Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації
(повна назва)

Кафедра Радіотехнологій та інформаційно-комунікаційних систем
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ГЮІК.ХХХХХХ.008 ПЗ
Дослідження та застосування технології та конструкторів для розробки інтернет сайтів
(тема)

Виконала: студент 2 курсу, групи ІКТм-19-1

Безручко Роман Юрійович
(прізвище, ініціали)

спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
інформаційно-комунікаційні технології
(повна назва освітньої програми)

Керівник к.т.н. Дудка О.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Цопа О. І.
(підпис) (прізвище, ініціали)

2020 р

Не містить відомостей заборонених для відкритого публікування.

Керівник к.т.н. Дудка О.О.

Студент Безручко Р.Ю.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації

Кафедра радіотехнологій інформаційно-комунікаційних систем

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма інформаційно-комунікаційні технології
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студенту Безручко Романа Юрійовича
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та застосування технології та конструкторів для розробки інтернет сайтів
затверджена наказом по університету від 02 11 2020 р. № 1506Ст
2. Термін подання студентом роботи до екзаменаційної комісії 11 грудня 2020 р.
3. Вихідні дані до роботи Науково-технічні публікації та інтернет джерела з тематики атестаційної роботи
4. Перелік питань, що потрібно опрацювати в роботі вступ 1. аналіз предметної області і постановка завдання 2 огляд інструментів розробки 3 аналіз теми створення веб додатку 4 основні можливості системи 5 розробка бази даних 6 розробка веб-додатку 7 тестування веб-додатку

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п. 5 включається до завдання за рішенням випускової кафедри)

Слайди презентації

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	к.т.н. Дудка О.О.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз літератури та Інтернет-джерел	10.10.20	виконано
2	Постановка задачі	20.10.20	виконано
3	Дослідження сучасного стану вирішення	09.11.20	виконано
4	Дослідження моделей та методів побудови систем	20.11.20	виконано
5	Написання пояснювальної записки	04.12.20	виконано
6	Підготовка презентації	06.12.20	виконано
7	Перевірка на плагіат	08.12.20	виконано
8	Нормоконтроль	08.12.20	виконано
9	Захист	16.12.20	виконано

Дата видачі завдання 02 10 2020 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 58 с., 11 рис., 11 джерел.

Робота присвячена розробці веб-застосунку для діагностування захворювань шлунка. Такий застосунок призначений для людей, котрим необхідно поставити діагноз та яку діагностику треба зробити, аби усвідомитися у правильності діагнозу. За допомогою системи можна ввести турбуючі симптоми та отримати результат з найбільш ймовірним діагнозом. У ході виконання атестаціної роботи було проведено огляд сучасних тенденцій в області медицини, підібрано найпоширеніші хвороби.

На основі отриманої інформації розроблена концептуальна модель, яка була основою для проектування бази даних. Логічна модель даних реалізована в СУБД Oracle. Для розробки бекенд-частини використовувалася мова програмування Java, Spring Framework та Hibernate. Фронтенд розроблений за допомогою фреймворків AngularJs та Angular Material.

ORACLE, JAVA, TOMCAT, HIBERNATE, SPRING FRAMEWORK, ANGULARJS, ANGULAR MATERIAL, РОЗРОБНИК, КОРИСТУВАЧ, ДІАГНОСТУВАННЯ.

ABSTRACT

This bachelor thesis consists from 79 pages, 11 pictures, and 11 references.

The work is devoted to the development of a web application for the diagnosis of stomach diseases. This application is intended for people who need to diagnose and what diagnosis should be done to understand the correctness of the diagnosis. Using the system, you can enter anxiety symptoms and get the result with the most likely diagnosis. During the certification work was carried out an overview of modern trends in medicine, selected the most common diseases.

On the basis of the received information, a conceptual model was developed that was the basis for designing the database. A logical data model is implemented in the Oracle database. The Java programming language, Spring Framework and Hibernate were used to design the backend part. Frontend is developed using the AngularJs and Angular Material frames.

ORACLE, JAVA, TOMCAT, HIBERNATE, SPRING FRAMEWORK, ANGULARJ, ANGULAR MATERIAL, DEVELOPER, USER, DIAGNOSTIC.

ЗМІСТ

ВСТУП

Останнім часом все частіше можна почути про збільшення масштабів розробки програмного забезпечення орієнтованого на веб. Веб індустрія займає велику долю ринку порівняно з іншими напрямками. Це багатомільярдні компанії, що займаються розробкою розважальних продуктів та програмних забезпечень, що складаються з мільйонів рядків програмного коду і файлів мультимедіа.

Цьому сприяло масштабне зростання числа користувачів інтернету. Вебдодатки стали складніші і масштабніші. Сучасні веб-додатки - великі і складні програмні комплекси. Витрати тільки на програмну розробку часто вимірюються сотнями людино-місяців. За обсягом задіяних технологій, залучених коштів і рівня професійної підготовки розробників, веб індустрія давно зайняла аж ніяк не останнє місце в світі IT.

Спочатку інтернету вся інформація була тільки в текстовому вигляді. Разом з збільшенням швидкості інтернет з’єднання виростають і можливості користувачів. Зараз в найпопулярніших сайтах української частини мереж зареєстрована велика кількість людей це наприклад - є rozetka (10 млн користувачів) і olx (20 млн). Світовий лідер - Facebook (більше 500 млн користувачів по всьому світу). Facebook дуже активно розвивається на нашому ринку, і зараз в мережі зареєстровано 3,7 млн співвітчизників. Інший популярний в світі сервіс Twitter (мікроблоги) в Україні зібрав вже понад 600 тис. користувачів.

З збільшення кількості людей ростуть і їх потреби, такі гіганти веб напрямку, як Facebook, Instagram, Twitter вводять штучний інтелект для своїх стрічок новин. Починають використовуватись такі шаблони програмування, як мікросервіси, маркетплейс платформи застосовують в своїй роботі блокчейн. В такому різноманітності потреб та вимог стає дуже важливим створення об’ємного та обґрунтованого технічно порівняння мов та інструментів, які б максимально швидко вирішили поставлені перед ними завдання.

Тому існує досить велика кількість мов програмування орієнтованих на розробку web-додатків і, очевидно, що за процесом активізація їх використання ховається все більш складний етап порівняння та виявлення більш підходящого до вирішуваної задачі. Головна причина цього явища - відсутність єдиного механізму порівняння. Технічним керівникам команд доводиться витрачати значну кількість годин для виявлення, порівняння, та тестування, мов програмування які підходять для поставлених завдань. А згодом, в разі переходу до іншої задачі, доводиться, по суті, все повторювати заново, знову витрачати час та нерви. На великих підприємствах нерідко виникає необхідність впровадження і одночасного використання декількох мов програмування, а поверхневе порівняння яке може провести технічний лідер не дасть достатньо об'ємного результату. В цьому випадку необхідні витрати на висококваліфікованих програмістів можуть легко вийти за рамки допустимих, оскільки фахівці, які володіють одночасно декількома мовами програмування, зустрічаються досить рідко, а їх «ціна» росте аж ніяк не пропорційно кількості виконаних завдань. Перерахування подібних ситуацій можна продовжити, але вже зі сказаного, очевидно, що необхідно якимось чином вирішувати проблеми систематизації, профілювання та порівняння, щоб уніфікувати всі наявні нині мови розробки програмного забезпечення.Актуальність створення даного програмного продукту полягає в розвитку мережевих технологій, зокрема Інтернету. Більшості людей зручніше та швидше скористатися онлайн діагностуванням , ніж проводити часи в черзі до лікаря. Управління будь-якою установою ставить перед собою мету домогтися просування вперед, розвитку та прогресування в цілому. Одним з можливих шляхів досягнення цих цілей є автоматизація безпосередньо основних процесів, а також системи документообігу установи, тобто впровадження нових продуктів сучасних технологій.

Метою даної роботи є часткова автоматизація роботи поліклініки. Зберігання даних користувачів у базі дозволить істотно скоротити час для їх обслуговування, а наявність можливості поставити правильний діагноз онлайн відразу підвищить продуктивність роботи поліклініки.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАННЯ

1.1 Постановка задачі

Дуже часто виникають ситуації, коли великій кількості користувачів необхідно надати доступ до деяких даних, дати їм можливість їх отримання та редагування. Ці дані є загальними для всіх користувачів і не можуть розташовуватися на кожній клієнтській машині: правка інформації одним користувачем не буде помітна для інших. Для вирішення завдань такого роду застосовуються web-додатки (веб-додатки, web-based applications) – програмні засоби, призначені для автоматизованого виконання дій на web-серверах.

При цьому дані зберігаються на сервері, там же при необхідності виконується їх програмна обробка. За запитом необхідна інформація надається клієнту. Для забезпечення діалогу з користувачем в web-додатках використовуються web-браузери. З одного боку це створює ряд переваг (немає необхідності в розробці та встановленні додаткового клієнтського програмного забезпечення, користувач може запускати додаток в браузері незалежно від встановленої на його комп'ютері операційної системи). З іншого – накладає певні умови при виборі засобів розробки інтерфейсів таких додатків (HTML , CSS, JavaScript або Java).

Однак існують технології, що дозволяють спростити створення web-інтерфейсів (без використання безпосередньо зазначених раніше інструментів) і їх зв'язування з логічною частиною додатків.

Незалежно від засобу реалізації інтерфейсу користувача, web-додатки є підходящим рішенням для вищезгаданого роду завдань, тому що вони:

- забезпечують централізоване зберігання і обробку інформації;
- надають доступ до єдиної інформації одночасно великій кількості користувачів;
- не залежать від операційної системи, встановленої на комп'ютері користувача.

Завдяки цьому ряду переваг web-додатки набули значного поширення.

Згідно з теми дипломного проекту, потрібно проаналізувати технічні параметри орієнтованих на веб в залежності від задачі[1]. Потрібно проаналізувати технічні параметри роботи мов програмування такі, як – швидкодія, споживання оперативної пам’яті, навантаження на центральний процесор. Їх можна виміряти за допомогою математичних алгоритмів та за допомогою моделювання різних задач, які виникають при розробці під веб.[2]

Наприклад:

- розробка сайтів візиток;
- розробка більш складних сайтів – магазинів, новинних, корпоративных і т.д.;
- розробка сайтів для налізу великих об’ємів даних;
- розробка сайтів на основі блокчейн підходу;
- розробка сайтів для математичних операцій.

Також слід брати до уваги швидкість розробки програм, кільки нових версій, які вийшли за останній час, кількість відомих складнощів з безпекою[3].

1.2 Історія Інтернету

Інтернет – всесвітня система сполучених комп'ютерних мереж, що базуються на комплекті Інтернет-протоколів. Інтернет також називають мережею мереж. Інтернет складається з мільйонів локальних і глобальних приватних, публічних, академічних, ділових і урядових мереж, пов'язаних між собою з використанням різноманітних дротових, оптичних і бездротових технологій. Інтернет становить фізичну основу для розміщення величезної кількості інформаційних ресурсів і послуг, таких як взаємопов'язані гіпертекстові документи та електронна пошта.

Інтернет не має централізованого управління, правил використання чи доступу. Кожна складова мережа встановлює свої власні стандарти. Централізовано визначаються правила використання адресного простору Інтернет-протоколу та системи доменних імен. Мережа побудована на використанні протоколу IP і маршрутизації пакетів даних.[2]

Перший сайт з'явився в онлайні 6 серпня 1991 року. На цьому сайті була опублікована концепція технології WWW (World Wide Web), що використовує протокол передачі даних HTTP (HyperText Transfer Protocol) за допомогою системи адресації URI (Uniform Resource Identifier) за допомогою написання коду на мові гіпертекстової розмітки HTML (HyperText Markup Language).

Даний сайт мав ім'я info.cern.ch. Автором і власником сайту був Тім Бернерс-Лі. На даному сайті також була розміщена інформація по принципам установки, настройки і роботи серверів і браузерів. Крім того, сайт став першим в історії інтернет-каталогом, тому що автор розмістив на його сторінках перший список інших сайтів.

Природно, що матеріали та інструменти, необхідні для роботи першого сайту були підготовлені заздалегідь. До грудня 1990 року було розроблено гіпертекстовий браузер з функціями веб-редагування під назвою WorldWideWeb, сервер на базі NeXTcube і, власне, самі сторінки, які сервер видавав браузеру.[4]

Тім Бернерс-Лі був упевнений, що гіпертекст може служити основою для мереж обміну даними. Своє перше дітище - гіпертекстове програмне забезпечення Enquire він створив за 10 років до створення свого першого сайту, в 1980 році.

У травні 1991 року в Європейському Центрі Ядерних Досліджень в Женеві (CERN) (в якому працював і презентував свій перший сайт Бернерс-Лі) був затверджений стандарт WWW.

До 1993 року були остаточно сформовані специфікації URI, HTTP і HTML. 30 квітня 1993 року CERN оголосила, що World Wide Web буде вільною і безкоштовною для всіх. Це був дуже важливий крок, адже CERN мав повне право використовувати розробку в комерційних цілях. В такому випадку, сьогоднішнього Інтернет просто не існувало б.

13 березня 1989 року Тім Бернерс-Лі звернувся до свого керівника Майку Сендаллу з пропозицією створити систему управління інформацією. Даний лист фактично став стартовою точкою початку створення системи, що пізніше отримала назву World Wide Web - «Всесвітня павутина» або Інтернет.

