer 6.0, кодова назва "Legend". SCO OpenServer 6 заснований на ядрі System V Release 5 і включає підтримку багатопотокових додатків на Java, С і С++ через POSIX інтерфейс.

Але нажаль зараз ця система майже повністю відійшла в історію. Одним із показників цього став судовий процес з Linux у березні 2003 року [SCO Group](https://uk.wikipedia.org/w/index.php?title=SCO_Group&action=edit&redlink=1) розпочала судовий процес проти [IBM](https://uk.wikipedia.org/wiki/IBM), стверджуючи, що [IBM](https://uk.wikipedia.org/wiki/IBM) додала до [ядра Лінукс](https://uk.wikipedia.org/wiki/%D0%9B%D1%96%D0%BD%D1%83%D0%BA%D1%81_%28%D1%8F%D0%B4%D1%80%D0%BE%29) код, авторські права на який належать [SCO Group](https://uk.wikipedia.org/w/index.php?title=SCO_Group&action=edit&redlink=1), порушивши таким чином умови ліцензії на використання [Unix](https://uk.wikipedia.org/wiki/Unix). Додатково, [SCO](https://uk.wikipedia.org/w/index.php?title=SCO_Group&action=edit&redlink=1) надіслала лист кільком великим компаніям із попередженням, що використання ними Лінукс без відповідної ліцензії SCO може стати причиною судового переслідування. Цей лист став наслідком появи припущення про можливість появи судових позовів проти рядових користувачів Лінукс. Ця дискусія втягнула в судовий процес [AutoZone](https://uk.wikipedia.org/w/index.php?title=AutoZone&action=edit&redlink=1) та [Red Hat](https://uk.wikipedia.org/wiki/Red_Hat). Питання про те, чи дійсно SCO володіє авторським правом на [Unix](https://uk.wikipedia.org/wiki/Unix), є спірним, і заперечується компанією [Novell](https://uk.wikipedia.org/wiki/Novell). [Але](https://uk.wikipedia.org/wiki/3_%D0%BB%D0%B8%D0%BF%D0%BD%D1%8F) у [2006](https://uk.wikipedia.org/wiki/2006) року окружний суд штату [Юта](https://uk.wikipedia.org/wiki/%D0%AE%D1%82%D0%B0) відхилив майже всі заяви, поданих [SCO Group](https://uk.wikipedia.org/w/index.php?title=SCO_Group&action=edit&redlink=1) проти [IBM](https://uk.wikipedia.org/wiki/IBM).

Серверні версії Linux

На даний час Linux є досить популярною серверною платформою, у тому числі і для комерційного застосування. Список СУБД і серверів застосувань для Linux є досить великим, бо, будучи стурбованими зростанням частки Windows на ринку серверних операційних систем, багато крупних виробників апаратного і програмного забезпечення, зокрема Oracle, IBM, Hewlett-Packard, Novell, почали вкладати чималі засоби в розвиток Linux, створювати серверні застосування для Linux, готові апаратно-програмні серверні рішення на основі Linux, а також реалізовувати сумісність з Linux з комерційними операційними системами. Найбільш показовим прикладом тут, мабуть, є Novell Enterprise Server, що побудовано на основі Novell NetWare і SUSE Linux.

Однією з серйозних переваг ОС Linux є низька вартість її придбання. Хоча ця операційна система є некомерційним продуктом, дистрибутиви Linux, що сертифіковані на сумісність з різним апаратним чи програмним забезпеченням, зазвичай є комерційними і за вартістю наближені до аналогічних версій Windows.

FREEBSD

FREEBSD - це некомерційна версія UNIX, що є доступною для 32- і 64-розрядних платформ Intel, DEC Alpha та інших. Основою FREEBSD є дистрибутив BSD UNIX, що випущений групою дослідження обчислювальних систем Каліфорнійського університету Берклі. Дана операційна система має модулі сумісності із застосуваннями інших версій UNIX і модулями ядра, які динамічно завантажуються і дозволяють додавати під час роботи підтримку нових типів файлових систем, мережних протоколів чи емуляторів без перегенерації ядра.

FREEBSD нерідко використовується Інтернет-провайдерами, а також застосовується як операційна система для корпоративних брандмауерів.

Solaris

Операційна система Sun Solaris входить до числа найвідоміших комерційних версій UNIX і є однією з найбільш популярних платформ для розробки корпоративних рішень: для неї існує близько 12 тис. різних застосувань, зокрема серверів застосувань і СУБД майже всіх провідних виробників. Дана ОС є доступною для платформ ULTRASPARC і x86.

Solaris відповідає вимогам багатьох промислових стандартів і відрізняється високою масштабованістю. У складі Solaris містяться такі ключові застосування для створення корпоративної інфраструктури, як сервер застосувань, служба каталогів, засоби доставки повідомлень, веб-сервер, сервер для створення порталів, а також засоби автоматичного визначення помилок і самовідновлення, засоби віртуалізації, динамічна файлова система, розширені засоби безпеки.

Інші ОС

Mac OS X Server

Серверна операційна система Mac OS X Server, що базується на BSD UNIX. Вона була створена фахівцями компанії Apple спільно з університетськими вченими. У 1999 році версія Mac OS X Server була випущена як OpenSource, що дозволило адаптувати Mac OS X для замовників, а також привернути розробників до подальшого розвитку цієї ОС. Mac OS X Server містить засоби сумісності з мережами на основі Windows, Linux, Mac OS

Багато виробників СУБД і серверів застосувань вже випустили для Mac OS X Server версії своїх продуктів, але на даний момент частка серверів, що працюють під управлінням цієї операційної системи є незначною.

NetWare і Open Enterprise Server

На початку 90-х років Novell NetWare була домінуючою мережною операційною системою і користувалася заслуженою популярністю завдяки своїй надійності. На даний час ця ОС вже не займає лідируючих позицій на ринку серверних ОС. Сьогодні ця операційна система в основному використовується для мережних і файлових серверів, серверів друку і групової роботи. Крім того, дану ОС можна використовувати і як сервер застосувань.

Novell NetWare характеризується високим ступенем надійності і масштабованості, відмінним рівнем безпеки даних, стійкістю до збоїв, здатністю керувати великою кількістю робочих станцій і зовнішніх пристроїв.

HP-UX

ОС HP-UX, розроблено в компанії Hewlett-Packard. Вона орієнтована головним чином на сервери виробництва Hewlett-Packard.

AIX

AIX є клоном UNIX, що створено компанією виробництва IBM. Вона призначена для застосування на серверах IBM pSeries і RS/6000 і є сумісною з Linux, що надає можливість перенесення застосувань між цими платформами.

### 2.2.2 Сервери баз даних

Сервер баз даних (БД) виконує обслуговування і управління базою даних і відповідає за цілісність і збереження даних, а також забезпечує операції введення-виведення при доступі клієнта до інформації.  
  
Архітектура клієнт-сервер складається з клієнтів і серверів. Основна ідея полягає в тому, щоб розміщувати сервери на потужних машинах, а додатків, що використовують мовні компоненти СУБД, забезпечити доступ до них з менш потужних машин-клієнтів за допомогою зовнішніх інтерфейсів.

Більшість СУБД використовують мову SQL (Structured Query Language - мова структурованих запитів), так як він зручний для опису логічних підмножин БД.

ПризначенняSQL:

* Створення БД і таблиці з повним описом їх структури;
* Виконання основних операцій маніпулювання даними (такі як вставка, модифікація і видалення даних з таблиць);
* Виконання простих і складних запитів.

Одна з ключових особливостей мови SQL полягає в тому, що з його допомогою формуються запити, що описують якусь інформацію з бази даних необхідно отримати, а шляхи вирішення цього завдання програма визначає сама.

Кожний сервер БД може працювати на певних типах комп'ютерів і мереж. Операційними системами серверів можуть бути MSDOS, OS / 2, Xenix, Unix, Dec VMS / Робочі станції користувачів зазвичай працюють під управління MSDOS, OS/2, Xenix, Unix.  
  
Існують можливості змішаного використання різних ОС. Велика частина SQL-серверів може зберігати опис БД в системному каталозі, який зазвичай буває доступний користувачам. Для звернення до цього каталогу використовуються SQL-запити. Реляційні СУБД можуть використовувати інформацію, що зберігається в системному каталозі для оптимізації SQL-запитів.

Більшість SQL-серверів підтримують кількість посилальна цілісність реляційних БД, що складаються з окремих таблиць, які можуть бути об'єднані на основі загальної інформації.

#### 2.2.2.1 Oracle

Oracle Database - це об'єктно-реляційна система підтримує деякі технології, які реалізують об'єктно-орієнтований підхід, тобто забезпечують управління створення та використання баз даних.

Сучасна СУБД Oracle це потужний програмний комплекс, що дозволяє створювати додатки будь-якого ступеня складності. Ядром цього комплексу є база даних, що зберігає інформацію, кількість якої за рахунок коштів, що надаються масштабування практично безмежно. C високою ефективністю працювати з цією інформацією одночасно може практично будь-яку кількість користувачів (за наявності достатніх апаратних ресурсів), не проявляючи тенденції до зниження продуктивності системи при різкому збільшенні їх числа.

Механізми масштабування в СУБД Oracle останньої версії дозволяють безмежно збільшувати потужність і швидкість роботи сервера Oracle і своїх додатків, просто додаючи нові й нові вузли кластера. Це не вимагає зупинки працюючих додатків, не вимагає переписування старих додатків, розроблених для звичайної одно-машинної архітектури. Крім того, вихід з ладу окремих вузлів кластера також не приводить до зупинки програми.

Вбудовування в СУБД Oracle JavaVM, повномасштабна підтримка серверних технологій (Java Server Pages, Java-сервлети, модулі Enterprise JavaBeans, інтерфейси прикладного програмування CORBA), призвело до того, що Oracle на сьогоднішній день де-факто є стандартом СУБД для Internet.

Ще однією складовою успіху СУБД Oracle є многоплатформенность, так як вона поставляється практично для всіх існуючих на сьогодні операційних систем. Працюючи під Sun Solaris, Linux, Windows або на іншій операційній системі з продуктами Oracle не виникатиме жодних проблем в роботі. СУБД Oracle однаково добре працює на будь-якій платформі. Таким чином, компаніям, які розпочинають роботу з продуктами Oracle не доводиться міняти вже склалося мережеве оточення. Існує лише невелика кількість відмінностей при роботі з СУБД, обумовлених особливостями тієї чи іншої операційної системи. В цілому ж це завжди та ж сама безпечна, надійна і зручна СУБД Oracle.

СУБД Oracle поставляється в чотирьох варіантах Oracle Database Enterprise Edition, Oracle Database Standard Edition, Oracle Database Personal Edition і зовсім полегшений мобільний варіант, призначений в першу чергу для laptop-ів. При цьому всі варіанти сервера Oracle мають в своїй основі один і той же код і функціонально ідентичні за винятком деяких опцій, які наприклад, можуть бути доступні тільки для Oracle Database Enterprise Edition і не поставлятися з іншими варіантами СУБД.

Oracle включає в себе дуже багато різних компонентів і модулів, ось деякі з них:

* Модуль interMedia забезпечує підтримку всіх типів даних, у тому числі виконання операцій пошуку по великим текстових документів різних форматів.
* Компонент Oracle Enterprise Manager являє собою універсальний засіб адміністрування баз даних, забезпечене зручним графічним інтерфейсом і дозволяє адміністратору баз даних виконувати широкий спектр операцій над безліччю баз даних Oracle, включаючи створення, модифікацію і видалення будь-яких об'єктів внутрікаждой з них.
* Модуль Advanced Replication Option дозволяє виконувати реплікацію даних в широкому діапазоні можливостей, включаючи синхронну, асинхронну, каскадну і інші типи реплікації.
* Модуль Oracle Workflow є засіб для автоматизації стандартних бізнес-процедур організації, для розробки процедур управління потоками робіт. Він пропонує розширені можливості автоматизації проходження і обробки інформації довільного типу і формалізації складних бізнес процедур і алгоритмів обробки інформації.

Одна з ключових можливостей сервера БД Oracle - механізм зберігання і обробки черг повідомлень, який називається Oracle Advanced Queuing (AQ). Він поставляється разом з сервером баз даних, і його не потрібно ліцензувати окремо. Компонент AQ відноситься до класу Message Oriented Middleware (ПО проміжного шару для обробки повідомлень). Наявність такого компонента дозволяє побудувати на базі сервера повнофункціональну інфраструктуру для обробки повідомлень і виключає необхідність придбання для цієї мети додаткових коштів третіх фірм (таких як IBM MQ Series), забезпечуючи, в той же час, зв'язок з ними в неоднорідних середовищах за рахунок продукту Oracle Messaging Gateways.  
  
Починаючи з версії Oracle8i, до складу сервера (в усі редакції) включена віртуальна Java-машина (JServer Enterprise Edition).

Компонент Oracle Obects for OLE надає можливість доступу до баз даних Oracle-додатків, розроблених на C ++, Microsoft Visual Basic, OLE 2.0. Повна підтримка мови макроозначень в Visual Basic дозволяє отримувати дані з Oracle безпосередньо в електронних таблицях Microsoft Excel.

Ключові можливості Oracle Database:

* Real Application Cluster (RAC) забезпечує роботу одного примірника бази даних на декількох вузлах grid, дозволяючи управляти навантаженням і гнучко масштабувати систему в разі потреби.
* Automatic Storage Management (ASM) дозволяє автоматично розподіляти дані між наявними ресурсами систем зберігання даних, що підвищує відмовостійкість системи і знижує загальну вартість володіння (TCO).
* Продуктивність. Oracle Database дозволяє автоматично управляти рівнями сервісу і тиражувати еталонні конфігурації в рамках всієї мережі.
* Прості засоби розробки. Новий інструмент розробки додатків HTML DB дозволить простим користувачам створювати ефективні програми для роботи з базами даних в короткі терміни.
* Самоврядування. Спеціальні механізми Oracle Database дозволяють самостійно перерозподіляти навантаження на систему, оптимізувати і коректувати SQL-запити, виявляти і прогнозування зировать помилки.
* Великі бази даних. Тепер максимальний розмір примірника бази даних Oracle може досягати 8 екзабайт.
* Недорогі серверні системи. Oracle Database може використовувати недорогі однопроцесорні комп'ютери або модульні системи з "серверів-лез".
* У новій версії бази даних реалізована підтримка переносите табличних просторів, система управління потоками даних Oracle Streams і модель розподілених SQL-запитів. Для перенесення існуючих баз даних в середу Grid в них не буде потрібно вносити змін, що дозволяє швидко почати використовувати всі переваги Oracle Database.

#### 2.2.2.2 MS SQL

Microsoft SQL Server — комерційна [система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), що розповсюджується корпорацією[Microsoft](https://uk.wikipedia.org/wiki/Microsoft). Мова, що використовується для запитів — [Transact-SQL](https://uk.wikipedia.org/w/index.php?title=Transact-SQL&action=edit&redlink=1), створена спільно Microsoft та [Sybase](https://uk.wikipedia.org/wiki/Sybase). Transact-SQL є реалізацією стандарту [ANSI](https://uk.wikipedia.org/wiki/ANSI)/[ISO](https://uk.wikipedia.org/wiki/ISO) щодо структурованої мови запитів ([SQL](https://uk.wikipedia.org/wiki/SQL)) із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

Microsoft SQL Server також підтримує [Open Database Connectivity](https://uk.wikipedia.org/wiki/ODBC) (ODBC) — інтерфейс взаємодії застосунків з СУБД. Версія SQL Server 2005 надає можливість підключення користувачів через веб-сервер-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не призначеним для [Windows](https://uk.wikipedia.org/wiki/Windows), кроссплатформенно з'єднуватися з SQL Server. Microsoft також випустила сертифікований драйвер [JDBC](https://uk.wikipedia.org/wiki/JDBC), що дозволяє застосункам під керування [Java](https://uk.wikipedia.org/wiki/Java) (таким як BEA і[IBM Websphere](https://uk.wikipedia.org/wiki/IBM_Websphere)) з'єднуватися з Microsoft SQL Server 2000 і 2005.

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL — це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами. Усі сервери мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

* Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.
* Історія змін: Всі зміни бази даних безперервно передаються користувачам.
* [Синхронізація](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

На сьогодні розроблено кілька версій систем : MS SQL Server-2000, MS SQL Server -2005, MS SQL Server-2008.

На мою думку, особливо варто зазначити Microsoft SQL Server випуску 2008. Це закінчене речення в області баз даних і аналізу даних для швидкого створення масштабованих рішень електронної комерції, бізнес-додатків і сховищ даних. Воно дозволяє значно скоротити час виходу цих рішень на ринок, одночасно забезпечуючи масштабованість, що відповідає найвищим вимогам. У SQL Server включена підтримка мови XML і протоколу HTTP, засоби підвищення швидкодії та доступності, що дозволяють розподілити навантаження і забезпечити безперебійну роботу, функції для поліпшення управління і настройки, що знижують сукупну вартість володіння.

Що до розробки застосунків Microsoft та інші компанії пропонують велику кількість програмних засобів розробки, які дозволяють розробляти [застосунки](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BA%D0%B8) для бізнесу з використанням баз даних Microsoft SQL Server. Microsoft SQL Server 2005 включає також [Common Language Runtime](https://uk.wikipedia.org/wiki/Common_Language_Runtime) (CLR) [Microsoft .NET](https://uk.wikipedia.org/wiki/Microsoft_.NET), що дозволяє застосункам, розробленим на мовах платформи [.ΝΕΤ](https://uk.wikipedia.org/w/index.php?title=.%CE%9D%CE%95%CE%A4&action=edit&redlink=1) (наприклад, [VB.NET](https://uk.wikipedia.org/wiki/VB.NET) або [C#](https://uk.wikipedia.org/wiki/C_Sharp)), реалізовувати [процедури, що зберігаються](https://uk.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0,_%D1%89%D0%BE_%D0%B7%D0%B1%D0%B5%D1%80%D1%96%D0%B3%D0%B0%D1%94%D1%82%D1%8C%D1%81%D1%8F&action=edit&redlink=1) та різні функції. Попередні версії засобів розробки Microsoft використовували лише [API](https://uk.wikipedia.org/wiki/API) для надання функціонального доступу до Microsoft SQL Server.

#### 2.2.2.3 PostgreSQL

PostgreSQL — [об'єктно-реляційна](https://uk.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&action=edit&redlink=1) [система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) (СКБД). Є альтернативою як комерційним СКБД ([Oracle Database](https://uk.wikipedia.org/wiki/Oracle_Database), [Microsoft SQL Server](https://uk.wikipedia.org/wiki/Microsoft_SQL_Server), [IBM DB2](https://uk.wikipedia.org/wiki/DB2) та інші), так і СКБД з відкритим кодом ([MySQL](https://uk.wikipedia.org/wiki/MySQL), [Firebird](https://uk.wikipedia.org/wiki/Firebird), [SQLite](https://uk.wikipedia.org/wiki/SQLite)).

Порівняно з іншими проектами з відкритим кодом, такими як [Apache](https://uk.wikipedia.org/wiki/Apache), [FreeBSD](https://uk.wikipedia.org/wiki/FreeBSD) або [MySQL](https://uk.wikipedia.org/wiki/MySQL), PostgreSQL не контролюється якоюсь однією компанією, її розробка можлива завдяки співпраці багатьох людей та компаній, які хочуть використовувати цю СКБД та впроваджувати у неї найновіші досягнення.

Сервер PostgreSQL написаний на [мові C](https://uk.wikipedia.org/wiki/C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)). Зазвичай розповсюджується у вигляді набору текстових файлів із [сирцевим кодом](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4). Для інсталяції необхідно [відкомпілювати](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%86%D1%96%D1%8F) файли на своєму [комп'ютері](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) і скопіювати в деякий каталог. Весь процес детально описаний в документації.

Варто зазначити функції, вони дозволяють виконувати деякий код безпосередньо сервером бази даних. Ці функції можуть бути написані на [SQL](https://uk.wikipedia.org/wiki/SQL), який має деякі примітивні програмні оператори, такі як галуження та цикли. Але гнучкішою буде функція написана на одній із багатьох мов програмування, з якими PostgreSQL може працювати. До таких мов належать:

Вбудована мова, яка зветься [PL/pgSQL](https://uk.wikipedia.org/w/index.php?title=PL/pgSQL&action=edit&redlink=1), подібна до процедурної мови [PL/SQL](https://uk.wikipedia.org/wiki/PL/SQL) компанії [Oracle](https://uk.wikipedia.org/wiki/Oracle_Corporation).

Мови розробки сценаріїв: [PL/Perl](https://uk.wikipedia.org/w/index.php?title=PL/Perl&action=edit&redlink=1), [PL/Python](https://uk.wikipedia.org/w/index.php?title=PL/Python&action=edit&redlink=1), [PL/Tcl](https://uk.wikipedia.org/w/index.php?title=PL/Tcl&action=edit&redlink=1), [PL/Ruby](https://uk.wikipedia.org/w/index.php?title=PL/Ruby&action=edit&redlink=1), [PL/sh](https://uk.wikipedia.org/w/index.php?title=PL/sh&action=edit&redlink=1).

Класичні мови програмування [C](https://uk.wikipedia.org/wiki/C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [C++](https://uk.wikipedia.org/wiki/C%2B%2B), [Java](https://uk.wikipedia.org/wiki/Java) (за допомогою [PL/Java](https://uk.wikipedia.org/w/index.php?title=PL/Java&action=edit&redlink=1)).

Функції можуть бути написані, а потім виконуватись із привілеями користувача, який її викликав, або із привілеями користувача, який її написав.

Також PostgreSQL підтримує одночасну модифікацію БД декількома користувачами за допомогою механізму Multiversion Concurrency Control (MVCC). Завдяки цьому виконуються вимоги ACID, і практично відпадає потрібність в блокуванні зчитування.

PostgreSQL може виконувати збережені процедури, написані на різних мовах програмування, включаючи Java, Perl, Python, Ruby, Tcl, C / C ++, і власному PL / pgSQL, аналогічному Oracle's PL / SQL. У стандартну бібліотеку функцій включені сотні вбудованих функцій - від базових математичних і строкових операцій до криптографічних функцій і функцій, що забезпечують сумісність з Oracle. Тригери і процедури можуть бути написані на Сі і завантажені в базу даних як бібліотеки, дозволяючи тим самим розширювати її можливості. Також PostgreSQL включає засоби розробки, що дозволяють створювати призначені для користувача типи даних разом з функціями і операторами, що описують їх поведінку. Як результат, створені і можуть бути додані до системи різні типи даних - від геометричних і просторових примітивів до типів даних, визначених у ISBN / ISSN (International Standard Book Number / International Standard Serial Number).

Поряд з безліччю мов, які можуть використовуватися для написання збережених процедур, існує і безліч інтерфейсних бібліотек, що дозволяють як інтерпретується, так і компільовані мов взаємодіяти з PostgreSQL. Це інтерфейси для Java (JDBC), ODBC, Perl, Python, Ruby, C, C ++, PHP, Lisp, Scheme, Qt і ін.

Крім того, вихідний код PostgreSQL доступний під найбільш ліберальної з відкритих ліцензій - ліцензією BSD. Ця ліцензія дає вам право вільно використовувати, модифікувати і поширювати PostgreSQL в будь-якій формі, яка вам подобається, з відкритим або закритим вихідним кодом. Ви можете чинити зі зробленими вами модифікаціями так, як вам буде завгодно. Таким чином, PostgreSQL це не тільки потужна система управління базами даних, що дозволяє забезпечити діяльність організації, але і платформа розробки для побудови додатків, що вимагають використання реляційної СУБД.