Слід зазначити, що теоретичні передумови для створення гіпертексту були закладені ще за півстоліття до створення першого сайту. На початку 1940- х років Ванневар Буш висунув ідею про можливість розширення людської пам'яті за допомогою технічних пристроїв. А також (що більш важливо, з позиції технології веб) можливість індексації накопиченої людством інформації для швидкого пошуку. Далі Даг Енгельбарт і Теодор Нельсон запропонували ідею «ветвящегося текста» - власне, технологію гіпертексту. Технологія передбачала можливість надання читачеві різні варіанти читання. Свою систему для зберігання і пошуку тексту автори назвали Xanadu. Система так і залишилася незавершеною.

1.3 Класифікація веб сайтів

Сайт візитка - найпростіший вид сайту. Сайт такого типу можна зробити навіть на простому HTML, без використання системи управління сайтом. Зазвичай сайт-візитка містить від 1 до 5 сторінок. Сайти цього виду як правило включають в себе тільки загальну інформацію про власника сайту і його контактні дані. Простота розробки такого виду сайту робить вартість його створення порівняно дешевої, що є очевидною перевагою для замовника.

Корпоративні сайти - це повнофункціональні представництва компаній в інтернеті. Цей тип сайту найкраще підходить для серйозних середніх і великих фірм. Корпоративні сайти містять повну інформацію про компанію та її діяльність. Такий тип сайту іноді називають віртуальним офісом, так як відвідування такого сайту можна порівняти з спілкуванням з менеджером по роботі з клієнтами. Корпоративні сайти потрібні, в першу чергу, для формування іміджу компанії і надання відвідувачам і клієнтам якнайповнішої інформації.

Інтернет-вітрина або інтернет-каталог товарів - це вид сайтів, основне завдання яких - продавати. На таких сайтах розміщується інформація про товари і контакти, зазвичай телефони, за якими слід дзвонити бажаючим придбати пропонований товар. На таких сайтах розміщуються технічні характеристики товарів, відгуки, рекомендації експертів і т.д.

Інтернет-магазини аналогічний інтернет-вітрин, але має додатковий функціонал: можливість замовити пропонований товар прямо через сайт.

Промо-сайти призначені для розкрутки і просування будь-якого товару або бренду.

Тематичний тип інтернет сайтів характеризується тим, що містить інформацію з будь-якої конкретної тематики. Сюди ж можна віднести інтернетен-ециклопедії.

Портали - це тип сайтів, що містять велику кількість різноманітної інформації. Як правило, портали схожі за структурою з тематичними сайтами, але мають більш розвинений функціонал і більшу кількість сервісів і розділів. Також на порталах часто бувають розділи для спілкування користувачів: чати, блоги і форуми.

Блог - це тип сайтів, на яких власник або редактор блогу пише пости зі своїми новинами, ідеями або інший постійно надходить інформацією. Відмінною особливістю блогів є актуальність інформації, що публікується інформації.

Каталоги сайтів, основним вмістом яких є структуровані посилання на інші сайти, а також їх короткі описи.

Пошукові системи, призначених для пошуку сторінок в інтернеті по певних запитах.

Поштові сервіси надає інтерфейс для роботи з електронною поштою.

Інтернет-форуми:

На Інтернет-форумах цього виду користувачі можуть створювати теми, а також коментувати їх. Як правило, форуми обмежені однією специфічною тематикою, хоча зустрічаються і форуми «про все».

На сайтах-хостингах цього типу реалізована функція зберігання будь-яких файлів. Також часто зустрічаються сайти-хостинги з можливістю перегляду завантажених файлів прямо через браузер.

На сайтах дошки оголошень користувачі можуть розміщувати або шукати інформацію у вигляді будь-яких оголошень, наприклад - про купівлю-продаж.

Соціальні мережі створені для спілкування користувачів між собою. Як и правило, на таких сайтах є рейтинги, сторінки користувачів, групи і безліч інших сервісів.

1.4 Веб термінологія

В ході роботи використовуються слова, які можуть ускладнити розуміння матеріалу. Тому в даному розділі будуть описані основні терміни та визначення до них.

Адаптивний веб-дизайн - підхід до розробки сайтів, який має на увазі створення декількох окремих макетів сайту з фіксованими розмірами і набором особливих функціональних можливостей для різних категорій пристроїв і завантаження відповідного макета при кожному відвідуванні.

Браузер - програма для перегляду web-сторінок, один з основних інструментів користувача Інтернету. На сьогоднішній день найбільш поширений браузер Microsoft Internet Explorer (MSIE), що поставляється в складі операційної системи MS Windows. Досить часто використовуються браузери Opera, Chrome і Firefox. Різні браузери можуть по-різному відображувати одні і ті ж веб-сторінки (особливо створені непрофесіоналами) і по-різному виконувати сценарії (скрипти) на них. Професійні веб-розробники завжди враховують особливості браузерів при HTML верстці сторінок, завдяки чому такі сторінки однаково добре відображуються і працюють у всіх найбільш поширених браузерах.

Адреса сайту (URL, Uniform Resource Locator) - унікальна адреса вебсторінки або якогось іншого ресурсу в Інтернеті. Логін - це частина реквізитів доступу до закритих даних. Логін майже завжди супроводжується паролем. У кожного користувача може бути багато логінів для доступу до різних даних. Вони можуть використовуватися для доступу до системи управління сайтом, електронної пошти та інших ресурсів.[5]

Сервер / Server - Це потужний комп'ютер (цілодобово підключений до мережі Інтернет), на якому розміщується сайт (або сайти). Найчастіше клієнтові не потрібен свій сервер (це виключно дорого) і йому простіше і зручніше розміщувати свій сайт на спеціальному сервері хостинг-провайдера.

Блокчейн – вибудувана за певними правилами безперервна послідовна ланцюжок блоків (зв'язний список), що містять інформацію. Найчастіше копії

ланцюжків блоків зберігаються і незалежно один від одного (надзвичайно паралельно) обробляються на безлічі різних комп'ютерів.

Мікросервіси – сучасне уявлення сервіс-орієнтованої архітектури (SOA), що використовується для створення розподілених програмних систем. Як і в SOA, модулі в архітектурі мікросервісов взаємодіють по мережі один з одним для виконання мети. Ще одна схожість в тому, що мікросервіси використовують протокол-незалежну технологію. Дана архітектура є першою реалізацією SOA, що з'явилася після впровадження DevOps, і вона поступово стає стандартом для безперервно розвиваються систем.

В архітектурі мікросервісов модулі повинні бути невеликими і протоколи повинні бути легкими. Перевагою розподілу різних функцій системи в різні невеликі модулі є те, що це підсилює ступінь зачеплення і зменшує зв'язаність. Це дозволяє простіше додавати і змінювати функції в системі в будь-який час.

Властивості, характерні для архітектури мікросервісов:

- модулі можна легко замінити в будь-який час;
- модулі організовані навколо функцій, наприклад, призначений для користувача інтерфейс, логістика, виставлення рахунку і т. д;
- модулі можуть бути реалізовані з використанням різних мов програмування, баз даних, апаратних засобів і програмного забезпечення, в залежності від того, що підходить найкраще.

Термін «мікросервіси» існує вже більше десяти років. На конференції Cloud Computing Expo в 2005 Пітер Роджерс використовує його в своїй презентації. Тоді він використовував термін «мікро-веб-сервіс» для іменування компонентів програмного забезпечення. Юваль Леві мав схожу думку щодо мікросервісов, він так-же вважав, що саме мікросервіси є наступним етапом розвитку архітектури Microsoft. Він описав, як добре розроблена сервісна платформа застосовує основні архітектурні принципи веб і веб-служб разом з Unix-подібними, щоб забезпечити радикальну гнучкість і простоту, надаючи платформу для застосування сервіс-орієнтованої архітектури у всій середовищі додатків. У майстерні архітекторів програмного забезпечення, що проводиться недалеко від Венеції в 2011 році,

використовувався термін «мікросервіс» для опис загального архітектурного стилю, який розвивали учасники останні роки. І, нарешті, в 2012 році, все та ж команда, що і в 2011, вирішила прийняти цей термін, як остаточний.

Філософія мікросервісов фактично копіює філософію UNIX, а саме «Роби щось одне і роби це добре». Суть полягає в наступному:

- сервіси – невеликі, виконують кожен свою, єдину функцію;
- організаційна культура включає в себе автоматизацію розробки і тестування, це знижує навантаження на управління;
- принципи культури і дизайну повинні включати в себе «обхід»
- колишніх помилок;
- кожен сервіс – еластичний, легко змінюється, елементарний і при цьому закінчений.

1.5 Опис математичних алгоритмів

Для дослідження було обрано математичні алгоритми, завдяки яким можна було б виміряти таки характеристики:

- Стек виклику;
- Швидкість виконання;
- Кількість операцій в секунду.

Числа Фібоначчі – елементи числової послідовності в якій перші два числа рівні або 1 і 1, або 0 і 1, а кожне наступне число дорівнює сумі двох попередніх чисел. Названі на честь середньовічного математика Леонардо Пізанського (відомого як Фібоначчі).[2]

Послідовність Фібоначчі була добре відома в стародавній Індії, де вона застосовувалася в метричних науках (просодії, іншими словами - віршуванні) набагато раніше, ніж стала відома в Європі.

Зразок довжиною n може бути побудований шляхом додавання S до зразка довжиною n-1, або L до зразка довжиною n-2; і просодіцісти показали, що число зразків довжиною n є сумою двох попередніх чисел в послідовності. Дональд Кнут розглядає цей ефект в книзі «Мистецтво програмування».

На Заході ця послідовність була досліджена Леонардо Пізанським, відомим як Фібоначчі, в його праці «Liber Abaci» (1 202). Цей алгоритм має кілька привабливих особливостей:

- якщо час розрахунку n-го числа t, то (n + 1) -го - t * φ, де φ - це золотий перетин (√5 + 1) / 2;
- саме обчислюється n-е число округлене до ближайщего цілого величиною φn / √5;
- розрахунок fib (n + 1) вимагає n-й вложеності викликів.[6]

Перша особливість дозволяє за невеликий час протестувати транслятори, швидкості роботи яких розрізняються в сотні тисяч разів. Друга особливість дозволяє швидко перевіряти правильність розрахунків. Третя особливість теоретично дозволяє досліджувати ємність стека, але через те, що розрахунок при n> 50 стає дуже повільним навіть на суперкомп'ютері, практично використовувати цю особливість не представляється можливим.

Функція Аккермана – простий приклад всюди певної обчислюваної функції, яка не є примітивно рекурсивної. Вона приймає два невід'ємних цілих числа в якості параметрів і повертає натуральне число. Ця функція зростає дуже швидко.

Наприкінці 20-х років XX століття математики-учні Давида ГільбертаГабріель Судан і Вільгельм Аккерман вивчали основи обчислень. Судан і Аккерман відомі за відкриття всюди певної обчислюваної функції, котра є примітивно рекурсивної. Судан відкрив менш відому функцію Судану, після чого, незалежно від нього, в 1928 році Аккерман опублікував свою функцію.

Крім її історичної ролі як першої всюди безумовно не примітивно рекурсивної обчислюваної функції, оригінальна функція Аккермана розширювала основні арифметичні операції за спорудження до рівня, хоча і не так добре, як спеціально призначені для цього функції на кшталт послідовності гіпероператор Гудстейн.[2]

У статті «On the infinite» Гільберт висловив гіпотезу про те, що функція Аккермана не є примітивно рекурсивної, а Аккерман довів цю гіпотезу в статті «On hilbert's construction of the real numbers». Роза Петер і Рафаель Робінсон пізніше

представили двухаргументную версію функції Аккермана, яку тепер багато авторів вважають за краще оригінальної.

Ця функція з ростом n росте дуже швидко, але вважається дуже повільно. Остання властивість теоретично зручно для тестування швидкодії. Однак, розрахунок цієї функції вимагає значного числа рекурсивних викликів і більшість тестованих мов виявилося не в змозі їх підтримувати для обчислень, що мають помітну тривалість. Відомо, що обчислення цієї функції не можна звести до ітерації. Розрахунок по цьому завданню дозволив досліджувати максимальну ємність стека досліджуваних мов: розрахунок ack (1,1, n-1) вимагає n-й вкладеності викликів і дуже швидкий.

2 ОГЛЯД ІНСТРУМЕНТІВ РОЗРОБКИ

2.1 Загальні відомості про СУБД Oracle

Oracle Database - це об'єктно-реляційна система, котра підтримує деякі технології, які реалізують об'єктно-орієнтований підхід, тобто забезпечують управління створення та використання баз даних.[3]

Програмне забезпечення баз даних - це ключ до вирішення проблем управління інформацією. Взагалі кажучи, система управління базою даних (СКБД) повинна бути здатна надійно управляти великими обсягами даних в багатокористувацької середовищі, так, щоб всі користувачі могли одночасно звертатися до одних і тих самих даних. Все це має досягатися при забезпеченні високої продуктивності користувачів бази даних. СУБД також повинна бути захищена від несанкціонованого доступу, і повинна надавати ефективні рішення для відновлення від збоїв.

Сервер ORACLE забезпечує ефективні і дієві рішення для основних засобів баз даних:

- ORACLE підтримує найбільші бази даних, потенційного розміру до сотень гігабайт. Щоб забезпечити дієвий контроль за використанням дорогих дискових пристроїв, він надає повний контроль розподілу простору.

- ORACLE підтримує велику кількість користувачів, що одночасно виконують різноманітні додатки, які оперують одними і тими ж даними. Він мінімізує суперництво за дані і гарантує узгодженість даних.

- ORACLE підтримує всі описані вище засоби, зберігаючи високу ступінь сумарною продуктивності системи. Користувачі бази даних не страждають від низької продуктивності обробки.

- На деяких установках ORACLE працює 24 години на добу, не маючи періодів розвантаження, що обмежують пропускну здатність бази даних. Нормальні

системні операції, такі як відкат бази даних, а також часткові збої комп'ютерної системи, не переривають роботу з базою даних.[7]

- ORACLE може вибірково керувати доступністю даних, як на рівні бази даних, так і на більш низьких рівнях. Наприклад, адміністратор може відключити доступ до певної програми (з тим, щоб можна було здійснити перезавантаження даних цього додатка), не зачіпаючи інших додатків.[1]

- ORACLE задовольняє промислово прийнятим стандартам по мові доступу до даних, операційним системам, інтерфейсів з користувачем і мережевим протоколам. Це "відкрита" система, яка захищає інвестиції замовника. Сервер ORACLE7 був сертифікований Національним інститутом стандартів і технологій США як 100% -сумісний зі стандартом ANSI / ISO SQL89. ORACLE7 повністю задовольняє вимогам урядового стандарту США FIPS127-1 і має "маркировщик" для підкреслення нестандартних застосувань SQL. Крім того, ORACLE7 був оцінений Урядовим національним центром комп'ютерної безпеки (NCSC) як сумісний з критеріями захисту Помаранчевої книги; сервер ORACLE7 і Trusted ORACLE7 відповідають відповідно як рівнями C2 і B1 Помаранчевої книги, так і порівнянним з ними європейським критеріям захисту ITSEC.

- Для захисту від несанкціонованого доступу до бази даних ORACLE надає захищені від збоїв засоби безпеки, що лімітують і відстежують доступ до даних. Ці засоби дозволяють легко управляти навіть найбільш складними схемами доступу.

- ORACLE автоматично підтримує цілісність даних, дотримуючись "організаційні правила", які диктують стандарти прийнятності даних. Як наслідок, усуваються витрати на кодування і супровід перевірок в численних додатках бази даних.

- Щоб витягти максимум переваг з даної комп'ютерної системи або мережі, ORACLE дозволяє розділяти роботу між сервером бази даних і прикладними програмами клієнтів. Весь тягар управління спільно використовуваними даними може бути зосереджена в комп'ютері, що виконує СУБД, в той час як робочі станції, на яких працюють додатки, можуть сконцентруватися на інтерпретації і відображенні даних.

- В комп'ютерних середовищах, з'єднаних мережами, ORACLE комбінує дані, фізично знаходяться на різних комп'ютерах, в одну логічну базу даних, до якої мають доступ усі користувачі мережі. Розподілені системи мають таку ж ступенем прозорості для користувачів і узгодженості даних, що і нерозподілені системи, надаючи в той же час переваги управління локальною базою даних.

- Програмне забезпечення ORACLE переносимо між різними операційними системами і однаково в усіх системах. Додатки, що розробляються для ORACLE, можуть бути перенесені в будь-яку операційну систему з мінімумом модифікацій чи взагалі без таких.[10]

- Програмне забезпечення ORACLE сумісно з промисловими стандартами, включаючи більшість стандартних операційних систем. Додатки, що розробляються для ORACLE, можуть використовуватися в будь-якій операційній системі з мінімумом модифікацій чи взагалі без таких. Програмне забезпечення ORACLE дозволяє різним типам комп'ютерів і операційних систем спільно використовувати інформацію за допомогою мереж.[6]

2.2 Hibernate та його особливості

Hibernate - це ORM фреймворк для Java з відкритим вихідним кодом. Ця технологія є вкрай потужною і має високі показники продуктивності.

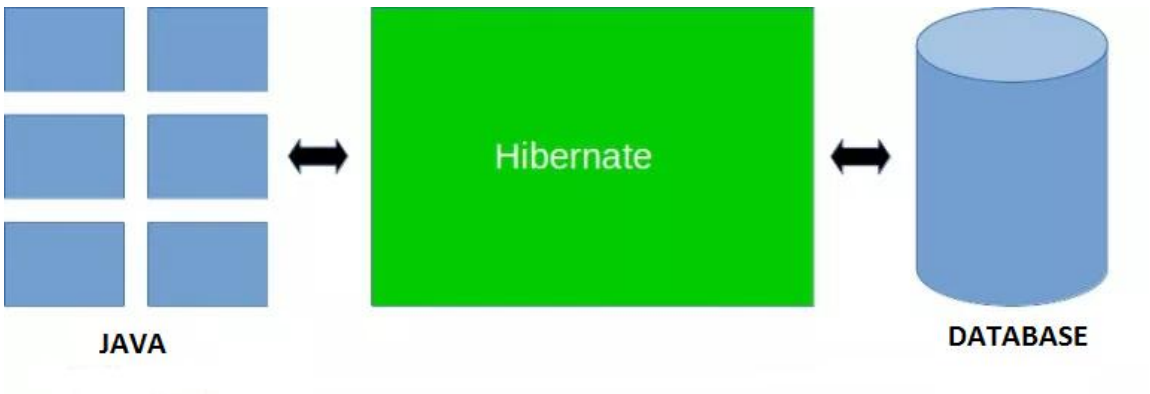

Рисунок 2.1 – Hibernate у загальному вигляді

Hibernate створює зв'язок між таблицями в базі даних та Java-класами і навпаки. Це позбавляє розробників від величезної кількості зайвої, рутинної роботи, в якій вкрай легко припуститися помилки і вкрай важко потім її знайти. В загальному вигляді схематично це можна зобразити, як показано на рисунку 2.1. Використання саме Hibernate надає наступні переваги.

- Забезпечує простий API для запису та отримання Java-об'єктів до/з БД.
- Мінімізує доступ до БД, використовуючи стратегії "fetching".
- Не вимагає сервера додатку.
- Дозволяє не працювати з типами даних мови SQL, а мати справу зі звичайними типами даних Java.[8]
- Піклується про створення зв'язків між Java-класами і таблицями БД за допомогою XML-файлів, не вносячи змін в програмний код. Якщо знадобиться змінити БД, то достатньо лише внести зміни в XML-файли конфігурації.

Додаток, що використовує Hibernate, має наступну архітектуру, що показана на рисунку 2.2.

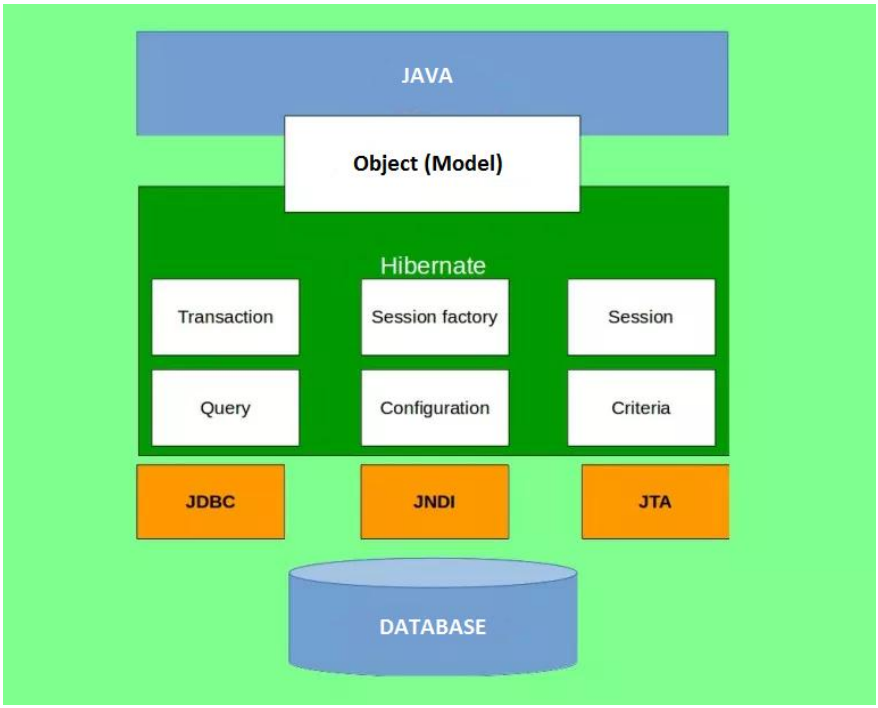

Рисунок 2.2 – Структура додатку з Hibernate

Hibernate підтримує такі API, як JDBC, JNDI, JTA. JDBC забезпечує найпростіший рівень абстракції функціональності для реляційних БД. JTA і JNDI, в свою чергу, дозволяють Hibernate використовувати сервери додатків J2EE.

Розглянемо детальніше кожен з елементів.[8]

Transaction – об'єкт, що представляє собою робочу одиницю роботи з БД. У Hibernate транзакції обробляються менеджером транзакцій. SessionFactory – найважливіший і найважчий об'єкт, що зазвичай створюється в єдиному екземплярі, у момент запуску додатку. Необхідна як мінімум одна SessionFactory для кожної БД, кожна з яких налаштовується окремим конфігураційним файлом. Session – це сесія, що використовується для отримання фізичного з'єднання з БД. Зазвичай, сесія створюється при необхідності, а після виконання якихось маніпуляцій закривається. Створення, читання, зміна і видалення об'єктів відбувається через об'єкт Session. Query – це об'єкт, котрий використовує HQL або SQL для читання, запису даних з або в базу. Об'єкт Configuration використовується для створення об'єкта SessionFactory та конфігурує сам Hibernate за допомогою XML-файла, який пояснює, як обробляти об'єкт Session. Criteria використовується для створення і виконання об'єктно-орієнтованих запитів.[4]

2.3 Основи Spring MVC

Метою SpringMVC Framework є підтримка в Spring архітектури модель-уявлення-контролер (model-view-controller). Spring забезпечує готові компоненти, які можуть бути використані (і використовуються) для розробки веб-додатків.

Головною метою MVC є поділ об'єктів, бізнес-логіки й зовнішнього вигляду програми. Всі ці компоненти слабо пов'язані між собою і при бажанні ми можемо змінити, наприклад, зовнішній вигляд програми, не вносячи суттєві зміни в інші два компонента.[9]

Отже, давайте пройдемося по кожному з цих блоків.

Модель (Model) Цей блок инкапсулирует дані програми. На практиці це POJO-класи (Plain Old Java Objects - Прості старі Java-об'єкти).

Подання (View) Модуль уявлення відповідає за виведення даних користувачеві. Зазвичай це JSP файл, який може бути пізнаний і інтерпретований браузером на користувальницької машині.

Контролер (Controller) Контролер відповідає за обробку запитів користувачів і передачу даних модулю View для обробки.

В основі Spring MVC Framework лежить DispatcherServlet, завдання якого - обробка всіх HTTP запитів і відповідей. У розумінні DispatcherServlet нам допоможе наступний рисунок 2.3.

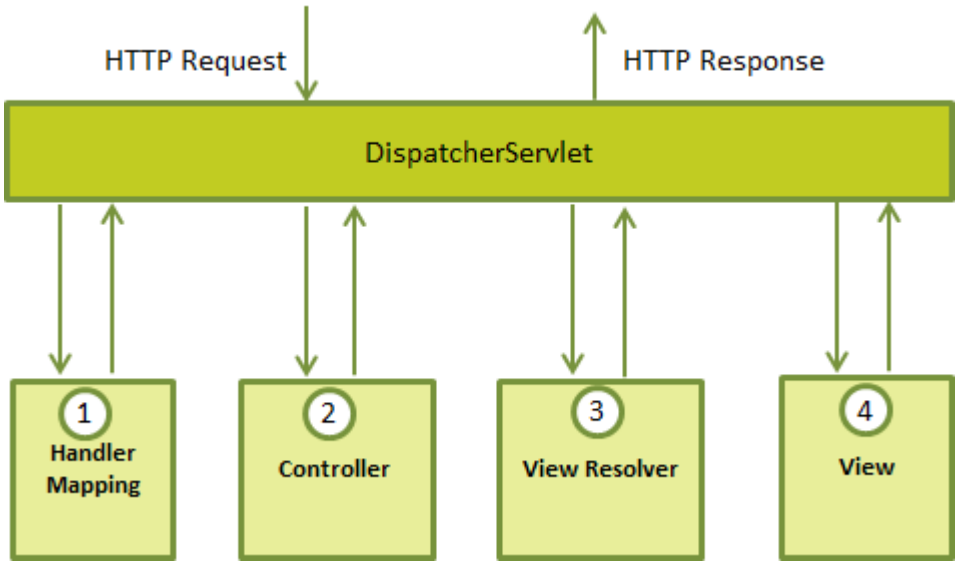

Рисунок 2.3 -Обробка HTTP запитів за допомогою DispatcherServlet

Після отримання HTTP-запиту DispatcherServlet (далі - DS) виконує наступні дії.

Після отримання HTTP-запиту DispatcherServlet дає вказівку об'єкту Handling Mapping (обробка зв'язування), який викликає наступний об'єкт.

DS надсилає запит контролеру і викликає відповідні методи, в основі яких лежать методи GET і POST. Ці методи повертають об'єкт, відповідно до бізнес логікою методу і передають назву (назву посилання) назад в DS.[9]

С допомогою View Resolver, DS підбирає необхідний вид для запиту.