Вище були представлені одні з найбільш розповсюджених СУБД, але звичайно, їх набагато більше, і кожна має свої певні недоліки та переваги.

### 2.2.3 WEB-сервера

В даний час на ринку присутні багато рішень від різних виробників, але лідерами є ці 3 програмні продукти:

Apache (сайт - apache.org) - це найпоширеніший і популярний безкоштовний сервер в мережі. Він є більш надійним і гнучким. Сервер не вимогливий до ресурсів процесора і здатний обслуговувати безліч сайтів. Додаток доступний для широкого спектра операційних систем, включаючи Unix, Linux, Solaris, Mac OS X, Microsoft Windows і інші. На даний момент використання Apache становить 71%. Однак, це складна програма, з якою не кожен новачок зможе впорається.

Microsoft IIS (сайт - www.iis.net) - ще один надійний сервер від компанії Microsoft. Він жорстко зміцнився на другому місці з 14% використання в мережі. Після установки програми, будуть підтримуватися тільки дві мови програмування (VBScript і JScript). Однак, можна відкрити додаткові можливості, встановивши для цього потрібні розширення. З установкою таких модулів, функціональність даного сервера значно підвищується.

NGINX (сайт - nginx.org/ru/) - це найбільш популярний веб-сервер в російському Інтернеті. У порівнянні з двома першими, він є найбільш простим і не володіє зайвими функціями. Також його хвалять за надійність і високу швидкість роботи. У 2004 році була випущена перша версія nginx. Зараз цей програмний продукт замикає трійку найпопулярніших веб серверів в світі. Його використання становить близько 6,5%.

Далі розглянемо детальніше кожен з них.

#### 2.2.3.1 Apache

Apache HTTP-сервер — [відкритий](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) [Інтернет](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) для [UNIX](https://uk.wikipedia.org/wiki/UNIX)-подібних, [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows), [Novell NetWare](https://uk.wikipedia.org/wiki/Novell_NetWare) та [інших операційних систем](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0). Apache розроблюється та підтримується спільнотою розробників [відкритого програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) під керівництвом[Apache Software Foundation](https://uk.wikipedia.org/wiki/Apache_Software_Foundation).

В 1996 році Apache обійшов [NCSA HTTPd](https://uk.wikipedia.org/wiki/NCSA_HTTPd) із того часу є найбільш популярним [веб-сервером](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) у світі. Apache встановлений на 53.34% (358 974 045 серверів) для порівняння на другому місці [Microsoft IIS](https://uk.wikipedia.org/wiki/Internet_Information_Services) їхня частка 17.22%

Web-сервер Apache є самостійним, некомерційним, вільно розповсюджуваним продуктом. Продукт підтримує безліч можливостей, багато з яких реалізовані як скомпільовані модулі, які розширюють основні функціональні можливості. Вони різняться від серверної підтримки мов програмування до схем аутентифікації. Існують інтерфейси для підтримки мов програмування [Perl](https://uk.wikipedia.org/wiki/Perl), [Python](https://uk.wikipedia.org/wiki/Python), [Tcl](https://uk.wikipedia.org/wiki/Tcl) і [PHP](https://uk.wikipedia.org/wiki/PHP).

Apache передусім використовується для передачі через [HTTP](https://uk.wikipedia.org/wiki/HTTP) статичних та динамічних веб-сторінок у всесвітній павутині. Багато [веб-застосунків](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) спроектовано, зважаючи на середовище і можливості, які надає цей веб-сервер.

Продукт може працювати в якості кешувального [проксі-сервера](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), що дозволяє істотно підвищити продуктивність роботи користувачів локальної мережі при роботі з документами, розташованими в Інтернет.

Ядро Apache включає в себе основні функціональні можливості, такі як обробка конфігураційних файлів, протокол [HTTP](https://uk.wikipedia.org/wiki/HTTP) і система завантаження модулів. Ядро (на відміну від модулів) повністю розробляється [Apache Software Foundation](https://uk.wikipedia.org/wiki/Apache_Software_Foundation), без участі сторонніх програмістів. Теоретично, ядро apache може функціонувати в чистому вигляді, без використання модулів. Однак, функціональність такого рішення вкрай обмежена. Ядро Apache повністю написано на мові програмування [C](https://uk.wikipedia.org/wiki/C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)).

В архітектурі Apache можна виділити наступні базові об'єкти:

* Сервер.
* Коннект.
* Запит.
* Процес.

Кожен з цих об'єктів представлений відповідною структурою в заголовки httpd.h.

Крім цих 4-х основних об'єктів в архітектурі Apache потрібно відзначити ще два об'єкти. Перший об'єкт - це пули. Управлінням ресурсів в Apache займаються пули - APR pools . Будь-який ресурс, що виділяється динамічно, якщо він прив'язаний до пулу, буде видалений автоматично після використання. Пул прив'язує ресурси до життєвого циклу основних об'єктів, про які ми сказали вище. Другий об'єкт - це конфігураційний масив, причому кожного модуля в ньому відводиться своє місце. Масив зберігає як глобальні конфігураційні дані, так і локальні.

Система конфігурації Apache заснована на [текстових конфігураційних файлах](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB). Має три умовних рівня конфігурації:

* Конфігурація сервера
* Конфігурація віртуального хоста
* Конфігурація рівня директорії

Має власну мову конфігураційних файлів, заснований на блоках директив. Практично всі параметри ядра можуть бути змінені через конфігураційні файли. Більша частина модулів має власні параметри.Частина модулів використовує в своїй роботі конфігураційні файли операційної системи.

Найголовніше, що мені подобається в Apache це досить багато нормальної документації що, звичайно ж, дозволяє спростити освоєння даного сервера. Наприклад, у Apache є офіційний сайт - www.apache.org де, до речі, ви можете завантажити будь-яку версію Apache для будь-якої операційної системи.

Однією з головних особливостей в Apache є те, що розробник сайту, наприклад ви, розмістивши свій сайт на хостингу, можете управляти та змінювати параметри даного сервера без його перезавантаження і без шкоди для інших сайтів, які розташовуються на цьому сервері, це робиться за допомогою файлу .htaccess.

#### 2.2.3.2 Microsoft IIS

Microsoft IIS (Internet Information Services) — це набір [серверів](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80) для декількох служб [Інтернету](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) від компанії [Майкрософт](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82). IIS поширюється з [операційними системами](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) родини [Windows NT](https://uk.wikipedia.org/wiki/Windows_NT).

Основний компонент IIS — [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), який дозволяє розміщувати в Інтернеті [сайти](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82). IIS підтримує [протоколи](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) [HTTP](https://uk.wikipedia.org/wiki/HTTP),[HTTPS](https://uk.wikipedia.org/wiki/HTTPS), [FTP](https://uk.wikipedia.org/wiki/FTP), [POP3](https://uk.wikipedia.org/wiki/POP3), [SMTP](https://uk.wikipedia.org/wiki/SMTP), [NNTP](https://uk.wikipedia.org/wiki/NNTP). IIS другий за популярністю [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) за кількістю [сайтів](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), після [Apache HTTP Server](https://uk.wikipedia.org/wiki/Apache_HTTP_Server). За даними компанії [Netcraft](https://uk.wikipedia.org/w/index.php?title=Netcraft&action=edit&redlink=1) , понад 37.13% сайтів обслуговується веб-сервером IIS.

На даному сервері можна розробляти сайти за допомогою технологій ASP (але вона вже застаріла), ASP.NET так як ця технологія від компанії Microsoft, але любителів інших технологій (мов програмування) компанія Microsoft не забула і зробила можливість прикручувати ці мови до цього WEB сервера , наприклад: PHP, PERL і інші. Найпопулярнішими версіями є IIS-6, IIS-7, IIS-7.5 і найостанніша версія (на сьогоднішній день) IIS-8 (в Windows 8 і Windows Server 2012).

Основними компоненти цього сервера є:

* + Сама web служба;
  + Служба FTP, може, до речі, функціонувати як самостійний сервер, якщо ви раптом хочете налаштувати ftp, але при цьому не використовувати web сервер
  + SMTP сервер, також можете використовувати його як окремий поштовий сервер у себе в організації.

У таких версіях, як Windows 7, Windows Server 2008 існує більш детальна розбивка на окремі компоненти. Треба зауважити що, вище представлена лише загальна частина всіх компонентів.

Багато web-майстрів не хочуть використовувати IIS тільки тому, що думають, що для цього потрібно встановлювати Windows Server там скажімо 2003 або 2008, з одного боку вони мають рацію, тобто якщо ви хочете розміщувати свої сайти на власному сервері (як наприклад хостингові компанії в Інтернеті) то тоді звичайно найкраще встановити серверну операційну систему. Але якщо хочемо просто розробляти сайти, а потім перенести на хостинг в Інтернет, то IIS можна встановлювати і на звичайні клієнтські операційні системи такі як: Windows XP Pro або Windows 7 Pro і вище.

Тепер трохи детальніше розглянемо головні відмінності версій.

Перш за все, IIS 5.0 та вищі підтримують такі механізми [аутентифікації](https://uk.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F):

* Базова аутентифікація(Basic access authentication) — ім'я та пароль передаються в мережі відкритим текстом.
* Стисла аутентифікація(Digest access authentication) — пароль обробляється [хеш-функцією](https://uk.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88%D1%83%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%8F) перед відправленням по мережі, це унеможливлює відтворення пароля у разі перехоплення зловмисником.
* Інтегрована аутентифікація Windows(Integrated Windows Authentication) — виконується спроба аутентифікації на сервері з тими обліковими даними, під якими працює [браузер](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) користувача.
* .NET Passport аутентифікація

Існують кілька відмінностей серверів IIS 6 і IIS 7 (і вище) якщо бути точніше, то переваг 7 версії (і вище) над 6:

* + У 7 версії, на мій погляд, більш зручний інтерфейс;
  + Якщо ви хочете розробляти сайти на PHP, то в 7 (і вище) версії більш зручна інтеграція і управління php розширенням;
  + IIS 6 з самого початку не був орієнтований для хостингів, а IIS 7 (і вище) вже орієнтований, наприклад: здатність власникам сайтів управляти і конфігурувати свої сайти (мається на увазі web сервер) без звернення в техпідтримку і без використання додаткових скриптів у хостера, так як на версії IIS 6 хостери використовували додаткові скрипти щоб в ручну не налаштовувати, виконувати прохання власників сайтів. IIS6 був орієнтований тільки на корпоративні web сайти, але його почали використовувати і хостери тобто він набув популярності. IIS7 відмінний варіант для тих хто любить Windows і windows програми та він уже нічим не поступається найпопулярнішому web сервера в інтернеті apache.

На сьогоднішній день в Інтернеті з серверів IIS найпопулярнішим поки залишається 6 версія але 7 версія вже майже наздогнала і все web майстра і хостери вже активно переходять на 7 або вище версію IIS.

#### 2.2.3.3 Nginx

Nginx (engine x) — [вільний](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) і [проксі-сервер](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). Є версії для сімейства [Unix](https://uk.wikipedia.org/wiki/Unix)-подібних [операційних систем](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) ([FreeBSD](https://uk.wikipedia.org/wiki/FreeBSD),[GNU/Linux](https://uk.wikipedia.org/wiki/GNU/Linux), [Solaris](https://uk.wikipedia.org/wiki/Solaris_(%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)), [Mac OS X](https://uk.wikipedia.org/wiki/Mac_OS_X)) та [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows).

В nginx робочі процеси обслуговують одночасно безліч з'єднань, мультиплексуючи їх викликами операційної системи select, epoll ([Linux](https://uk.wikipedia.org/wiki/Linux)) і kqueue ([FreeBSD](https://uk.wikipedia.org/wiki/FreeBSD)). Робочі процеси виконують цикл обробки подій від дескрипторів. Отримані від клієнта дані розбираються з допомогою кінцевого автомата. Розібраний запит послідовно обробляється ланцюжком модулів, що задається конфігурацією. Відповідь клієнту формується в буферах, які зберігають дані або в [пам'яті](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0_%D0%BF%D0%B0%D0%BC%27%D1%8F%D1%82%D1%8C), або вказують на відрізок файлу. Буфери об'єднуються в ланцюжки, що визначають послідовність, в якій дані будуть передані клієнтові. Якщо операційна система підтримує ефективні операції вводу-виводу, такі як writev і sendfile, то nginx застосовує їх при нагоді.

Конфігурація [HTTP-сервера](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) 'nginx' розділяється на віртуальні сервери (директива server). Віртуальні сервери поділяються на локації (location). Для віртуального сервера можливо задати адреси і порти, на яких будуть прийматися з'єднання, а також імена, які можуть включати \* для позначення довільній послідовності в перший і останній частині, або задаватися регулярним виразом. Локації можуть задаватися точним [URI](https://uk.wikipedia.org/wiki/URI), частиною [URI](https://uk.wikipedia.org/wiki/URI), або [регулярним виразом](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D1%96_%D0%B2%D0%B8%D1%80%D0%B0%D0%B7%D0%B8). location'и можуть бути налаштовані для обслуговування запитів зі статичного файлу, [проксування](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) на http, fastcgi чи memcached сервер.

Для ефективного управління пам'яттю nginx використовує [пули](https://uk.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%B8%D0%B9_%D0%BF%D1%83%D0%BB&action=edit&redlink=1). Пул — це послідовність попередньо виділених блоків динамічної пам'яті. Довжина блоку змінюється в дапазоні від 1 до 16 кілобайт. Спочатку під пул виділяється лише один блок. Блок розділяється на зайняту область і незайняту. Виділення дрібних об'єктів виконується шляхом просування покажчика на незайняту область з урахуванням вирівнювання. Якщо незайнятої області в усіх блоках бракує для виділення нового об'єкта, то виділяється новий блок. Якщо розмір виділеного об'єкта перевищує значення константи NGX\_MAX\_ALLOC\_FROM\_POOL, або довжину блоку, то він повністю виділяється з купи.

Таким чином, дрібні об'єкти виділяються дуже швидко і мають накладні витрати тільки на [вирівнювання](https://uk.wikipedia.org/wiki/%D0%92%D0%B8%D1%80%D1%96%D0%B2%D0%BD%D1%8E%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

До основних функцій можна віднести:

HTTP-сервер

* обслуговування статичних запитів, індексних файлів, автоматичне створення списку файлів, кеш дескрипторів відкритих файлів
* акселероване [проксіювання](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BA%D1%81%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) з підтримкою [кешування](https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%88)
* акселерована підтримка [FastCGI](https://uk.wikipedia.org/wiki/FastCGI) і [memcached](https://uk.wikipedia.org/wiki/Memcached) серверів, простий розподіл навантаження і відмовостійкість
* модульність, фільтри, [gzip](https://uk.wikipedia.org/wiki/Gzip), byte-ranges (докачка), chunked відповіді, HTTP-[аутентифікація](https://uk.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F), [SSI](https://uk.wikipedia.org/wiki/SSI_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F))-фільтр
* вкладені запити на одній сторінці виконуються паралельно
* підтримка [SSL](https://uk.wikipedia.org/wiki/SSL)
* експериментальна підтримка вбудованого [Perl](https://uk.wikipedia.org/wiki/Perl)
* експериментальна підтримка [HTTP/2](https://uk.wikipedia.org/w/index.php?title=HTTP/2&action=edit&redlink=1)

IMAP/POP3-проксі сервер

* перенаправлення користувача на [IMAP](https://uk.wikipedia.org/wiki/IMAP)/[POP3](https://uk.wikipedia.org/wiki/POP3)-бекенд з використання зовнішнього [HTTP](https://uk.wikipedia.org/wiki/HTTP)-сервера аутентифікації
* проста аутентифікація (LOGIN, USER/PASS)
* підтримка [SSL](https://uk.wikipedia.org/wiki/SSL) і [StartTLS](https://uk.wikipedia.org/wiki/StartTLS)

Враховуючи вище вказані дані, на мою думку головні особливості Nginx це: простота, швидкість, надійність. Загалом, також як і перераховані вище web сервера, це відмінний web сервер.

## **2.3 Рішення проблеми для адміністрування медичних карт пацієнтів**

Головна ідея, на якій буде базуватися розробка даного веб-застосунку, створення єдиної бази даних, у якій будемо зберігати всі данні о пацієнтах. Суть ідеї полягає у тому, що доступ з будь-якого підрозділу здійснюється через веб-інтерфейс, для кожного окремого користувача, до цієї бази даних. З можливістю заносити та завантажувати звідти документи. Які звісно належать користувачу.

У підрозділі 2.1 одним з рішень приводився приклад системи пенсійного фонду. Даний веб-застосунок частково буде відтворено за її прикладом, але з поліпшеннями. Зокрема можна визначити такі:

* 1. Відритий код, у якому завжди можна змінити правила доступу або заточити застосунок під потреби поточного власника.
  2. Незалежність від платформи, завдяки якій зміни обладнання та вихід ліпших версій серверних операційних систем, або перехід системи на інше програмне забезпечення, не буде впливати на, так би мовити, моральну актуальність даного програмного продукту. А це у свою чергу дає і постійну матеріальну актуальність.
  3. Територіальна незалежність користувача. Доступ буде здійснюватися лише за умови наявності акаунту у даній системі, браузеру і інтернет підключенню.
  4. Захищена передача даних. Звичайно, будь-яку систему можна зламати і дістати необхідні дані, але ми забезпечимо максимальну безпечність, за умови мінімальних затрат ресурсів.
  5. Автоматичне розподілення документів на серверах в залежності від вихідного користувача, дати і підрозділу.
  6. Виключення можливості потрапляння до бази даних пошкоджених під час передачі документів, завдяки алгоритму хешування MD5 (Message Digest 5). Як тільки документ буде відправлений на сервер, одразу ж відбувається перевірка на збіги кодів MD5. І якщо частини документу були загублені під час транспортування, документ не буде покладено до бази.

Не дивлячись на те, що поліпшення користування і транспортування забезпечило більш високий рівень безпеки, можуть виникнути питання з приводу безпеки документів вже безпосередньо у базі даних. Наприклад, як застерегти документи від втручання сторонніх людей або від шкідливого впливу людей, які вже знаходяться у системі, але можуть щось випадково видалити, зламати та безліч інших варіантів невиправної шкоди, яку вони можуть гіпотетично спричинити.

Перш за все, безпека доступу буде забезпечуватися ще на рівні реєстрації. Як тільки користувач зареєструвався і адміністратор підтвердив його існування та дозволив доступ до загальних файлів, він (користувач) має право завантажувати та посилати власні документи на сервер, які будуть автоматично перевірені на цілісність та розподілені у базі даних у окрему папку згідно з датою відправлення, категорією документів, підрозділу та інформацією про відправника. Користувач не має вільного доступу до заміни будь-яких документів, окрім своїх, на власному ПК. Він може тільки відправити запит на отримання файлів з інших підрозділів. Таким чином ми інкапсулюємо файли від несанкціонованого доступу. Якщо ж щось трапиться на стороні адміністратора, не потрібно буде багато часу, для визначення людини, яка задала шкоду. Тому, при відповідальному підході до вибору співробітника на керуючу роль, можна уникнути 90% неприємностей.

Може ще виникнути проблема випадкового відправлення документів не туди, куди було потрібно. Це саме та шкода від людського фактору, якої майже неможливо уникнути. Єдине що, завжди можна мінімізувати втрати від таких дій. У даному дипломному проекті це зроблено чином прозорого і інформативного збереження документів в базі даних, це реалізується шляхом деталізованого опису інформації про файл. Завжди буде відомо звідки, коли і, найголовніше, від кого надійшли файли. А отже проблема буде вирішена майже одразу по її виявленню. Обираючи стратегію керування базою даних та документообігом, було вирішено, що саме вище вказаний підхід має найбільший шанс на успішну реалізацію, за умови наявних ресурсів і ментальних особливостей потенційних користувачів.

Також, важливо зазначити одну з головних переваг даного програмного продукту по відношенню до оригінальної версії (за оригінал беремо приклад реалізації задачі пенсійним фондом). Як було вже вище сказано, реалізація відбувалася на зв’язці систем SCO UnixWare з закритим кодом та FTP-протоколом. Тож, доступ для кожного нового користувача прописувався через нові труднощі та обхідні шляхи, адже правила доступу закриті від очей програмістів. Наш програмний продукт вирішує це питання і дає, не тільки можливість змінювати правила доступу на свій розсуд, але і значно спрощує шляхи доступу до бази даних.

Як можемо побачити на рисунку 2.3.1, архітектура документообігу доволі проста і у той же час надає можливість раціонального управління всією системою без надлишкового використання людських і матеріальних ресурсів. Всі підрозділи можуть вільно спілкуватися між собою та передавати документи, які зберігатимуться на сервері. Таким чином, уникаючи використання посередників і прискорюючи роботу персоналу.

Підрозділ 1

Сервер

Web

База даних

Web PC

­­

Підрозділ 2

Центральне управління

Web PC

Web

Admin

Підрозділ N

Web PC

Рисунок 2.3.1 Архітектура шляхів документообігу та загальне управління

На період дипломування, даний програмний продукт буде представлено на тестових версіях бази даних та серверу. Подальші апгрейди та повну розгорнуту версію буде представлено вже на реальній системі замовника. Актуальність і систематику покращення якості праці можна буде продивлятися вже через тиждень після підключення і введення продукту в експлуатацію.

### 2.3.1 Формування бази даних та її нормалізація

Визначимо основні сутності(майбутні таблички), з якими будемо працювати. Документ – «зовнішня» інформація о файлі, за якою буде відбуватися розподілення у базі даних. Інформація о файлі – саме тут буде розміщене посилання, де розташований файл, MD5 коди, наявність архівації. Реквізити –один з аспектів файлової інформації, де вказана дата створення та зміни файлу. Логін – інформація про зареєстрованого користувача, із вказанням його персональних даних. Підрозділ, категорія, відділ – внутрішня ієрархічна структура документообігу.

Представимо дані у таблицях та поступово нормалізуємо їх за нормальними формами.

Далі будемо оперувати такими поняттями:

Нормалізація схеми бази даних — покроковий процес розбиття одного відношення (на практиці: таблиці) відповідно до алгоритму нормалізації на декілька відношень на базі функціональних залежностей.

Атрибут - властивість деякої сутності. Часто називається полем таблиці.

Домен атрибута – скінченна множина допустимих значень, які може приймати атрибут.

Кортеж - скінченна множина взаємопов'язаних допустимих значень атрибутів, які разом описують деяку сутність (рядок таблиці).

Ставлення - скінченна множина кортежів (таблиця).

Схема відносини - скінченна множина атрибутів, що визначають деяку сутність. Іншими словами, це структура таблиці, що складається з конкретного набору полів.

Проекція - відношення, отримане з заданого шляхом видалення і (або) перестановки деяких атрибутів.

Функціональна залежність між атрибутами (множинами атрибутів) X і Y означає, що для будь-якого допустимого набору кортежів в даному відношенні: якщо два кортежу збігаються за значенням X, то вони збігаються за значенням Y. Наприклад, якщо значення атрибута «Назва компанії» - Canonical Ltd , то значенням атрибута «Штаб-квартира» в такому кортежі завжди буде Millbank Tower, London, United Kingdom. Позначення: {X} -> {Y}.

Нормальна форма - вимога до структури таблиць в теорії реляційних баз даних для усунення з бази надлишкових функціональних залежностей між атрибутами (полями таблиць).

Метод нормальних форм (НФ) полягає в зборі інформації про об'єктах рішення задачі в рамках одного відносини і подальшої декомпозиції цього відносини на кілька взаємопов'язаних відносин на основі процедур нормалізації відносин.