І, коли зовнішній вигляд сформований, DS передає ці дані в модуль View, який обробляється браузером користувача.

2.4 Загальні відомості про AngularJs

AngularJS - JavaScript-фреймворк з відкритим вихідним кодом. Призначений для розробки односторінкових додатків [5]. Його мета - розширення браузерних додатків на основі MVC-шаблону, а також спрощення тестування і розробки.

Фреймворк працює з HTML, що містить додаткові атрибути, які описуються директивами, і пов'язує введення або виведення області сторінки з моделлю, яка представляє собою звичайні змінні JavaScript. Значення цих змінних задаються вручну або витягуються з статичних або динамічних JSON-даних.[11]

AngularJS спроектований з переконанням, що декларативне програмування найкраще підходить для побудови призначених для користувача інтерфейсів і опису програмних компонентів [8], в той час як імперативне програмування відмінно підходить для опису бізнес-логіки [9]. Фреймворк адаптує і розширює традиційний HTML, щоб забезпечити двосторонню прив'язку даних для динамічного контенту, що дозволяє автоматично синхронізувати модель і уявлення. В результаті AngularJS зменшує роль DOM-маніпуляцій і покращує тестованого.

Відділення DOM-маніпуляції від логіки додатки, що покращує тестованих коду.

Ставлення до тестування як до важливої частини розробки. Складність тестування безпосередньо залежить від структурованості коду. [10]

Поділ клієнтської і серверної сторони, що дозволяє вести розробку паралельно.

Проведення розробника через весь шлях створення програми: від проектування призначеного для користувача інтерфейсу, через написання бізнес-логіки, до тестування. [11]

Angular дотримується MVC-шаблону проектування і заохочує слабкий зв'язок між поданням, даними і логікою компонентів. Використовуючи впровадження залежності, Angular переносить на клієнтську сторону такі класичні серверні

служби, як відозавісімие контролери. Отже, зменшується навантаження на сервер і веб-додаток стає легше.

За допомогою директив AngularJS можна створювати призначені для користувача HTML-теги і атрибути, щоб додати поведінку деяких елементів. [12]

ng-app

Оголошує елемент кореневих для додатка. [13]

ng-bind

Автоматично замінює текст HTML-елемента на значення переданого вираження:

ng-model. Те ж, що і ng-bind, тільки забезпечує двостороннє зв'язування даних. [14] Чи зміниться вміст елемента - ангуляр змінить і значення моделі. Чи зміниться значення моделі - ангуляр змінить текст всередині елемента.

ng-class. Визначає класи для динамічного завантаження.

ng-controller. Визначає JavaScript-контролер для обчислення HTML-виразів відповідно до MVC; [11]

ng-repeat. Створює екземпляр DOM для кожного елемента з колекції; [11]

ng-show і ng-hide. Показує або приховує елемент, в залежності від значення логічного виразу;

ng-switch. Створює екземпляр шаблону з безлічі варіантів, в залежності від значення виразу;

ng-view. Базова директива, відповідає за обробку маршрутів [10], які приймають JSON перед відображенням шаблонів, керованих зазначеними контролерами;

ng-if. Видаляє або створює частина DOM-дерева в залежності від значення виразу. Якщо значення виразу, призначеного ngIf, так само false, об'єкт був видалений з DOM, інакше - знову клонований елемент вставляється в DOM. [9]

Також існує можливість створювати власні директиви, використовуючи в тому числі шаблони в тезі script.

2.5 Середовище розробки Intellij IDEA

У якості середовища розробки буде використовуватися Intellij IDEA від компанії JetBrains. При чому слід зазначити, що необхідно використовувати саме Ultimate версію продукту, оскільки саме тоді з’являється можливість легко підключити та налаштувати DataSource завдяки спеціальній вкладці "Database".

Сьогодні про IntelliJ IDEA знає, без перебільшення, весь світ. Хоча платформа орієнтована, в першу чергу, на Java-кодинг, її універсальність дозволяє працювати з різними мовами. Звичайно, тут немає такого масованого набору плагінів, як у будь-якого open-source конкурента, але зате в їх якості можна бути відносно переконаним. Якщо порівнювати цю середу з операційними системами, то IntelliJ буде те саме що Mac OS - закрита, злегка консервативна, але надійності не позичати. Та й немає великих складнощів в роботі. Як тільки ви захочете створити новий проект, інтуїтивне меню підкаже, як це зробити.[11]

Але лише сьогодні у IntelliJ IDEA красивий і інтуїтивний інтерфейс, який дозволяє розібратися навіть новачкові. Кілька років тому, коли команда тільки стартувала, вони розробляли свою середу розробки на основі тих же продуктів з відкритим кодом. Спочатку це навіть не повинно було бути повноцінною IDE. Замість цього компанія створювала інструменти, які повинні були взаємодіяти з середовищем, що володіє популярністю в ті роки - JBuilder. Але з урахуванням того, що остання була далека від досконалості, перша IntelliJ IDEA побачила світ вже через рік після початку роботи стартапа.

В першу чергу, команда сконцентрувалася на створенні засобів для рефакторинга і аналізу. Якраз з цього народилася сучасна IDEA. Розвиваючись в цьому напрямку, творці змогли розробити продукт, в якому код пишеться частково вами, а частково машинним інтелектом. А все починалося з того, щоб забезпечити зовсім утилітарні функції: комфортне перейменування класів, методів, автоматичне визначення і інші.

Загалом, IntelliJ IDEA надає інструменти для продуктивної роботи та ідеально підходить для створення комерційних, мобільних і веб-додатків.[11]

Продовжуючи працювати над середовищем і деякими іншими проекторами, вони усвідомили, що Java істотно обмежує їх можливості. Але це не стало приводом

для переходу на C або інший подібний мову високого рівня. Замість цього компанія розробила свій, під назвою Kotlin, який повинен замінити Java на всіх етапах. Цікаво, що Google включила цю мову в Android Studio як один з основних. AS - це офіційна IDE для створення продуктів під операційну систему Android.[9]

IntelliJ IDEA - це, перш за все, середовище розробки для Java. З цією мовою вона дружить найбільше, відмінно його розуміє і допомагає в написанні розробнику. Але це не означає, що все закінчується на Джаві і власною мовою Kotlin. Не менш важливим є підтримка їх коробки таких передових технологій, як Groovy, Scala і інших. Вони поєднують в собі можливості динозаврів, як Java разом з функціоналом Ruby, Smalltalk і інших. Не забувають в компанії і про тренди, адже для більшості веб-мов створені свої окремі середовища, засновані на IntelliJ IDEA. Серед них:

- PhpStorm - середовище розробки на PHP, яка активно використовується як для вивчення мови, так і для професійної детальності;
- RubyMine - середовище для взаємодії з мовою Ruby;
- PyCharm для Python;
- AppCode для Objective-C
- інші.

Все це створюється невеликими командами по всьому світу. І сьогодні IntelliJ IDEA - це найбільш інтелектуальне середовище. Вона по ходу написання і виконання може проаналізувати код, виявити помилки і запропонувати гідне рішення. Наприклад, IDEA може побудувати синтаксичне дерево, коли ви ще тільки набираєте код. Проаналізувавши посилання і ті шляхи, за якими програма може бути виконана, IDE пропонує варіанти того, як код може бути доповнений. Як тільки ви пройдете стадію адаптації до такого ритму роботи, дуже складно повернутися на безкоштовні програми.[9]

Звичайно, автодоповнення - не єдиний функціонал. Точно такі ж можливості ми можемо спостерігати практично у кожної безкоштовної програми. Причому, реалізовані вони на досить високому рівні. Але інтелект, як завжди, відрізняє IntelliJ IDEA. Наприклад, ви пишете код, який система опрацювала і готова дати варіанти для автодоповнення. Там, де звичайна програма запропонує кілька, IntelliJ IDEA

видає всього один, але вірний. Це економить лише секунду, зате в сукупності робить розробку набагато більш оптимізованої.

Однією з сильних сторін в JetBrains вважають підтримку широкого кола технологій. Наприклад, у всьому світі технології Flash вважаються застарілими. Але незважаючи на це, в деяких продуктах Flash реалізується досить широко, і скидати його з рахунків рано. Так вирішили і творці IntelliJ IDEA. Вони не виключають технологію, поки все навколо не перестануть нею користуватися. У підсумку, більшість Flash-розробників мігрує на IntelliJ IDEA. Компанії не так важливі світові тенденції, в той час як лояльність користувачів зростає.[11]

Не менш важливим інструментом є і графічний редактор для створення інтерфейсу. Це значно зручніше, ніж постійно писати шаблонний код. Розробник може лише перетягувати мишкою елементи на робоче поле, в той час як код генерується самостійно. Серед досвідчених програмістів такий підхід може вважатися дилетантським. Але правда в тому, що таким чином економиться багато часу, а якість коду не стає гірше. Через це візуальні редактори використовують не тільки новачки, але і ті, хто просто хоче оптимізувати свою роботу.

Повертаючись до Ultimate версії, зазначимо наступні переваги:

- розумне автодоповнення, інструменти для аналізу якості коду, зручна навігація, розширений рефакторинг і форматування для Java, Groovy, Scala, HTML, CSS, JavaScript, CoffeeScript, ActionScript, LESS, XML і багатьох інших мов;
- підтримка всіх популярних фреймворків і платформ, включаючи Java EE, Spring Framework, Grails, Play Framework, GWT, Struts, Node.js, AngularJS, Android, Flex, AIR Mobile та багатьох інших;
- інтеграція з серверами додатків, включаючи Tomcat, TomEE, GlassFish, JBoss, WebLogic, WebSphere, Geronimo, Resin, Jetty і Virgo;
- інструменти для запуску тестів та аналізу покриття коду, включаючи підтримку всіх популярних фреймворків для тестування.

2.6 Мова програмування Java

Java - це мова програмування загального призначення, який слід парадигмі об'єктно-орієнтованого програмування і підходу «Написати один раз і використовувати скрізь». Java використовується для настільних, мережевих, мобільних і корпоративних додатків.

Java - це не тільки мова програмування, а й екосистема інструментів, що охоплює майже всі, що може знадобитися при програмуванні на Java. У неї входять:

Java Development Kit (JDK) - комплект розробника Java. За допомогою JDK і стандартного блокнота можна писати і запускати / компілювати код на Java;

Java Runtime Environment (JRE) - виконуюча система Java. Механізм розповсюдження програмного забезпечення, складається з автономної віртуальної машини Java, стандартної бібліотеки Java (Java Class Library) і інструментів настройки.

Integrated Development Environment (IDE) - інтегроване середовище розробки. Інструменти, які допомагають запускати, редагувати і компілювати код. Найпопулярніші з них - IntelliJ IDEA, Eclipse і NetBeans.[11]

Java можна знайти всюди. Це основна мова розробки для Android. Він використовується в веб-додатках, урядових веб-сайтах і технологіях обробки великих даних, таких як Hadoop і Apache Storm. Java підходить і для наукових проектів, особливо в області обробки природної мови.

Як Java змінила світ програмування:

Гнучкість. Java довела, що C - процедурний, керований вручну і залежить від платформи код - це не межа досконалості. Завдяки Java, все більше людей почали застосовувати об'єктно-орієнтоване програмування, яке зараз використовується повсюдно.[8]

Аплети. Ще до появи JavaScript, в Java додали аплети - невеликі веб-програми, які надають інтерактивні елементи для візуалізації та навчання. Вони не використовуються ні для чого, крім простої анімації, однак аплети привернули увагу багатьох програмістів і підштовхнули їх до розробки HTML5, Flash і JavaScript.

Розробка через тестування. Java TDD - вже давно не експериментальна практика, а стандартний спосіб розробки програмного забезпечення. Введення JUnit 2000 року вважається одним з найбільших досягнень Java.

Java-технології мають багато особливостей, що відрізняють їх від інших технологій розробки програмного забезпечення.

- Переносімость. Програми, написані на мові Java, після одноразової трансляції в байт-код можуть бути виконані на будь-якій платформі, для якої реалізована віртуальна Java-машина. Найбільш ефективно можливості реального комп'ютера може використовувати тільки програма, написана з використанням «рідного» машинного коду.[11]
- Безопасность.Функціонірованіе програми повністю визначається (і обмежується) віртуальної Java-машиною. Відсутні покажчики та інші механізми для безпосередньої роботи з фізичної пам'яттю і іншим апаратним забезпеченням комп'ютера. Додаткові обмеження знижують можливість написання ефективних Java-програм.
- Надежность.В мовою Java відсутні механізми, потенційно призводять до помилок: арифметика покажчиків, неявне перетворення типів з втратою точності і т.п. Присутній суворий контроль типів, обов'язковий контроль виняткових ситуацій. Багато логічні помилки виявляються на етапі компіляції. Наявність додаткових перевірок знижує ефективність виконання Java-програм.[9]
- Сборщік мусора.Освобожденіе пам'яті при роботі програми здійснюється автоматично за допомогою «збирача сміття», тому програмувати з використанням динамічно розподіляє пам'яті простіше і надійніше. При інтенсивній роботі з динамічно розподіляє пам'яттю можливі помилки через те, що «збирач сміття» не встиг звільнити невикористовувані області пам'яті.
- Стандартние бібліотекі.Многіе завдання, що зустрічаються при розробці програмного забезпечення, вже вирішені в рамках стандартних бібліотек. Використання об'єктно-орієнтованого підходу дозволяє легко використовувати готові об'єкти в своїх програмах. Для запуску програми необхідна установка JRE, що містить повний набір бібліотек, навіть якщо всі вони не використовуються в

додатку. Відсутність бібліотеки необхідною версії може перешкодити запуску додатка.