Мета нормалізації: виключити надмірне дублювання даних, яке є причиною аномалій, що виникли при додаванні, редагуванні і видаленні кортежів (рядків таблиці).

Аномалією називається така ситуація в таблиці БД, яка призводить до протиріччя в БД або істотно ускладнює обробку БД. Причиною є зайве дублювання даних в таблиці, яке викликається наявністю функціональних залежностей від не ключовим атрибутів.

Аномалії-модифікації проявляються в тому, що зміна одних даних може спричинити перегляд всієї таблиці і відповідну зміну деяких записів таблиці.

Аномалії-видалення - при видаленні будь-якого кортежу з таблиці може пропасти інформація, яка не пов'язана на пряму з видаляється записом.

Аномалії-додавання виникають, коли інформацію в таблицю не можна помістити, поки вона не повна, або вставка запису вимагає додаткового перегляду таблиці.

#### 2.3.1.1 Створення таблиць

Створимо таблиці за вищевказаними сутностями, одразу додавши їх типи, та визначимо первинні та зовнішні ключі.



Рисунок 2.3.2 Таблиця «Документ»

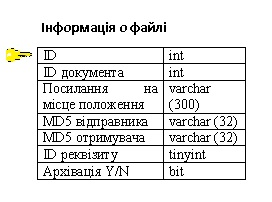


Рисунок 2.3.3 Таблиця « Інформація о файлі»

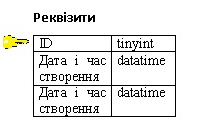


Рисунок 2.3.4 Таблиця « Реквізити»



Рисунок 2.3.5 Таблиця «Логін»

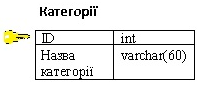


Рисунок 2.3.6 Таблиця «Категорії»

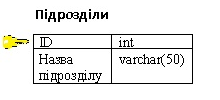


Рисунок 2.3.7 Таблиця «Категорії»

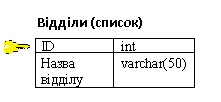


Рисунок 2.3.8 Таблиця «Відділи»

#### 2.3.1.2 Поєднання та нормалізація

Об’єднаємо наші таблиці та подивимось на скільки вони відповідають нормальним формам.

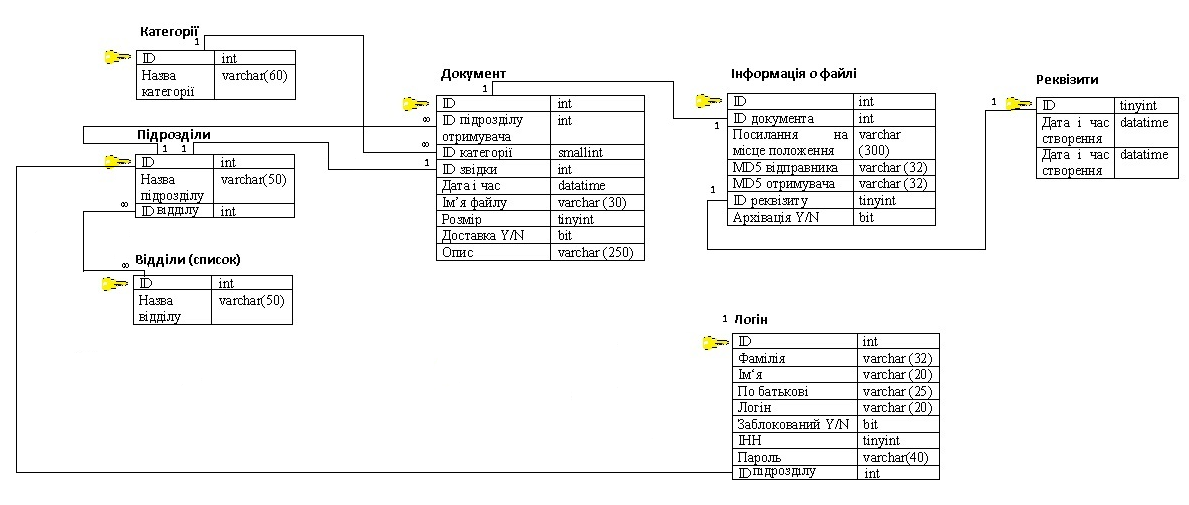


Рисунок 2.3.9 Об’єднання таблиць

Тепер перевіримо на відповідність нормальним формам.

Перша нормальна форма

[Перша нормальна форма](https://uk.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%88%D0%B0_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%84%D0%BE%D1%80%D0%BC%D0%B0) (1НФ, 1NF) утворює ґрунт для структурованої схеми бази даних:

* Кожна таблиця повинна мати основний ключ: мінімальний набір колонок, які ідентифікують запис.
* Уникнення повторень груп (категорії даних, що можуть зустрічатись різну кількість разів в різних записах) правильно визначаючи неключові атрибути.
* Атомарність: кожен атрибут повинен мати лише одне значення, а не множину значень.

Таблиця відповідає 1НФ , має основний ключ, усті атрибути атомарні, дані не повторюються.

Друга нормальна форма

[Друга нормальна форма](https://uk.wikipedia.org/wiki/%D0%94%D1%80%D1%83%D0%B3%D0%B0_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%84%D0%BE%D1%80%D0%BC%D0%B0) (2НФ, 2NF) вимагає, аби дані, що зберігаються в таблицях із композитним ключем не залежали лише від частини ключа:

* [Схема бази даних](https://uk.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) повинна відповідати вимогам першої нормальної форми.
* Дані, що повторно з'являються в декількох рядках виносяться в окремі таблиці.

Таблиця відповідає 2НФ, адже задовольняє вимоги 1НФ, немає даних, що з'являються повторно у декількох рядках.

Третя нормальна форма

[Третя нормальна форма](https://uk.wikipedia.org/wiki/%D0%A2%D1%80%D0%B5%D1%82%D1%8F_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%84%D0%BE%D1%80%D0%BC%D0%B0) (3НФ, 3NF) вимагає, аби дані в таблиці залежали винятково від основного ключа:

* [Схема бази даних](https://uk.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) повинна відповідати всім вимогам другої нормальної форми.
* Будь-яке поле, що залежить від основного ключа та від будь-якого іншого поля, має виноситись в окрему таблицю.

Таблиця відповідає першій, але не другій умові. Адже у кожного підрозділу може бути більше одного відділу і вони будуть повторюватись у таблиці, а отже залежність від первинного ключа неповна. Для рішення цієї проблеми винесемо в окрему таблицю відношення підрозділу і відділів, яка буде суміжним ключем. Якщо первинний ключ складається з двох і більше атрибутів, його називають суміжним ключем.

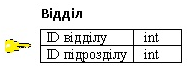


Рисунок 2.3.10 Суміжний ключ «Відділ»

Важливо зазначити, що у кожний користувач, за вимогою замовника, може мати лише один акаунт.

Для покращення подальшої взаємодії з запитами додамо можливість швидкого доступу до користувача, додавши таблицю «Користувач»



Рисунок 2.3.11 Суміжний ключ «Користувач»

Можна було б і надалі покращувати структуру бази даних, шляхом декомпозиції та подальшого редагування за нормальними формами, але для збереження гнучкості системи та спрощення вірогідних змін у майбутньому, було вирішено обмежити нормалізацію лише трьома етапами. Це суттєво не вплине на роботу бази даних, не буде викликати проблем із надмірністю і дублюванням даних, завдяки попереднім етапам нормалізації та характерною особливістю інформації, з якою буде взаємодіяти база даних у майбутньому.

На рисунку 2.3.12 зображено остаточну версію схеми бази даних.

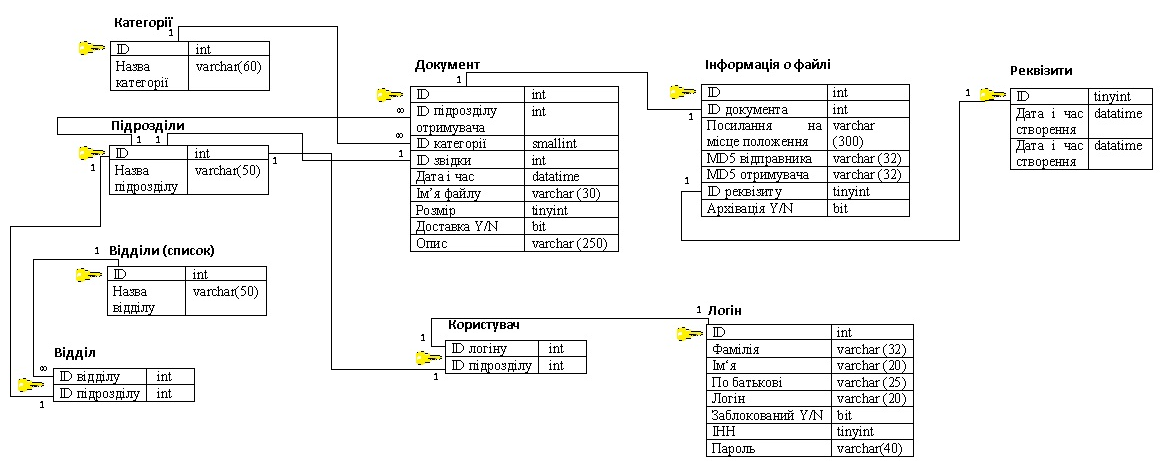


Рисунок 2.3.12 Остаточна версія відношень таблиць

## **2.4 Використані технології**

У даному дипломному проекті було вирішено використовувати такі програмні продукти:

Сервер:

* Операційна система: CentOs 7
* СКБД: MySql Workbench 6.3.4
* WEB-сервер додатку: Apache Tomcat 9.0
* Фреймворк: Spring Framework

Клієнт:

* Операційна система: Windows 7
* WEB: будь-який браузер, який підтримує Java

### 2.4.1 CentOs

CentOS (Community ENTerprise Operating System)  — вільно доступний [дистрибутив Лінукс](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D1%82%D1%80%D0%B8%D0%B1%D1%83%D1%82%D0%B8%D0%B2_%D0%9B%D1%96%D0%BD%D1%83%D0%BA%D1%81) на основі комерційного дистрибутиву [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux) компанії [Red Hat](https://uk.wikipedia.org/wiki/Red_Hat). Проект доклав зусиль, щоб зібрати 100% двійково сумісний зі своїм предком дистрибутив у плані шляху його розвитку, і старається розвиватися з ним в один бік, не відступаючи від його основної лінії та модифікацій, щоб не змінювати своєї мети.

[Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux) складений переважно з [вільного](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) та [відкритого програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), але наявний у доступній для використання, двійковій формі (наприклад, на CD або DVD дисках) лише для передплатних користувачів. Як і вимагається, [Red Hat](https://uk.wikipedia.org/wiki/Red_Hat) випускає усі [сирцеві тексти](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D1%96_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%B8) своїх продуктів під [GNU General Public License](https://uk.wikipedia.org/wiki/GPL) та іншими вільними ліцензіями. Розробники CentOS використовують цей [сирцевий код](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) для створення кінцевого продукту, котрий є дуже подібним до [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux) і вільним для завантаження та використання, однак без відповідної технічної підтримки з боку компанії [Red Hat](https://uk.wikipedia.org/wiki/Red_Hat). Існують й інші дистрибутиви, що базуються на сирцевих текстах [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), однак жоден з них не досяг такого рівня спільноти; загалом CentOS — єдиний дистрибутив, котрий йде у ногу із змінами, що вносяться до [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux).

CentOS віддав перевагу програмному забезпеченню для оновлення на основі [yum](https://uk.wikipedia.org/wiki/Yellow_Dog_Updater_Modified), хоча підтримка [up2date](https://uk.wikipedia.org/w/index.php?title=Up2date&action=edit&redlink=1) також присутня. Можна використовувати для звантаження та встановлення як додаткові пакунки і залежності, так і спеціальні та періодичні оновлення безпеки з репозиторію на [CentOS Mirror Network](http://www.centos.org/modules/tinycontent/index.php?id=13).

CentOS придатний для використання базованих на [X Window](https://uk.wikipedia.org/wiki/X_Window) стільницях, однак більш звично використовувати його, як серверну операційну систему для [веб-хостингу](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%85%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3). Багато великих хостингових компаній використовують CentOS разом із[cPanel Control Panel](https://uk.wikipedia.org/w/index.php?title=CPanel&action=edit&redlink=1) для забезпечення стабільної роботи для своїх [веб-застосунків](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA). У січні 2014 року, CentOS оголосила про приєдання до [Red Hat](https://uk.wikipedia.org/wiki/Red_Hat), але CentOS все ще буде незалежною від RHEL, керівництво CentOS зміниться.

Номер версії CentOS складається з двох частин, головної (major) та незначної (minor). Головна версія рівна відповідній версії [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), з якої беруться джерельні пакунки для побудови CentOS. Незначна версія відповідає номеру випуску модифікацій для [Red Hat Enterprise Linux](https://uk.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), з якого беруться пакунки для побудови відповідної версії CentOS. Наприклад, CentOS 5.3 (актуальна нині) базується на Red Hat Enterprise Linux 5 update 3. У нашій роботі ми викиростовуємо версію CentOS 7.

Важлива особливість Centos - на даній ОС працюють всі програмні продукти, які розраховані на Linux. Система підтримується ентузіастами, але тим не менш нові версії виходять у світ кожні два роки і постійно оновлюються. Процес установки Centos простий і зрозумілий навіть для звичайного користувача, і це зовсім не означає низкопробность системи, навпаки Centos - повноцінна ОС.

Характеристики системи: поновлення даної ОС отримують через програму yum; співтовариство підтримки, яке швидко зростає, має систему збирання, тесту і виправлення помилок; здійснюється підтримка через форуми, є великий і зрозумілий FAQ; є можливість купити платне супровід.

Один з найпопулярніших і актуальних релізів - Centos 7 (всього на даний момент налічують 5 версій ОС). Які основні зміни були внесені в дану версію в порівнянні з попередніми релізами:

* Оновлення ядра до 3.10.0
* Забезпечення Linux Containers
* Open VMware Tools і графічні 3D драйвери «з коробки»
* За замовчуванням налаштовані OpenJDK-7 - JDK
* Забезпечення оновлення з версії 6.5 до 7.0
* Снапшоти LVM з ext4 і XFS
* Перехід на systemd, firewalld і GRUB2
* XFS - файлова система за замовчуванням
* iSCSI і FCoE в ядрі
* забезпечення PTPv2
* Підтримка мережевих карт 40G Ethernet
* Забезпечення установки в режимі UEFI Secure Boot на підтримуваному обладнанні.

Основні особливості ОС Centos в цілому. В першу чергу, Centos привертає користувачів своєї безкоштовністю на відміну RHEL, яка надається на комерційній основі. З боку технічних аспектів виділяють оперативність репозиторіїв RHEL на високому рівні, чим забезпечується безпека системи. Конструктор GCC як SSP (захист стека), PIE. Набір ПО актуальний і типовий для сучасних ОС: надаються версії офісних, серверних і девелоперських пакетів, програм і утиліт (KDE та Gnome з compiz і AIGLX, Firefox і Evolution, MySQL і PostgreSQL, Apache і PHP, і т.д.). Також надається докладна технічна документація і є великий штат підтримки ОС, до якого можна звернутися і отримати відповіді на всі запитання по даній системі.

З приводу недоліків системи, користувачі часто наголошують на тому, що дистрибутив комплектується не завжди свіжими версіями програм, в тому числі ядро ​​Linux теж не завжди нове. Тому дана система не підходить для тих, хто любить щоденні оновлення. Хоча будь-яку систему можна оновити на свій "смак", і цей недолік не буде вважатися таким істотним.

### 2.4.2 MySql Workbench

MySQL - вільна реляційна система управління базами даних. Розробка та підтримка сайта MySQL здійснює корпорація Oracle, яка отримала права на торговельну марку разом з поглиненої Sun Microsystems, яка раніше придбала шведську компанію MySQL AB. Продукт поширюється як під GNU General Public License, так і під власною комерційною ліцензією. Крім цього, розробники створюють функціональність за замовленням ліцензійних користувачів. Саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації.

MySQL є рішенням для малих і середніх додатків. Входить до складу серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP, VertrigoServ. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі і GPL-ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць. Програмне забезпечення MySQL - це ПЗ з відкритим кодом.

ПО з відкритим кодом означає, що застосовувати і модифікувати його може будь-хто. Таке ПО можна отримувати по Internet і використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб. Використання програмного забезпечення MySQL регламентується ліцензією GPL (GNU General Public License), в якій зазначено, що можна і чого не можна робити з цим програмним забезпеченням в різних ситуаціях. Якщо робота в рамках GPL вас не влаштовує або планується вбудовування MySQL-коду в комерційне додаток, є можливість купити комерційну ліцензовану версію у компанії MySQL AB.

Спочатку сервер MySQL розроблявся для управління великими базами даних з метою забезпечити більш високу швидкість роботи в порівнянні з існуючими на той момент аналогами. І ось уже протягом кількох років даний сервер успішно використовується в умовах промислової експлуатації з високими вимогами. Незважаючи на те що MySQL постійно вдосконалюється, він вже сьогодні забезпечує широкий спектр корисних функцій. Завдяки своїй доступності, швидкості і безпеки MySQL дуже добре підходить для доступу до баз даних по інтернету.

Але реалізація проводилась у програмному коді, тому окрім СУБД нам потрібний інструмент зв’язку, який поєднає сутності у коді, з реальними таблицями і відношеннями у базі даних. За такий інструмент було обрано бібліотеку Hibernate.

Hibernate - бібліотека для мови програмування Java, призначена для вирішення завдань об'єктно-реляційного відображення (object-relational mapping - ORM). Вона являє собою вільне програмне забезпечення з відкритим вихідним кодом (open source), яке розповсюджується на умовах GNU Lesser General Public License. Дана бібліотека надає легкий у використанні каркас (фреймворк) для відображення об'єктно-орієнтованої моделі даних в традиційні реляційні бази даних.

Метою Hibernate є звільнення розробника від значного обсягу порівняно низкоуровневого програмування щодо забезпечення зберігання об'єктів в реляційної базі даних. Розробник може використовувати Hibernate як в процесі проектування системи класів і таблиць «з нуля», так і для роботи з вже існуючою базою даних.

Hibernate не тільки вирішує завдання зв'язку класів Java з таблицями бази даних (і типів даних Java з типами даних SQL), а й також надає кошти для автоматичної генерації і оновлення набору таблиць, побудови запитів і обробки отриманих даних і може значно зменшити час розробки, яке зазвичай витрачається на ручне написання SQL- і JDBC-коду. Hibernate автоматизує генерацію SQL-запитів і звільняє розробника від ручної обробки результуючого набору даних і перетворення об'єктів, максимально полегшуючи перенесення (портирование) додатки на будь-які бази даних SQL.

Hibernate забезпечує прозору підтримку збереження даних (persistence) для «POJO» (тобто для стандартних Java-об'єктів); єдине сувора вимога для зберігається класу - наявність конструктора за замовчуванням (без параметрів). Для коректної поведінки в деяких додатках потрібно також приділити увагу методам equals () і hashCode ().

Для використання запитів, замість звичного нам язику SQL використовуємо HQL. Hibernate забезпечує використання SQL-подібного мови Hibernate Query Language (HQL), який дозволяє виконувати SQL-подібні запити, записані поруч з об'єктами даних Hibernate. Запити критеріїв надаються як Об'єктно-орієнтована альтернатива до HQL.

### 2.4.3 Apache Tomcat

Tomcat (у старих версіях - Catalina) - контейнер сервлетів з відкритим вихідним кодом, що розробляється Apache Software Foundation. Реалізує специфікацію сервлетів і специфікацію JavaServer Pages (JSP) і JavaServer Faces (JSF). Написаний на мові Java.  
  
Tomcat дозволяє запускати веб-додатки, містить ряд програм для самоконфігурірованія. Tomcat використовується в якості самостійного веб-сервера, як сервер контенту в поєднанні з веб-сервером Apache HTTP Server, а також в якості контейнера сервлетів в серверах додатків JBoss і GlassFish.

До компонентів томкату можна віднести три основних:

1. Catalina - контейнер сервлетів Tomcat'а, який реалізує специфікацію сервлетів Servlet API. Servlet API є основою для всіх інших технологій Java, що стосуються Web і дає можливість динамічно генерувати будь-який web-контент, використовуючи будь-які бібліотеки, доступні для java. Архітектором Catalina був Craig McClanahan.  
  
2. Coyote - компонент стека HTTP Tomcat'а, який підтримує протокол HTTP 1.1 для веб-серверів або контейнера додатків. Coyote прослуховує вхідні з'єднання на певному TCP порту сервера, пересилає запити в механізм Tomcat для обробки запитів і відправляє відповідь назад запитувачу клієнту.  
  
3. Jasper - механізм JSP Tomcat'а. Tomcat 5.x використовує Jasper 2, який є реалізацією специфікації JavaServer Pages 2.0 Sun Microsystems. Jasper аналізує JSP-файли, щоб компілювати їх в Java код, як сервлети (які можуть бути оброблені за допомогою Catalina). Під час виконання, Jasper може автоматично виявляти зміни JSP-файлу і перекомпілювати його.

### 2.4.4 Spring Framework

Spring Framework (або коротко Spring) - універсальний фреймворк з відкритим вихідним кодом для Java-платформи. Особливості ядра Spring Framework застосовні в будь-якому Java-додатку, і існує безліч розширень і удосконалень для побудови веб-додатків на Java Enterprise платформі. З цих причин Spring набув великої популярності і визнається розробниками як стратегічно важливий фреймворк. Центральною частиною Spring Framework є контейнер Inversion of Control, який надає кошти конфігурації і управління об'єктами Java за допомогою відображення. Контейнер відповідає за управління життєвим циклом об'єкта: створення об'єктів, виклик методів ініціалізації і конфігурація об'єктів шляхом зв'язування їх між собою.

Об'єкти, які створюються контейнером, також називаються керованими об'єктами (beans). Зазвичай конфігурація контейнера здійснюється шляхом завантаження XML-файлів, що містять визначення bean'ов і надають інформацію, необхідну для створення bean'ов.

Об'єкти можуть бути отримані одним з двох способів:

* 1. Пошук залежності - шаблон проектування, в якому викликає об'єкт запитує у об'єкта-контейнера екземпляр об'єкта з певним ім'ям або певного типу.
  2. Впровадження залежності - шаблон проектування, в якому контейнер передає екземпляри об'єктів по їх імені іншим об'єктам за допомогою конструктора, властивості або фабричного методу.

Spring має власну MVC-платформу веб-додатків, яка не була спочатку запланована. Розробники Spring вирішили написати її як реакцію на те, що вони сприйняли як невдалість конструкції (тоді) популярного Apache Struts, а також інших доступних веб-фреймворків. Зокрема, на їхню думку, не вистачило б поділ між шарами подання та обробки запитів, а також між шаром обробки запитів і моделлю.

Клас DispatcherServlet є основним контролером фреймворка і відповідає за делегування управління різних інтерфейсах, на всіх етапах виконання HTTP-запиту. Про ці інтерфейси слід сказати більш детально.

Як і Struts, Spring MVC є фреймворком, орієнтованим на запити. У ньому визначені стратегічні інтерфейси для всіх функцій сучасної запитання-орієнтованої системи. Мета кожного інтерфейсу - бути простим і зрозумілим, щоб користувачам було легко його заново імплементувати, якщо вони того побажають. MVC прокладає шлях до більш чистого front-end-коду. Всі інтерфейси тісно пов'язані з Servlet API. Цей зв'язок розглядається деякими як нездатність розробників Spring запропонувати для веб-додатків абстракцію більш високого рівня. Однак цей зв'язок залишає особливості Servlet API доступними для розробників, полегшуючи все ж роботу з ним.