- Самодокументіруемий код.Імеется механізм автоматичного генерування документації на основі коментарів, розміщених в тексті програм.
- Многообразіе типів пріложеній.На мовою Java можливо реалізувати абсолютно різні за способом функціонування і сфері використання програми.[8]

Переваги мови:

Легкість вивчення: Дана мова можна швидко вивчити і домогтися високої продуктивності програмування. PHP призначений для Web-розробників і HTML-кодувальників, дозволяє їм без проблем додавати до своїх Web-сайтах сучасні можливості, такі як динамічна генерація сторінок.

Відкриті джерела: PHP розповсюджується за ліцензією Apache, яка передбачає комерційне і некомерційне використання і розробку. Це означає, що програмою можна вільно користуватися без відрахування ліцензійних зборів. Крім того, існує всесвітня мережа талановитих розробників, які постійно покращують і розвивають PHP. Завдяки доступності вихідного програмного коду можна налагодити програму або налаштувати її під свої потреби.[11]

Підтримка баз даних: PHP забезпечує ефективну підтримку бах даних. Може працювати з ODBC, відкритими базами даних (MySQL і PostgreSQL), а також з комерційними (Microsoft SQL Server, Oracle і Sybase).

Розширення: Існує безліч вільно доступних розширень і вихідного коду для будь-яких прикладних задач: від маніпуляції з XML до доступу до каталогів. Програмісти можуть використовувати цю масу готового коду для швидкого компілювання найсучасніших додатків.

Недоліки мови:

Мова в повному обсязі об'єктно-орієнтований: Мова не має таких властивостей повністю об'єктно-орієнтованої мови, як індивідуальні змінні, множинне спадкування і.т.д.[10]

Відповідність корпоративним вимогам: Мова досить популярний в світі програм з відкритим кодом і технічно перевершує багато комерційних аналоги.

Однак йому не вистачає деяких важливих, з точки зору корпоративного середовища-особливостей. Це означає, що якщо ми захочемо використовувати PHP в корпорації, то це або взагалі не вдасться зробити, або буде потрібно значно більше додаткових програмних засобів, ніж при використанні Java або C ++.

3 АНАЛІЗ ТЕМИ СТВОРЕННЯ ВЕБ ДОДАТКУ

3.1 Огляд проблеми довіри лікарям

53% опитаних довіряють своєму сімейному лікареві, 21% натомість не довіряють, 26% не змогли оцінити свій рівень довіри до нього. Рівень довіри прямо залежить від частоти відвідувань лікарів респондентами та їхнього рівня обізнаності про нього – серед тих, хто знає свого сімейного лікаря, майже 80% йому довіряють. Жінки та особи старшого віку більше довіряють лікареві, аніж чоловіки та молодші.[1]

Більша половина (63%) заявляють, що сімейний лікар надавав їм поради щодо попередження захворювань або рекомендації щодо здорового способу життя рис3.1.

Рисунок 3.1Чим порадам довіряють українці

Водночас стільки ж запевнили, що не отримували подібних рекомендацій. Серед тих, хто отримував ці поради 82%, довіряють своєму сімейному лікареві.[2]

Вартість лікування в Україні опитані оцінили як дуже високу. 61% українців відповіли, що лікування обходиться для їхніх родин "дуже дорого", а для 28% – "дорого". Лише 8% українців лікуватися по кишені, а для 1% опитаних це навіть дешево рис 3.2.

Рисунок 3.2Наскільки дорого обхожиться лікування

Дві третини опитаних не приймають ліків або вітамінів для лікування або попередження серцевих захворювань, третина – приймають. 42% жінок вживають «серцеві» ліки, серед чоловіків таких лише 22%. Половина опитаних старше 51-го року також приймають подібні препарати. Серед тих, хто часто звертається до лікарів або цього року проходив обстеження серцево-судинної системи, таких майже дві третини.

39% опитаних довіряють новинам про зміни в медичній галузі, які поширюються на телебаченні, натомість 56% − не довіряють. До подібної інформації, яка поширюється в інтернеті з довірою ставиться 36% респондентів, натомість 39% не довіряють їй, крім того 26% не змогли оцінити свій рівень довіри

до медичних новин з всесвітньої павутини. До такої інформації в друкованій пресі з довірою ставляться 29% опитаних, з недовірою – 56%, медичним новинам на радіо довіряють 27% опитаних, не довіряють 52%. Серед мешканців Заходу найбільше тих, хто довіряє новинам про зміни в медичній галузі в усіх видах ЗМІ. Найменше таких серед мешканців Сходу. Молодь більше довіряє подібній інформації з мережі Інтернет, старше покоління – з телебачення. Жінки дещо більше довіряють таким новинам на телебаченні, аніж чоловіки.

Цікаво, що найбільше людей, для яких лікування обходиться дорого або дуже дорого, проживає на півдні країни – 92%. І саме там у пацієнтів найчастіше вимагали хабарі – у чверті випадків рис3.3.

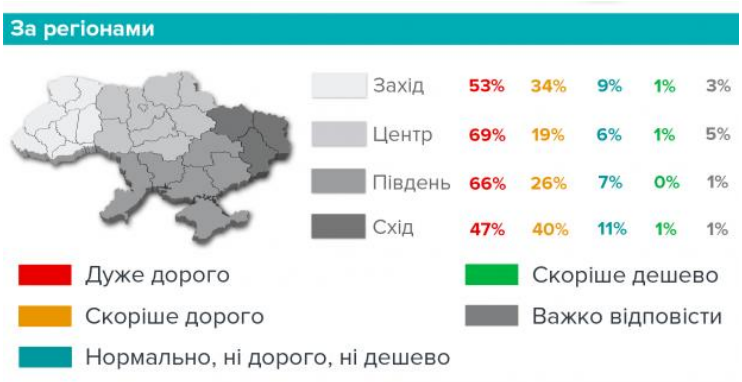

Рисунок 3.3Наскільки дорого обхожиться лікування за регіонами

37% опитаних консультуються із лікарем або будь-яким медичним спеціалістом із приводу стану свого здоров'я чи надання будь-яких рекомендацій щодо стану свого здоров'я рідше аніж раз на рік. Чверть користуються такою послугою один раз в рік, 17% − консультуються з лікарем принаймні кожні 6 місяців, 10% − кожні три місяця, 5% − щомісяця. Жінки частіше спілкуються з медиками стосовно свого здоров’я або ж рекомендацій, аніж чоловіки. Також особи старшого віку частіше відвідують лікарів аніж молодь.[1]

42% респондентів заявили, що мають від одного до трьох захворювань, які турбують їх у даний момент. 17% зазначили, що страждають від більше як чотирьох хвороб. Натомість третина заявила, що не мають жодних проблем із здоров’ям на

даний час. Стан здоров’я залежить від віку – чим старші респонденти, ти більше серед них тих, кого хвилюють захворювання. Чоловіки менше скаржаться на своє здоров’я аніж жінки.

27% опитаних перевіряли своє серце та судини цього року, ще 16% заявили, що проходили подібні обстеження у минулому році, 20% − більше аніж два роки тому, а 12% − більше аніж чотири роки тому. 18% взагалі не перевіряли стан серцево-судинної системи. Найбільше тих, хто перевірявся на предмет серцево-судинних захворювань цього року серед мешканців Півдня (31%) та Сходу (40%). Натомість чверть опитаних з Заходу взагалі ніколи не проходила подібних обстежень. Знову ж таки, серед осіб старшого віку більше тих, хто проходив подібні перевірки цього року, аніж серед молоді. Також подібних респондентів більше серед жінок, аніж серед чоловіків.

76% респондентів довіряють пожежно-рятувальній службі у їх регіоні, 14% − не довіряють. Службі швидкої медичної допомоги довіряють 74% опитаних, не довіряють – 22%. Медичним закладам (поліклінікам, лікарням) довіряють 57%, не довіряють – 36%, аптекам довіряють 55%, не довіряють – 38%, поліції довіряють 48% опитаних, не довіряють – 45%.[3]

Складається ситуація, коли люди з одного боку не задоволені станом медицини і вважають, що загальна якість державних медичних послуг в країні погіршується або не змінюється, з іншого – продовжують довіряти медичним закладам і лікарям.

Серед останніх урядових ініціатив щодо реформування медичної сфери респонденти найбільше підтримують відшкодування пацієнтам витрат на ліки від діабету, астми та серцево-судинних захворювань (84%) та запровадження обов’язково медичного страхування (64%). Створення Центрів громадського здоров’я підтримують 58%, не підтримують 21%, 22% не змогли визначитися; запровадження стандартизованих протоколів лікування підтримують 57%, не підтримують 12%, 30% не змогли визначитися; ухвалення концепції розвитку системи громадського здоров’я підтримують 57%, не підтримують – 22%, така ж кількість не змогла оцінити; запровадження механізму контрактів, що зумовить

підвищення зарплат лікаря, підтримують 44% опитаних, не підтримують 35%, 21% не відповіли на це питання, укладання контрактів із гарантованою ціною за лікування підтримують 38% опитаних, не підтримують − 42%, не змогли оцінити – 20%; заснування госпітальних округів підтримують 36%, водночас не підтримують 44%, 20% не змогли визначитися із своїми оцінками. [12]

З 64% у жовтні ц.р. до 69% у грудні зріс рівень підтримки ініціативи щодо створення фельдшерської служби з метою надання першої медичної допомоги при травматичних ситуаціях, щоб врятувати життя пацієнта, доки він не отримає повноцінного і незалежного лікування у відділенні швидкої допомоги. Водночас практично не змінився рівень підтримки скасування Наказу №33, згідно з яким бюджетні державні кошти виділялись медичним установам на підставі кількості лікарняних ліжок. Цю ініціативу як і восени підтримують дещо більше половини опитаних (58%).

Застосування процедури конкурентних, прозорих тендерів для державних закупівель певних лікарських препаратів підтримують 64%, не підтримують – 17%.

Лише 27% опитаних вважають, що запровадження механізму укладання контрактів між пацієнтом і лікарем зі збільшенням заробітної плати лікаря як мінімум на 30% покращить якість медичної допомоги. 36% стверджують, що це не матиме ніякого впливу, а 16% взагалі думають, що після цього якість допомоги погіршиться. 21% не змогли відповісти на це питання. Найбільше тих, хто оптимістично сприймає цю ініціативу на Заході (33%), найменше – на Сході (17%). Водночас, у східних регіонах найбільше тих, хто просто не зміг ніяким чином висловити свої оцінки стосовно цих змін (37%). Також найбільша підтримка цього механізму серед тих, хто доволі часто відвідує лікарів.

77% підтримують вакцинацію дітей проти таких хвороб як поліомієліт, кір, туберкульоз, гепатит В, дифтерія та ін. Не підтримують цей захід профілактики 12%, 11% не змогли формувати своє ставлення до цього. Рівень підтримки вакцинації дещо вищий серед міського населення, жінок, осіб старшого віку та тих, хто довіряє медичним закладам.

Запровадження обов’язкових щепленнь дітям проти таких хвороб як поліомієліт, туберкульоз, кір, гепатит В, дифтерія і т.д., за умови, що ці хвороби можуть загрожувати суспільному здоров'ю населення підтримують 71% опитаних, не підтримують – 13%, не визначилися із ставленням – 16%. Знову ж таки, цю ініціативу дещо більше підтримують міські мешканці, жінки, та ті, хто довіряють медичним закладам. [12]

Аудиторія: населення України від 18 років і старші. Вибірка репрезентативна за віком, статтю, регіонами і типом поселення. Вибіркова сукупність: 1500 респондентів. Особисте формалізоване інтерв’ю (face-to-face). Помилка репрезентативності дослідження: не більше 2,5%. Терміни проведення: 12-22 грудня 2016 р.

3.2 Методи діагностування

Медична діагностика — комплекс заходів та досліджень, спрямованих на встановлення діагнозу, тобто точної причини захворювання, а також змін внутрішнього середовища організму та супутніх захворювань, та призначення ефективного лікування захворювання. Медична діагностика поділяється на семіотику; методи обстеження хворих, які поділяються на лабораторні, інструментальні та фізикальні методи обстеження; а також методологічні основи встановлення діагнозу.

Процес встановлення діагнозу розпочинається вже від самого початку огляду хворого в лікувальному закладі або під час виклику лікаря за місцем проживання хворого. Розпочинається діагностика захворювання зі збору анамнезу захворювання. Після збору анамнезу лікар проводить огляд хворого, під час якого він здійснює перкусію та аускультацію пацієнта, пальпацію ділянки захворювання, вимірює у хворого артеріальний тиск, частоту серцевих скорочень та частоту дихання, і вимірює температуру тіла пацієнта. Дані анамнезу та огляду хворого заносяться до медичної документації (історії хвороби або амбулаторної картки пацієнта).[14]

Для уточнення діагнозу захворювання хворому також призначаються лабораторні, інструментальні та фізикальні методи обстеження.