### 2.4.5 Java

Мовою програмування у даній дипломній роботі було обрано Java, яка на мою думку найбільш підходить для написання таких програмних продуктів, як веб-застосунок.

Java - об'єктно-орієнтована мова програмування, розроблений компанією Sun Microsystems (в подальшому придбаної компанією Oracle). Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-який віртуальної Java-машині незалежно від комп'ютерної архітектури. Саме цей критерій і слугував орієнтиром при виборі – ми отримуємо платформену незалежність, а отже надаємо можливість розширити потенційну аудиторію користувачів.

## **2.5 Обґрунтування вибору програмних компонентів**

Перед обґрунтуванням обраних програмних продуктів, перш за все, маємо підтвердити актуальність обраного шляху для рішення питання адміністрування медичних карт.

Зазвичай, для подібних операцій, люди користуються електронною поштою. У такому разі будь-які веб-застосунки можуть здатися непотребом та марнотратством. Але є переваги, які змушують передивитись свою точку зору на це питання і підтвердити актуальність та необхідність цього дипломного проекту.

1. Центральне управління базою даних має на увазі, що адміністратор може «витягнути» будь-яку інформацію, незалежно від її місця створення.
2. Замість хаосу та невпорядкованості вхідної пошти, всі документи, які були отримані або відправлені, будуть впорядковано складатися у базу даних (або у комп’ютері користувача, в залежності від дії), у відповідні папки з датами, без стороннього втручання.
3. Завжди є людський фактор. Хтось стомився і вказав неправильну дату, ненавмисне видалив – інформація буде втрачена якщо не назавжди, то, у всілякому випадку спричинить багато проблем з пошуком та подальшим корегуванням оної. Наша структура не дозволить виникнення подібної ситуації, та мінімізує подальші ризики, пов’язані з втручанням необережного користувача.

Так буде виглядати «обличчя» для відправки документа:

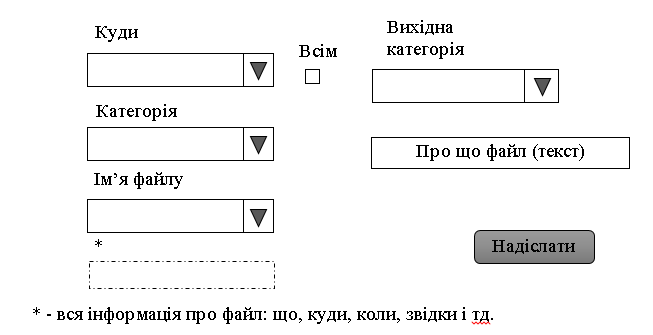


Рисунок 2.5.1 Відправка документа

Відправка документа включає:

1. Перевірка розміру документа.
2. Перевірка доступності сервера.
3. Перевірка правильності заповнення форми.
4. Передача файлу(-ів) з подальшим занесенням у базу даних.
5. При використанні електронної пошти, ми могли користуватися або POP3, або IMAP протоколами. POP3 одразу відкидаємо, бо він використовує offline парадигму, зберігає всі данні на сервері, та завантажує дані по запиту на єдину клієнтську машину (якщо на різні, інформація розсіюється). Нехай це и мінімізує з’єднання з сервером та витрачає мінімум серверних ресурсів, нас це не влаштовує. Все ще можемо користуватися протоколом IMAP. Безумовно він має багато переваг: доступ до серверу з поштою так, нібито вся кореспонденція розташована безпосередньо на комп’ютері користувача, електронними листами можна маніпулювати без постійної пересилки з сервера і назад, платформо-незалежний доступ до скриньок, дозволяє здійснювати пошук листів на сервері, можна скачати фрагмент, починаючи з будь-якого місця, будь-якої довжини. Але ми не можемо гарантувати завантаження інформації адміністратором у будь-який момент часу, а в нашій реалізації цієї проблеми уникаємо, бо адміністратор і сервер знаходяться у одному місці. Це надає нам мінімальні втрати. Також доступ тільки зареєстрованим користувачам, що у свою чергу забезпечує конфіденційність інформації. А це один із головних аспектів, тому що, якщо документооборот здійснюється у таких структурах як пенсійний фонд, мед. установа та інші, будь-який витік інформації або втрата документа може призвести навіть до кримінальної діяльності.
6. Використання протоколу HTTPS.

Далі буде приведена аргументація, чому обрані програмні продукти кращі та більш актуальні, для даного технічного рішення, та буде проведений порівняльний аналіз між конкуруючими фірмами.

### 2.5.1 CentOS

При виборі системи варто звернути увагу, який хостинг буде актуальнішим для нас. Як правило, вибір для розміщення веб-сайтів залежить від мови сценаріїв, які використовуються у нашому сайті. Якщо це PHP, MySQL, Perl, то linux хостинг є краще. Але якщо ми обираємо ASP.Net, то найкраще підійде windows хостинг.

Якщо потрібні інтерактивні об'єкти, такі, як в чаті або в пошуковій базі даних, Linux можливо не найкращий вибір. Крім того, Linux на основі веб-сервера в повному обсязі сумісний з технологіями Windows. Якщо ми використовуємо будь-які Windows-орієнтовані технології, вибір очевидний.

 Переваги сервера Linux хостингу

 1. Linux є операційною системою з відкритим вихідним кодом, і на відміну від інших операційних систем не потребує ліцензійних зборах. Ви можете його вільно завантажити і використовувати безкоштовно.

 2. Сайти на Linux можуть бути легко перетворені в веб-сайти Windows. Крім того, сайти можуть бути легко змінені в залежності потреб користувачів.

 3. Linux хостинг є дуже економічним, так як Linux є вільною операційною системою. Як правило, витрати на поширення несе власник сайту.

 4. Є багато видів баз даних, які працюють на Linux хостингу. Найбільш поширеними, швидше за все, є mSQL, MySQL і PostgreSQL. Ці бази даних є за своєю природою реляційними і дозволяють значно оптимізувати зв'язок з вашим сайтом для швидкого освоєння даних.

 5. У середовищі Linux хостингу існує набагато більше програмного забезпечення, ніж для Windows хостингу. З цієї причини саме Linux хостинг є популярним вибором для веб-дизайнерів і програмістів.

 Переваги Windows Server хостингу

 1. .NET технології (ASP.Net) доступні на цій платформі. Якщоє веб-сайт, який розроблений на Microsoft .NET технології, доведеться вибирати сервер Windows-хостингу.

 2. Якщо потрібна база даних корпоративного класу, є кілька різних функцій, які працюють на платформі Linux. Але якщо вам потрібно MSSQL бази даних Microsoft, Windows хостинг стане для вас найкращим вибором.

 3. Аccess бази даних можна отримати лише на сервері Windows. Аccess може працювати тільки на платформі Microsoft Windows, і тому недоступна для Linux.

Як правило, Linux сервер принаймні 20% економічніше, ніж Windows Server хостинговий план. Якщо не потрібні унікальні можливості для вашого сайту, придатні для пошуку бази даних і т.д., Linux сервер кращий для наших потреб.

Беручи до уваги вищеописане порівняння , кращим варіантом для нас буде Linux сервер.

FreeBSD - одна з найстаріших ОС з багатою історією. Але з кожним роком кількість користувачів цієї системою скорочується, хоча вона вважається однією з числа надійних і безпечних ОС. Є кілька причин падінь популярності даної ОС, головна з них - одна команда розробників і зовсім невелика кількість комерційного ПЗ для FreeBSD. Якщо виникне проблема з FreeBSD, вона з великою ймовірністю може залишитися невирішеною. На даний момент випущена 10 версія FreeBSD, яка успішно працює на безлічі процесорних архітектур і в руках майстра дає можливість для тонкої настройки ядра. У переваги FreeBSD потрібно внести відмінну реалізацію роботи з пам'яттю, мережею і системою введення-виведення.

Red Hat Enterprise Linux - одна з найпопулярніших і затребуваних ОС. В основному система призначена для корпоративного використання, і відрізняється особливою надійністю і безпекою. Доказ цьому те, що дану ОС використовують для розгортання важливих додатків на світових біржах, у фінансових установах, провідних телекомунікаційних компаніях, анімаційних студіях і тд. Основний недолік - ОС платна. Головна особливість - є комерційна підтримка дистрибутива. Періодичність виходу нових версій - 3 роки.

Windows Server 2008 R2 - система практична і має великий запас продуктивності. Серверна операційна система Microsoft, удосконалена версія Windows Server 2008, яка прийшла на зміну Windows Server 2003. Найкраще рішення для сервера файлів або терміналу, має інструмент для бекапів, що значно підвищує надійність. Серед головних плюсів та удосконалень можна визначити такі особливості: Server Core, Active Directory і служби терміналів.

Server Core - це істотно полегшена установка Windows Server 2008 в яку не включена оболонка Windows Explorer. Вся настройка і обслуговування виконується за допомогою інтерфейсу командного рядка Windows, або підключенням до сервера віддалено за допомогою консолі управління.

Доменні служби Active Directory (AD DS) зберігають дані каталогів і управляють взаємодією між користувачами і доменами, в тому числі входом в домен, перевіркою достовірності і пошуком в каталозі. За допомогою Active Directory замовники можуть управляти посвідченнями і взаємовідносинами, що формують мережу організації. Служби Active Directory інтегровані з Windows Server 2008 R2, можуть використовуватися відразу після розгортання і дозволяють організаціям централізовано налаштовувати параметри систем, користувачів і додатків і управляти цими параметрами.

Служби терміналів тепер підтримують Remote Desktop Protocol 6.0. Найпомітніше удосконалення, назване Terminal Services RemoteApp, дозволяє опублікувати одне конкретне додаток, замість всього робочого столу. Інша важлива особливість, додана в Служби терміналів - Terminal Services Gateway і Terminal Services Web Access (тепер повністю через web-інтерфейс). Terminal Services Gateway дозволяє авторизованим комп'ютерам безпечно підключатися до служб терміналів або віддалених робочих столів з інтернету використовуючи RDP через HTTPS без використання VPN. Для цього не потрібно відкривати додатковий порт на міжмережевому екрані; трафік RDP тунелюють через HTTPS. Terminal Services Web Access дозволяє адміністраторам забезпечувати доступ до служб терміналів через Web-інтерфейс. При використанні TS Gateway і TS RemoteApp, передача даних відбувається через HTTP (S) і віддалені програми виглядають для користувача так, як ніби вони запущені локально. Кілька додатків запускаються через один сеанс щоб гарантувати відсутність потреби в додаткових ліцензіях на користувача.

Не дивлячись на ці переваги, головним недоліком потрібно виділити вимогливість до апаратної частини. Windows Server 2008 R2 вже не підтримує 32-х розрядну архітектуру, та й в цілому вимагає для роботи істотно більше ресурсів, ніж його аналоги. І, одна з найважливіших особливостей, враховуючи, що проект розгортається без фінансової підтримки , те, що ОС вимагає покупку ліцензії. Також, варто не забувати про те, що саме на Windows орієнтоване величезна кількість існуючих вірусних програм. У більшості випадків цей продукт використовують для «домашнього» тестування сайту або при використання IIS і ASP.