Лабораторні методи обстеження включають загальноклінічні аналізи, до яких входять загальний аналіз крові, загальний аналіз сечі та аналіз калу. До лабораторних методів обстеження відносяться також біохімічні методи обстеження, під час яких визначається рівень глюкози, креатиніну, сечовини, білірубіну, ферментів печінки, ліпідів крові; коагулограма, при якій аналізуються показники зсідання крові; обстеження із визначення гормонів крові; визначення онкомаркерів; аналізи крові та інших біологічних матеріалів на інфекційні захворювання; алергологічні, токсикологічні, цитологічні та паразитологічні обстеження.

До інструментальних методів обстеження відносяться рентгенологічні, ендоскопічні, ультразвукові, методи реєстрації електричної активності органів (зокрема ЕКГ та ЕЕГ) та ряд інших методів обстеження.[13]

До рентгенологічних методів обстеження відноситься рентгенографія, рентгеноскопія, томографія, скринінговий метод для раннього виявлення захворювань дихальної системи — флюорографія, а також метод обстеження із створенням зображень органів із високою роздільною здатністю — комп'ютерна томографія. Близьким до цього методу, хоча й із використанням інших фізичних явищ, є магнітно-резонансна томографія та позитрон-емісійна томографія.[5] У діагностиці також можуть застосовуватися рентгенологічні методи обстеження із використанням рентгеноконтрастних препаратів.

Ендоскопічні методи обстеження, принципом яких є спостереження змін внутрішніх органів та порожнин людського організму з допомогою спеціального прилада — ендоскопа, застосовуються переважно для діагностики захворювань порожнистих і порожнинних органів. До них відносяться фіброгастродуоденоскопія, колоноскопія, ректороманоскопія, риноскопія, ларингоскопія, бронхоскопія, цистоскопія, кольпоскопія, лапароскопія, артроскопія та ряд інших обстежень. Ендоскопічні методи можуть поєднувати у собі як діагностичну мету, в тому числі взяття біопсії ураженого органу, при проведенні даних методів обстеження можуть також проводитись лікувальні маніпуляції. Окрім того, найчастіше для вивчення стану органів травної системи застосовується відеокапсульна ендоскопія, під час якої в травний тракт хворого вводиться відеокапсула, яка самостійно рухається

травним трактом і робить знімки стінок органів травної системи, що допомагає лікарю краще оцінити стан ураженого органу.[3]

Для обстеження щільних органів застосовуються ультразвукові методи обстеження. Ультразвукове обстеження застосовується для діагностики захворювань печінки, підшлункової залози, жовчного міхура, селезінки, нирок, сечового міхура, простати, жіночих статевих органів, молочних залоз, серця і судин, суглобів, застосовується також для діагностики патологічних станів у плода.

Для діагностики розладів частини систем та органів використовуються методи реєстрації електричної активності органів, до яких відносяться, зокрема, ЕКГ та ЕЕГ.

Ще можно сюди віднести діагностування за допомогою інтернет ресурсів. У далекому 1989 році, коли була винайдена Всесвітня павутина, ніхто не припускав, що Інтернет стане невід'ємною частиною нашого життя, а у деяких і її чільною складовою. У вільному доступі знаходиться інформація на будь-який смак – від опису життя простих мікроорганізмів до «очманілих ручок» по збиранню бомби. У цьому бездонному океані інформації, головне завдання сучасності – вміти нею правильно користуватися.[11]

В Інтернеті ми можемо знайти рішення будь-якої проблеми, ми замовляємо їжу з доставкою додому, купуємо речі сидячи на дивані, ми можемо працювати не вилазячи з ліжка або ніжачись на піску далеких островів – це неоціненні переваги вседоступності, але у всьому є і зворотна сторона медалі .

Наприклад – люди більше часу проводять зі своїм смартфоном, ніж з друзями за живим спілкуванням, менше рухаються, що провокує безліч захворювань і поява зайвої ваги. І найнебезпечніше – багато хто хоче вирішити виникаючі проблеми зі здоров'ям теж за допомогою Інтернету, але, не дивлячись на всі складні алгоритми роботи Мережі, замінити кваліфікованого лікаря ніщо не може, та й психосоматику теж ніхто не відміняв.[14]

Попит породжує пропозицію. І спочатку з'явилися просто розділи на сайтах, де можна було позначивши кілька симптомів прочитати свій «вердикт», а зараз вже існує безліч повноцінних спеціалізованих сайтів, які присвячені онлайн-діагностиці.

Також є форуми і розділи, де можна задавати питання, і на них відповідають фахівці, АЛЕ: це часто робиться просто щоб привернути увагу до свого ресурсу або збільшити потік пацієнтів до медичного закладу, такі відповіді засновані тільки на припущеннях доктора (якщо взагалі відповідає дійсно доктор) і зазвичай закінчуються запрошенням звернутися до того чи іншого фахівця або провести певну діагностику. Виняток становить тільки попередня віддалена консультація або висновок доктора, які ґрунтуються на наданих пацієнтом медичних документах і матеріали діагностичних досліджень.

Перспектива отримати діагноз і план лікування не виходячи з дому, не сидячи в чергах і не витрачаючи гроші на обстеження – досить приваблива, але чи можна вірити такій діагностиці?[14]

Таку діагностику можна порівняти з захопленням гороскопами, в теорії Ви можете наштовхнутися на правильну відповідь, рада і вирішення проблеми, але це вже йде гра з теорією ймовірності, і відсоток ймовірності цього украй невеликий.

Тільки факти:

Всесвітньою організацією охорони здоров’я описані десятки тисяч захворювань різного походження. З кожним роком їх список поповнюється, а деякі захворювання настільки складно диференціювати, що для правильної діагностики окремих випадків збираються цілі консиліуми кваліфікованих лікарів.

В Україні, щоб отримати медичну освіту потрібно, в залежності від спеціалізації, відучитися 7-9 років. І навіть коли лікар приступає до практики, він все одно продовжує постійно вчитися, іншого способу відстежити всі новинки і зміни в медичній сфері просто немає. І як після цього зазначивши де болить на 3D-фігурці можна отримати хоча б приблизно правильний висновок?

Не вся інформація медичної спрямованості в Інтернеті подається фахівцями, не кажучи просто про людей з медичною освітою

Існують різні завдання самих статей про здоров’я та медицині. Так звані «жовті» публікації спрямовані на те, щоб шокувати, а не надасть корисну інформацію, або змусити Вас клікнути і перейти на сайт. Звідси «Як вилікувати рак народними засобами», «Схудніть за тиждень на 40 кг без шкоди для здоров’я» і т. д.

Найнебезпечніше, що така самодиагностика і самолікування не просто не дасть ніяких результатів, а може погіршити наявну проблему.

Кілька прикладів симптомів і можливих діагнозів:

Одні і ті ж хворобливі відчуття в районі шлунка можуть сигналізувати про банальне нетравленні, про запалення апендикса або навіть інфаркту.

Болі у вусі можуть бути симптомами простудних проявів, патологію інфекційних захворювань і, навіть на початок запальних процесів, що відбуваються в мозку.

Першими ознаками раку передміхурової залози можуть виявитися біль в спині (особливо часто в попереку), ногах, грудях, при цьому інтенсивність болю у хворих різна.

Безсумнівно, користуватися можливостями сучасних технологій потрібно, але з користю для себе і свого здоров'я. Наприклад, багато клінік надають послугу онлайн-консультації з реальними фахівцями, яким можна розповісти про свою проблему, показати фотографії, відправити результати аналізів, обстежень або записатися на прийом. Це зручно і дійсно економить час, але лише в тому випадку, якщо це передує реальному візиту до лікаря.

Ще консультація онлайн – може бути першою допомогою в разі, якщо людина знаходиться на віддалі від найближчого медзакладу, але потребує термінового раді / підказкою. Онлайн-діагностика – це лише інформаційна допомога. Вона може задовольнити ваш інтерес і поповнити багаж знань, але ні в якому разі не є заміною повноцінного обстеження або керівництвом до дії. [14]

Сучасні технології швидко розвиваються, тому не можна сказати, що у майбутньому з'являться у легкому доступі такі онлайн додатки, які будуть допомогати хворому швидко, буде ставити правильний діагноз та насамперед парвильне лікування. Якщо люди вже знають про таку потребу, то в недалекому майбутньому такі ресурси будуть у вільному доступі.

У діагностиці різних захворювань також можуть використовуватися інші методи обстеження, зокрема введення в організм радіоактивних ізотопів та отриманні зображення шляхом визначення виділеного ними випромінювання

(сцинтиграфія); реєстрація виділеного тепла з організму людини (термографія); пункційна біопсія та ряд інших методів діагностики захворювань, які є специфічними для різних розділів медицини.

3.3 Хвороба та симптоми

Темою атестаційної роботи є створення веб-додатку, що дозволятиме користувачам вводити симптоми, що їх турбують, та миттєво отримувати результат діагностування та способи лікування. Додаток охоплюватиме захворювання шлунка, оскільки є тестовою версією. Але можливе подальше розширення за умови успіху та напливу користувачів.

Отже, предметною областю атестаційної роботи є захворювання та основні способи діагностування.[11]

Хвороба (лат. morbus — хвороба, захворювання) — захворювання окремої людини, поняття про Х. як нозологічну одиницю та узагальнене поняття про Х. як біологічне і соціальне явище. Найкоротше з відомих визначень Х. — «життя за ненормальних умов» належить Р. Вірхову. Ю. Конгейм (1878) визначав Х. як відхилення від нормального життєвого процесу, зумовлене взаємодією зовнішніх і внутрішніх умов і регуляторних процесів організму. У 1973 р. А.Д. Адо дав таке визначення: «Х. — це життя ушкодженого організму за участю процесів компенсації порушених функцій. Х. знижує працездатність людини». Певне захворювання, виділене на основі встановленої причини (етіологія), особливостей розвитку (патогенез), типових проявів, має назву нозологічна форма. Класифікація Х. постійно зазнає різних змін. На практиці застосовують такі класифікації: етіологічна, топографо-анатомічна, за віком і статтю, екологічна, за спільністю патогенезу. З 1970 р. використовується Міжнародна класифікація хвороб (МКХ), прийнята ВООЗ. Сьогодні діє МКХ-10 (10-го перегляду).

У розвитку Х. розрізняють чотири стадії або періоди: латентний період, період продромальних явищ, гострий період і період закінчення Х. Латентний, або прихований, період починається з моменту контакту організму з патогенними факторами і закінчується в момент появи перших клінічних ознак Х. В інфекційній

патології латентний період називають також інкубаційним, маючи на увазі період часу від зараження до виникнення продромальних явищ або гострого початку Х. Продромальним періодом вважають відрізок часу від появи перших клінічних проявів Х. до повного розвитку симптомів Х. Гострий період, який настає після продромального, характеризується повним розвитком властивої для даної Х. клінічної картини. Х. може закінчитися видужанням, рецидивом, тобто поверненням Х. у новий цикл, переходом гострої форми в хронічну, смертю.

Поступове клінічне видужання може бути повним або неповним; межі цих понять нечіткі, видужання може бути практично повним, хоча в організмі залишається стійкий анатомічний дефект, напр. відсутність однієї нирки, якщо функція її повністю компенсується функцією другої нирки. Але й повне видужання не є повним поверненням організму до вихідного стану. Напр., після перенесення захворювання на віспу або кір організм набуває нової властивості — імунітету до цих інфекцій.

Виділення періодів Х., варіантів перебігу, фаз загострення й ремісії, вивчення причин переходу гострої форми в хронічну мають велике клінічне значення, оскільки терапевтична тактика в різні періоди і фази Х. різна. Вивчення закономірностей виникнення рецидивів і ускладнень також необхідне для планування лікувальних і профілактичних заходів. Важливими завданнями системи охорони здоров’я є профілактика Х., їх вчасна діагностика та адекватне лікування, яке залежно від характеру захворювання може бути як з використанням ЛП, так і хірургічним, фізіотерапевтичним, променевим, за допомогою лікувального харчування та ін. Серед завдань спеціалістів фармації в рамках фармацевтичної опіки великого значення набуває виявлення симптомів небезпечних Х., при яких недопустиме самолікування та є обов’язковим надання медичної допомоги. Прогрес науки зумовлює постійний перегляд сформованих уявлень та оцінок поняття «Х.», а також принципів класифікації й періодизації Х.[13]

Номенклатура хвороб - перелік прийнятих, встановлених медичною наукою назв хвороб, об'єднаних в групи і класи.

У медичній практиці застосовується більше п'яти тисяч позначень діагнозів. При цьому одна і та ж хвороба часто іменується по-різному. Для вивчення захворюваності населення необхідна раціонально побудована номенклатура хвороб, завдання якої полягає в тому, щоб величезне різноманіття діагнозів об'єднати в порівняно невелика кількість назв класифікаційних.

В межах однієї країни без єдиної номенклатури хвороб неможливо порівняння захворюваності і смертності в окремих областях, містах, районах. Для порівняння захворюваності та смертності населення різних країн, необхідна єдина міжнародна номенклатура хвороб. Встановлення правильних назв діагнозів необхідно при реєстрації хвороб, яка є початковою ланкою дослідження захворюваності та смертності населення.

У зв'язку з розвитком медичної науки і завданнями вивчення захворюваності та смертності населення номенклатура хвороб періодично переглядається та вдосконалюється.

Нова номенклатура хвороб має триступінчату будову. Першу, найбільш широку ступінь становить номенклатура хвороб. Кожна з них має свій номер і становить окрему рубрику номенклатури.[11]

Другу сходинку становлять групи, кожна з яких формується шляхом об'єднання того чи іншого числа рубрик. Нарешті, третю, найбільш високу і узагальнюючу щабель, сформовану шляхом об'єднання того чи іншого числа груп, складають класи. Всього номенклатура включає 17 класів і 999 рубрик.

Номери і назви класів:

I Інфекційні та паразитарні хвороби;

II Новоутворення (пухлини) ;

III Хвороби ендокринної системи, розлади харчування та порушення обміну речовин;

IV Хвороби крові і кровотворних органів;

V Психічні розлади;

VI Хвороби нервової системи та органів чуття;

VII Хвороби системи кровообігу;

VIII Хвороби органів дихання;

IX Хвороби органів травлення;

X Хвороби сечостатевих органів;

XI Ускладнення вагітності, пологів та післяпологового періоду;

XII Хвороби шкіри та підшкірної клітковини;

XIII Хвороби кістково-м'язової системи та сполучної тканини;

XIV Вроджені аномалії;

XV Деякі причини перинатальної захворюваності та смертності;

XVI Симптоми і неточно визначені стани;

XVII Нещасні випадки, отруєння і травми;

XVII клас особливо багатий групами і рубриками (39 груп і 369 рубрик, тобто найменувань окремих видів травм та отруєнь). Така обширність цього класу пояснюється головним чином тим, що в ньому є дві незалежні один від одного класифікації: [11]

- за причинами травми (нещасні випадки на залізниці, на водному, повітряному транспорті тощо);
- за видами травм (переломи черепа, переломи кінцівок, вивихи без перелому кісток, удари, опіки тощо).

Застосування єдиної номенклатури хвороб у практиці вивчення причин захворюваності або смертності населення полягає у наступному.

В процесі обстеження хворих встановлюються і записуються в медичних документах (амбулаторних картах, історії хвороби) їх діагнози захворювань.

В цілях вивчення захворюваності велика різноманітність застосовуваних діагностичних позначень зводять до порівняно небагатьом рубриками єдиної номенклатури. Для цієї мети застосовують шифрування діагностичних позначень - поруч з діагнозом медпрацівник ставить номер, під яким це захворювання позначено у міжнародній номенклатурі хвороб.

Після шифрування документи розкладають за цими номерами з подальшим підрахунком числа захворювань по кожному шифру.

Для успішного вивчення захворюваності і полегшення роботи по шифровці діагнозів велике значення має точне і докладний позначення діагнозу хвороби в медичних документах. Замість діагнозу не можна ставити симптом (головний біль, задишка, кровотеча тощо). Для статистики захворюваності необхідні уточнені заключні діагнози, що встановлюються після спостереження за хворим і проведення спеціальних досліджень (лабораторних, рентгенологічних та ін).

Для вивчення захворюваності і правильної шифровки діагнозів по багатьом формам хвороби необхідно в діагнозах писати певні уточнення, від яких залежить їх правильна шифрування, наприклад при таких хворобах, як ларингіт, трахеїт, фарингіт, синусит, тонзиліт, бронхіт, гастрит, ентерит, коліт, нефрит, апендицит та ін, слід зазначати їх характер - гострий або хронічний. При встановленні діагнозу новоутворення вказують його локалізацію і характер (злоякісне або доброякісний); при діагноз грижі відзначають - неущемленная або защемлена і т. д.

При діагнозах судинних уражень центральної нервової системи вказують, чи пов'язані вони з гіпертонічною хворобою, атеросклерозом або іншими захворюваннями. При встановленні діагнозу гіпертонічної хвороби відзначають її стадію і який орган переважно вражений (серце, мозок, нирки). При діагноз ревматизму вказують - гостра або хронічна форма протікає захворювання з ураженням серця або без нього, з ураженням будь клапанів серця.

При нещасних випадках і травмах відзначають локалізацію травми, її характер (перелом, вивих, забиття, опіки та ін); чітко характеризують травму ока, зокрема наявність впровадження чужорідного тіла і опіків.[13]

Симптом (від грец. Symptomos - ознака, збіг) - це ознака будь-якого захворювання - статистично значуще відхилення того чи іншого показника від кордонів його нормальних величин або виникнення якісно нового, не властивого здоровому організму явища.

За інформативною значимості симптоми класифікують на наступні види.

Діагностичні (патогномонічні) симптоми - властиві тільки одному захворюванню.

Специфічні симптоми - характерні для групи захворювань органів однієї системи.[11]

Неспецифічні симптоми - характерні для багатьох захворювань.

Нехарактерні для даного захворювання симптоми.

Залежно від виду методу дослідження, за допомогою яких симптоми виявлені, останні діляться на наступні види.

- Суб'єктивні симптоми - виявляються методом розпитування хворих, засновані на описі хворим своїх відчуттів, що виникають в ході розвитку хвороби.

- Об'єктивні симптоми - виявляються об'єктивними методами дослідження (оглядом, пальпацією, перкусією, аускультацией, лабораторними та інструментальними дослідженнями).

За часом появи симптоми поділяють на:

ранні (початкові) - виникають на самих ранніх стадіях розвитку хвороби (н.п. болю в горлі та гарячка при ангіні, біль в серці при інфаркті міокарда, кашель і гарячка - при пневмонії та ін.);

пізні - виникають в період розпалу захворювання або в період його дозволу (н.п. нальоти на мигдалинах при ангіні, шум тертя перикарда при інфаркті міокарда, бронхіальна подих при пневмонії та ін.).[11]

За прогностичної значимості симптоми поділяють на:

- сприятливі, що вказують на легке або звичний плин захворювання, на його дозвіл;

- несприятливі (загрозливі), що свідчать про важку форму хвороби, про можливість її несприятливого результату.

Синдром (від грец. Syn - разом, dromos - переміщатися, бігти) - це поєднання симптомів, їх група, об'єднана спільним патогенезом. Термін «синдром» - введено в практику медицини Клавдієм Галеном.

Симптомокомплекс - це група симптомів або синдромів, характерних для захворювання, але не об'діненія загальним походженням.

3.4 Постановка задачі

Метою виконання атестаційної роботи є:

- Створення бази даних «Діагностування захворювань», в якій буде зберігатися і оброблятися інформація про користувачів (пацієнтів), їхні симптоми, а також отримані в результаті діагностування діагнози, що суттєво поліпшить якість обслуговування пацієнтів.
- Створення Java-додатку, налаштування підключення до бази даних за допомогою Hibernate. Розробка сприятливої архітектури, використання контролерів, сервісів, конвертерів для виконання своїх задач.
- Розробка "user-friendly" дизайну додатку та налаштування зв’язку між фронтенд та бекенд частиною за допомогою відправлення різного типу запитів.

Таким чином, кінцевим результатом виконання атестаційної роботи буде повноцінний веб-застосунок, що надаватиме можливість користувачам дізнаватися діагноз та способи лікування, не зходячи з місця.

4 ОСНОВНІ МОЖЛИВОСТІ СИСТЕМИ

4.1 Можливості користувачів

Функціональність системи повинна бути різною для різних категорій, зокрема для звичайного користувача та розробника (адміністратора). Розглянемо основні можливості користувача додатку.

Потрапивши на сторінку розроблюваного веб-додатку, користувач має змогу ввести усі необхідні для діагностування параметри – особисті дані та симптоми, що його турбують, що показано на рисунку 4.1.

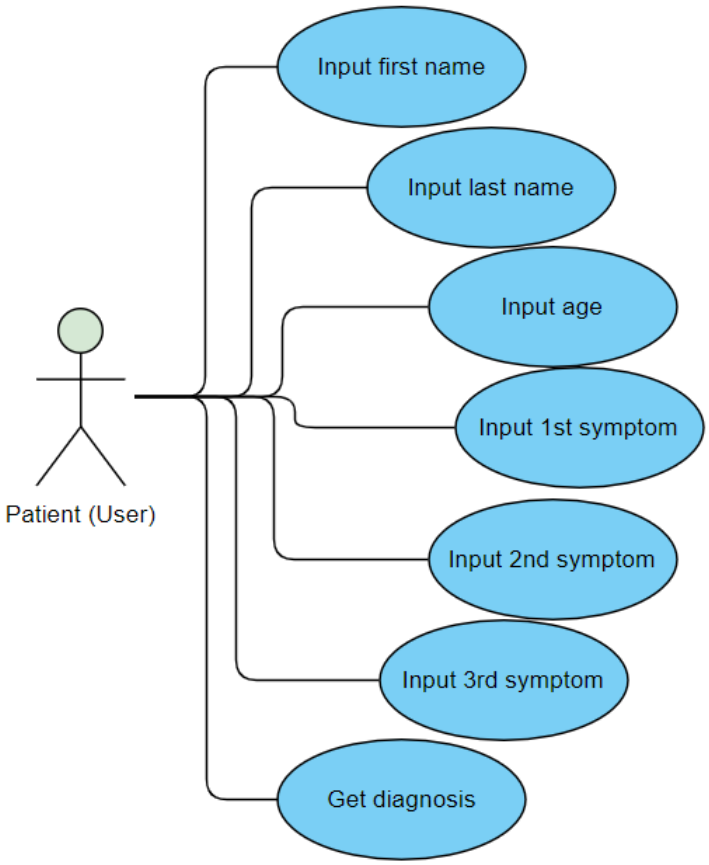

Рисунок 4.1 – Варіанти використання відвідувача

Тож простий відвідувач може цілком скористатися найголовнішим функціоналом додатку – діагностуванням хвороби, але він не має право на перегляд будь-якої іншої додаткової інформації.

Для цього користувачеві необхідно просто перейти на сторінку додатку та ввести усю інформацію для проведення діагностування. Як це виглядатиме з «середини» самого додатка розглянемо на наступній діаграмі послідовності, що зображена на рисунку 4.2.

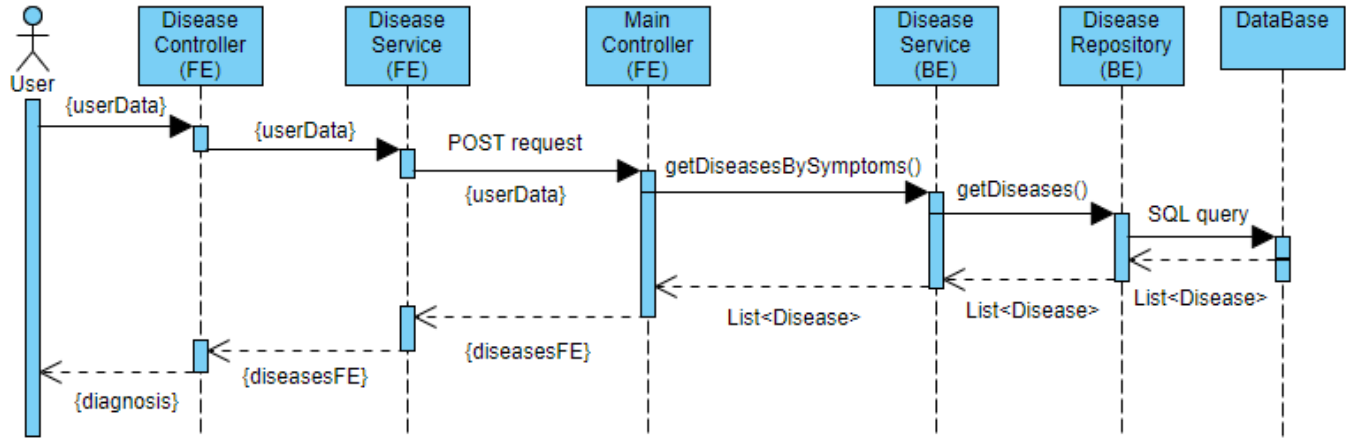

Рисунок 4.2 – Діаграма послідовності дій

Більш детально про роботу додатку щодо діагностування хвороби розглянемо у розділі опису розробки бекенд частини.

4.2 Можливості адміністратора

Як у більшості систем, адміністратор має право майже на усе. Щодо розроблюваного додатка, окрім того, що адміністратор може також відвідати веб-сайт та діагностувати хворобу за симптомами, він має право скористатися доступом до бази даних та за допомогою певних запитів отримати усю необхідну інформацію про пацієнта, захворювання та симптом.

Також, адміністратор за допомогою запитів до бази даних може подивитися список хвороб, які найбільш часто зустрічаються у результаті діагностування, або ж

виконати обернену операцію – отримати симптоми введеної хвороби. Усі можливі дії адміністратора показані на рисунку 4.3.