На перших етапах можна одразу відкинути варіант FreeBSD, адже для вирішення проблем важлива не тільки команда розробників, але і частота використання ПЗ, чим більше людей використовує, тим більше питань вже пророблювались і можна швидко знайти потрібну інформацію. Windows Server у нашому варіанті не актуальний, по-перше, через вимогливість до апаратної частини, по-друге, будь-якому користувач Windows має доступ до диску C, і, по-третє, для користування потрібно придбати ліцензію. Для віддалених з’єднань це особливо затратно, бо не має цінової різниці між машинами. Наприклад, ПК користувача і принтер будуть коштувати однаково, близько 120$. Red Hat Enterprise Linux міг би бути ідеальним варіантом, але він платний. Тому у якості серверної бази даних було обрано операційну систему CentOS, яка є повністю безкоштовною паралельною гілкою RHEL (Red Hat Enterprise Linux). Число користувачів цієї ОС величезна, що дозволяє оперативно вирішувати всі виниклі проблеми і баги. Перевага - наявність безлічі форумів, де Вам зможуть допомогти з питаннями про CentOS. До значних плюсів Centos можна також віднести дуже зручний і спритний менеджер пакетів yum, а мінусом вважають наявність не найостанніших версій супутнього програмного забезпечення в стандартних сховищах (репозиторіях) пакетів для Centos.

### 2.5.2 MySql

СУБД – дуже важлива частина будь-якого проекту, де є робота з базою даних, тому до її вибору треба підходити дуже ретельно. Далі буде порівняно переваги і недоліки двох найпопулярніших СУБД - PostgreSQL і MySQL.

MySQL має наступні переваги перед PostgreSQL:

1. MySQL зазвичай набагато перевершує PostgreSQL за швидкістю роботи. Крім того, в MySQL 4.0 реалізований кеш запитів. Він дозволяє в багато разів збільшити швидкість обробки запитів для сайтів, на яких переважають неодноразово повторювані запити на читання.

2. За кількістю користувачів MySQL також набагато перевершує PostgreSQL. Тому код тестується значно прискіпливіше і досвідченим шляхом доведено велика його надійність, ніж у PostgreSQL. MySQL частіше, ніж PostgreSQL, використовується на виробництві, в основному тому, що компанія MySQL AB (раніше - TCX DataKonsult AB) надає високоякісну комерційну технічну підтримку MySQL з моменту появи цієї системи на ринку, а у PostgreSQL до самого останнього часу ніякої підтримки не було .

3. MySQL працює в середовищі Windows краще, ніж PostgreSQL. MySQL Server запускається як сьогодення (рідне) Windows-додаток (в NT / 2000 / XP - сервіс), в той час як PostgreSQL запускається в середовищі емуляції, Cygwin.

4. MySQL оснащений великою кількістю API для інших мов і підтримується великою кількістю існуючих програм, ніж PostgreSQL.

5. MySQL працює на високонадійних промислових системах 24/7 (включених 24 години на добу 7 днів на тиждень). У більшості випадків ніяких `` чисток '' в MySQL проводити не потрібно. PostgreSQL ж поки що не може працювати в таких системах, так як іноді доводиться запускати VACUUM для звільнення зайнятого наслідками роботи команд UPDATE і DELETE простору і проводити статистичний аналіз, необхідний для досягнення максимальної продуктивності PostgreSQL. Запускати VACUUM необхідно і після кожного додавання до таблиці декількох стовпців. На напружено працюючих системах VACUUM потрібно запускати частіше, в найгірших випадках - по кілька разів на день. Але ж під час роботи VACUUM (а її робота може тривати години, якщо база даних досить велика) база практично `` мертва ''. Втім, в PostgreSQL версії 7.2 виконання основних функцій цієї програми більше не призводить до блокування бази, і користувачі можуть продовжувати нормально працювати з нею. Нова команда VACUUM FULL береться за справу серйозніше: вона, як і в старих версіях, блокує таблицю і стискає копію таблиці на диску.

6. Реплікація MySQL відмінно протестована і використовується в таких сайтах, як: Yahoo Finance (<http://finance.yahoo.com/>), Mobile.de (<http://www.mobile.de/>), Slashdot (http://www.slashdot.org/).

7. У комплект поставки MySQL входять два тестових пакета, mysql-test-run і crash-me , а також пакет для замірів продуктивності. Тестова система постійно оновлюється, в неї додається код для тестування всіх нових можливостей і майже всіх відтворюваних помилок. Перед випуском кожної нової версії використовуються ці пакети для тестування MySQL на декількох платформах. Ці тести значно перевершують за своїми можливостями всі існуючі в PostgreSQL аналоги, і забезпечують високу якість коду MySQL.

8. MySQL підтримує більше стандартних функцій ODBC, ніж PostgreSQL.

9. MySQL має значно більш потужною реалізацією ALTER TABLE.

10. В MySQL передбачена можливість створення таблиць без транзакцій, що необхідно додатків, що вимагають максимально можливої ​​швидкості роботи. Ці таблиці можуть зберігатися в пам'яті, ставитися до типу HEAP-таблиць або дискових MyISAM.

11. MySQL може працювати з двома підтримують транзакції обработчиками таблиць, а саме - InnoDB і BerkeleyDB. Так як всі системи підтримки транзакцій в різних умовах працюють по-різному, це дає розробнику можливість знайти оптимальне рішення для умов, в яких буде працювати його система.

12. Можливість стиснення доступних тільки для читання таблиць, які не скасовує прямого доступу до їх записів, підвищує продуктивність системи, знижуючи кількість операцій зчитування з диска. Це особливо корисно при архівування.

13. В MySQL реалізований повнотекстовий пошук.

14. Система MySQL з самого початку розроблялася в розрахунку на многопоточность, а PostgreSQL використовує процеси. Перемикання контекстів і доступ до загальних даних декількома потоками здійснюється значно швидше, ніж окремими процесами. Таким чином MySQL Server в багатокористувацьких додатках отримує непогану перевагу в продуктивності, а крім того, таким чином MySQL Server вдається значно ефективніше користуватися перевагами, наданими симетричними мультіпроцесорними системами (SMP).

15. В MySQL реалізована значно потужніша система привілеїв, ніж в PostgreSQL. У той час як PostgreSQL забезпечує лише привілеї INSERT, SELECT і UPDATE / DELETE над базою або таблицею, MySQL надає можливість визначення повного набору різноманітних привілеїв на рівні бази, таблиці і стовпці. Крім того, MySQL дозволяє задавати привілеї для комбінацій хост / користувач.

16. В MySQL використовується протокол зв'язку між клієнтом і сервером із стисненням даних, що збільшує продуктивність системи в умовах низькошвидкісних каналів зв'язку.

17.Всі типи таблиць в MySQL (крім InnoDB) реалізовані у вигляді файлів (по одній таблиці у файлі), що значно полегшує створення резервних копій, перенесення, видалення і навіть створення символьних посилань між базами даних і таблицями, навіть якщо сервер знаходиться в неробочому стані.

18. Оновлення (апгрейд) MySQL проходить абсолютно `` безболісно ''. При модернізації MySQL немає потреби в копіюванні / відновленні даних, що доводиться робити при установці більшості оновлень PostgreSQL.

Недоліки MySQL в порівнянні з PostgreSQL:

1. Підтримка транзакцій в MySQL поки що не так добре перевірена, як в системі PostgreSQL.

2. Так як MySQL заснований на використанні потоків (threads), поки що ще не безпомилково працюють в деяких ОС, для забезпечення стабільної роботи доводиться або використовувати один з відкомпільованих пакетів, доступних за адресою http://www.mysql.com/downloads.

3. Блокування таблиць, що застосовується в нетранзакціонних таблицях MyISAM, у багатьох випадках працює швидше, ніж блокування на рівні сторінок, рядків або контроль версій. Недолік цього підходу в тому, що якщо не враховувати механізм роботи блокування таблиць, один тривалий запит може надовго заблокувати таблицю. Зазвичай цього ефекту можна уникнути, прийнявши відповідні заходи при розробці програми. Якщо це не вдасться, завжди можна змінити тип таблиці і зробити її транзакційною.

4. За допомогою UDF (user-defined functions, визначені користувачем функції) можливості MySQL можна розширити і доповнити звичайними SQL-функціями або їх об'єднаннями. Але це зробити не так просто, та й система не настільки гнучка в цьому відношенні, як PostgreSQL. See section 9.2 Додавання нових функцій в MySQL.

5. В MySQL складніше організовувалися поновлення, що зачіпають кілька таблиць відразу. Втім, це було виправлено в MySQL 4.0.2 реалізацією багатотабличного UPDATE і в MySQL 4.1 - за допомогою підзапитів. В MySQL 4.0 можна одночасно видаляти дані з декількох таблиць.

Але звичайно є певні переваги, за які можна обрати PostgreSQL:

1. У деяких випадках PostgreSQL виявляється ближче до ANSI SQL.

2. Роботу PostgreSQL можна прискорити, виконуючи код у вигляді збережених процедур.

3. При зберіганні географічних даних R-дерева дають PostgreSQL перевагу перед MySQL.

4. Оптимізатор PostgreSQL в деяких випадках здатний дати кращий в порівнянні з існуючим на сьогоднішній день оптимізатором MySQL результат. Особливо це помітно при злитті таблиць без відповідних ключів або при злитті з використанням різних ключів в поєднанні з логічним оператором OR.

5. Команда розробників PostgreSQL, які пишуть код для сервера, більше.

Отже, беручи до уваги вищевказані аргументи, перевагу можна віддати згідно з власною забаганкою, або згідно з поточними потребами програмного продукту. Обидві СУБД мають певний ряд недоліків і переваг, але у нашому випадку, була обрана саме MySQL, через відсутність таких недоліків опонента, як:

1. VACUUM ускладнює використання PostgreSQL в постійно діючих системах.

2. Наявність тільки транзакційних таблиць.

3. Значно більш повільна робота команд INSERT, DELETE і UPDATE.

### 2.5.3 Java

При написанні веб-застосунку, найважливіша частина і базис – мова програмування. Роздивимось переваги Java перед PHP у аспекті даної задачі.

Перевага Java - існування Google Web Toolkit, який дозволяє на одній мові, досить-таки «прозоро», писати як серверний шматок додатку, так і клієнтський. Переваги Java загалом не стосуються заліза. Це мова, на якій можна вирішувати в принципі будь-яке завдання, яке буде працювати майже де завгодно. Чисто «ідейний» аспект - на ньому складніше вчитися, але легше писати, він більш логічний і цим не дратує програміста, він розвинений і розширюється «до нескінченності». Це дійсно інструмент на всі випадки життя. Але у нього «важка» інфраструктура. Рефакторинг в Java набагато більш реальний, ніж в PHP через суворої типізації та чітких взаємозалежностей.

До того ж, модель доступу до БД в Java набагато більш універсальна, ніж в PHP, оскільки один і той же код на Java можна використовувати для різних БД (змінюючи тільки драйвер БД). На PHP код для доступу до іншої БД потрібно повністю переписувати, тому що функції доступу до різних БД на PHP абсолютно не сумісні одна з одною.

PHP простий у вивченні, «легкий» в навантаженнях, працює всередині HTML сторінок. Через що затребуваний і поширений. Але все добре тільки на початку. Потім відсутність строгих обмежень стає джерелом важковловимих помилок, а сама мова сприймається нелогічним і непередбачуваним.

Саме за вищеописані переваги та платформену незалежність, перевага надалась саме мові Java. А з огляду на це, веб-сервером було обрано Apache Tomcat. Це сучасний сервер додатків, що забезпечує роботу проектів на мові Java. Tomcat є окремим веб-сервером, який забезпечує роботу як динамічної частини сайту, так і статичних файлів (зображення і т.д.), дозволяє запускати веб-застосунки і містить ряд програм для самоконфігурування.

//не забыть рассказать про будущие улучшения впн и тд

//спринг и тд

Метою даного дипломного проекту є створення web-додатку для адміністрування бази медичних карт. А саме заходи забезпечення захищеного документообігу між клієнтом та сервером, під час взаємодії користувача або адміністратора з медичною карткою пацієнта , за допомогою веб-застосунку.

Головна ідея, на якій буде базуватися розробка даного веб-застосунку, створення єдиної бази даних, у якій будемо зберігати всі данні о пацієнтах. Суть ідеї полягає у тому, що доступ з будь-якого підрозділу здійснюється через веб-інтерфейс, для кожного окремого користувача, до цієї бази даних. З можливістю заносити та заванажувати звідти документи. Які звісно належать користувачу.