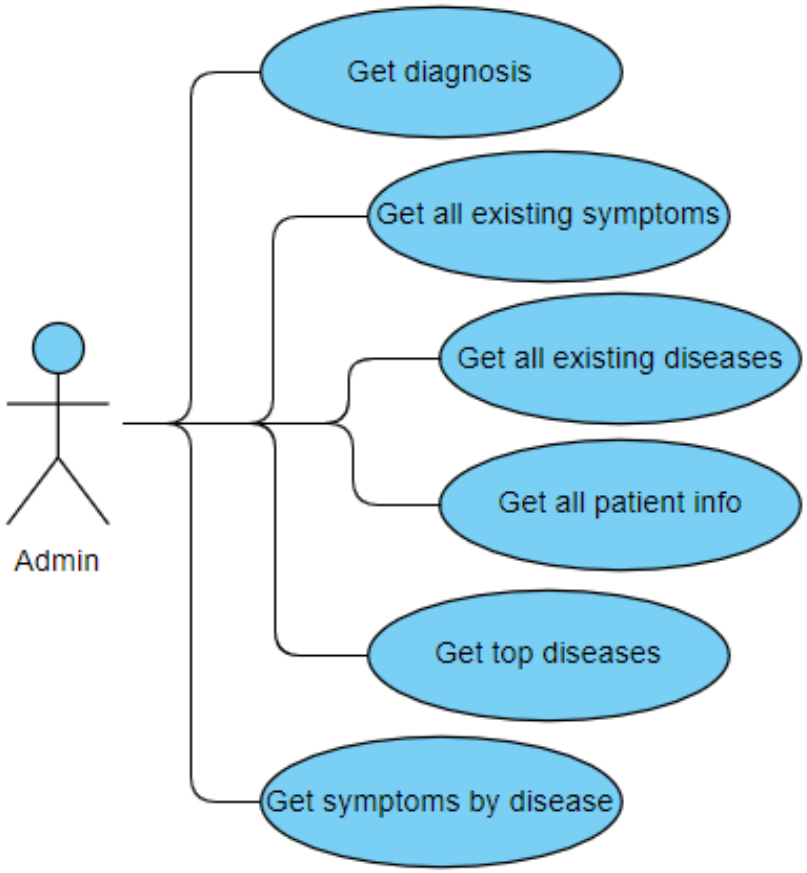

Рисунок 4.3 - Варіанти використання адміністратора

5 РОЗРОБКА БАЗИ ДАНИХ

5.1 Проектування бази даних

На етапі логічного та фізичного проектування було використано програмний засіб Oracle SQL Developer, за допомогою якого було створено усі необхідні таблиці та зв’язки, що точно відображають функціонування створюваної системи.

Модель даних показана на рисунку 4.1.

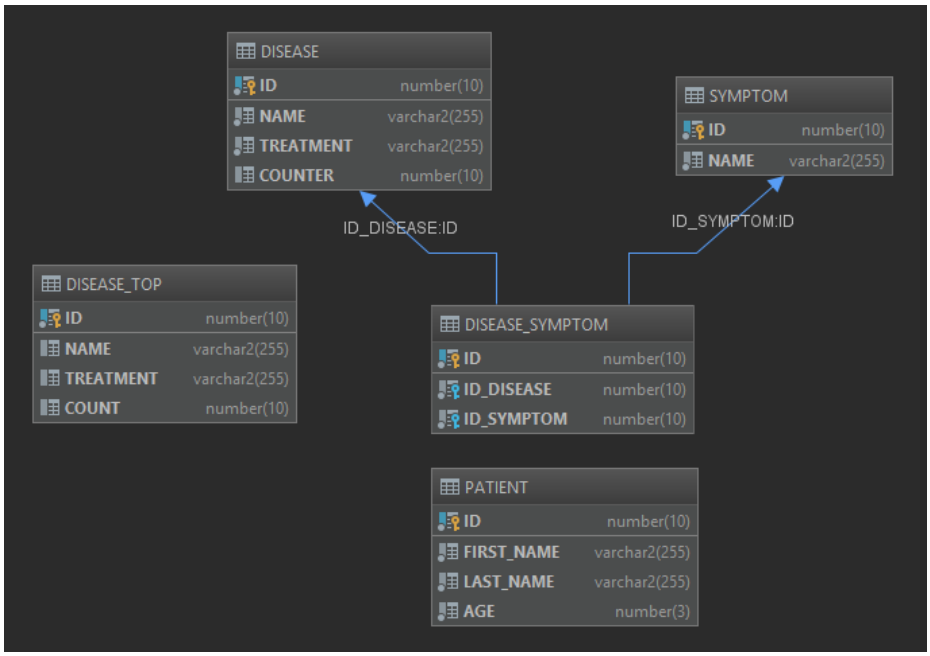

Рисунок 5.1 – Модель даних

Для всіх ідентифікаторів (Id) були створені послідовності для автоматичного генерування значення та додавання його до бази. Створення таблиць здійснювалось за допомогою команди CREATE TABLE, далі розглянемо скрипти створення послідовностей та таблиць.

5.2 Створення бази даних

Приклад створення послідовностей приведений в лістингу 1.

```
CREATE SEQUENCE symptom_sequence start with 1
increment by 1;
CREATE SEQUENCE disease_sequence start with 1
increment by 1;
CREATE SEQUENCE patient_sequence start with 1
increment by 1;
CREATE SEQUENCE disease_top_sequence start with 1
increment by 1;
CREATE SEQUENCE disease_symptom_sequence start with 1
increment by 1;
```

Лістинг 1 – Створення послідовностей

Створювалися таблиці за допомогою команди CREATE TABLE, приклад створення таблиць приведений у лістингу 2.

```
CREATE TABLE PATIENT
(ID NUMBER(10) NOT NULL PRIMARY KEY,
FIRST_NAME VARCHAR2(255) NOT NULL,
LAST_NAME VARCHAR2(255) NOT NULL,
AGE NUMBER(3) NOT NULL
);

CREATE TABLE SYMPTOM
(ID NUMBER(10) NOT NULL PRIMARY KEY,
 NAME VARCHAR2(255) NOT NULL);
```

```
CREATE TABLE DISEASE
(ID NUMBER(10) NOT NULL PRIMARY KEY,
 NAME VARCHAR2(255) NOT NULL,
 DIAGNOSTICS VARCHAR2(255) NOT NULL,
 COUNTER NUMBER(10)
);

CREATE TABLE DISEASE_SYMPTOM
(ID NUMBER(10) NOT NULL PRIMARY KEY,
 ID_DISEASE NUMBER(10) NOT NULL,
 ID_SYMPTOM NUMBER(10) NOT NULL,
 CONSTRAINT FK_DISEASE FOREIGN KEY (ID_DISEASE) REFERENCES DISEASE (ID),
 CONSTRAINT FK_SYMPTOM_D FOREIGN KEY (ID_SYMPTOM) REFERENCES SYMPTOM (ID)
);

CREATE TABLE DISEASE_TOP
(ID NUMBER(10) NOT NULL PRIMARY KEY,
 NAME VARCHAR2(255) NOT NULL,
 TREATMENT VARCHAR2(255) NOT NULL,
 COUNT NUMBER(10)
);
```

Лістинг 2 – Створення таблиць

6 РОЗРОБКА ВЕБ-ДОДАТКУ

6.1 Розробка бекенд-частини

Для розробки бекенд-частини використовувалась мова програмування Java. Був створений Maven проект із залежностями на необхідні бібліотеки, перш за все – це Spring Framework. За допомогою анотацій вказуються класи-біни, які потім легко додаються та використовуються. Проект був розподілений на пакети.

У пакеті Domain зберігаються усі необхідні класи, що використовуються для зв’язку із таблицями бази даних за допомогою Hibernate. Для цього необхідно над класом вказати анотацію @Entity та над полями @Column(name = “BD_Field_Name).

Пакет Controller призначений для зберігання контролерів, котрі отримують запити від фронтенд-частини, обробляють їх та надсилають відповідь. Для цього над класом необхідно вказати анотацію @Controller, а над методами анотацію @RequestMapping (value = “/someUrl”), та вказати тип очікуваного запиту (наприклад, method = RequestMethod.POST).

Контролер використовує сервіс для взаємодії з базою даних. Тому в пакеті Service знаходяться класи, що відмічені анотацією @Service, та в яких і знаходиться логіка обробки запитів. Сервіси можуть звертатися до інших сервісів, інстанціюючи їх за допомогою @Autowired і до репозиторію, або "dao" (Data Access Object).

Для зберігання таких "dao" класів існує пакет Repository, в якому вже виконуються запити до бази даних. Такі класи мають бути відмічені анотацією @Repository. Також, слід використовувати анотацію @Transactional для правильного виконання усіх методів роботи з базою.

Іноді буває достатньо відповісти на запит із фронтенду статусом, тобто Http.Status.OK, якщо все пройшло вдало, або, наприклад, Http.Status.BAD_REQUEST, якщо у процесі обробки виникла помилка. Але у більшості запитів необхідно окрім статусу відправити певні дані, об’єкт даних. Для того, щоб не виникло помилок при передачі даних, цей об’єкт конвертується у

JSON, звичний формат даних для Angularjs. Але не завжди об’єкти з фронтенду повністю відображають об’єкти бекенду, тому для конвертації таких "конфліктуючих" об’єктів використовуються конвертери, що розташовані у пакеті Converter. Таким чином, структура бекенд-частини має вигляд, як показано на рисунку 6.1.

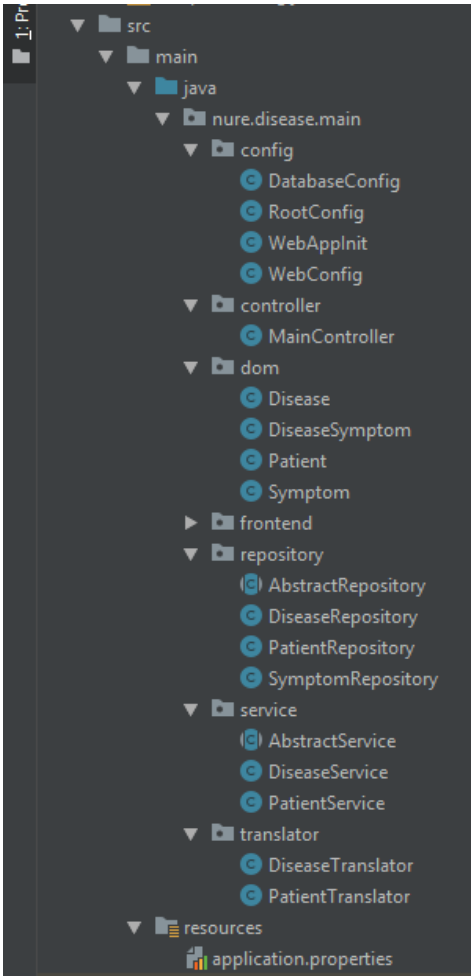

Рисунок 6.1 – Структура бекенд-частини

6.2 Розробка фронтенд-частини

Для розробки фронтенд-частини використовувався шаблон на основі Angularjs + EcmaScript6 із використанням Angular Material.

Усі компоненти складаються із трьох файлів: component.html, component.js,

component.less, відповідно кожен компонент має свій сервіс, з якого і відправляються запити на бекенд.

Приклад сервісу та функцію відправлення POST запиту для діагностування хвобори показаний на рисунку 6.2.

```
export default class DataService {

    static get $inject() {
        return ['$q', '$http'];
    }

    constructor($q, $http) {
        this.$http = $http;
        this.baseUrl = 'http://localhost:8086/Desease_war_exploded';
    }

    requestDiagnose(userData) {
        return this.$http.post(`${this.baseUrl}/getDisease`, userData);
    }
}
```

Рисунок 6.2 – Відправлення POST запиту

Обробка цього запиту відбувається у контролері, як показано на рисунку 6.3.

```
diagnose() {
    if (!this.userData.firstName || !this.userData.lastName || !this.userData.age || !this.userData.symptoms[0]) {
        this.toastr.error('You missed some data or made an error!', 'Attention!');
    } else {
        this.DataService.requestDiagnose(this.userData).then((disease) => {
            this.toastr.success('We found disease in our database!', 'Good news!');
            this.realDiseases = disease.data;
        }, (error) => {
            console.error(error);
            this.toastr.error("We didn't found anything!", 'Oops!');
        });
    }
}
```

Рисунок 6.3 – Обробка відповіді POST запиту

Таким чином, структура фронтенд-частини представлена на рисунку 6.4.

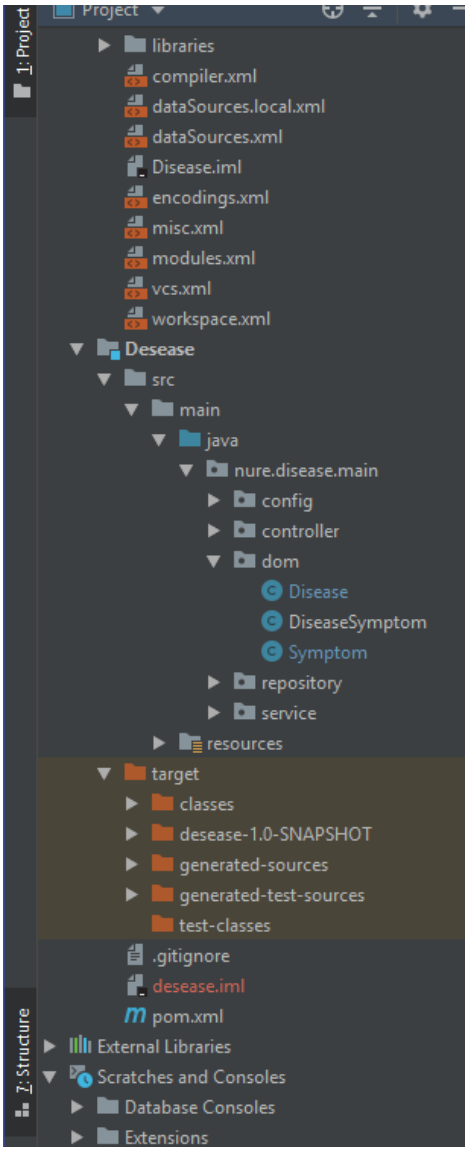

Рисунок 6.4 – Структура фронтенд-частини

7 ТЕСТУВАННЯ ВЕБ-ДОДАТКУ

Запустимо додаток та відкриємо посилання http://localhost:8089 у браузері. Головна сторінка виглядає, як показано на рисунку 7.1.

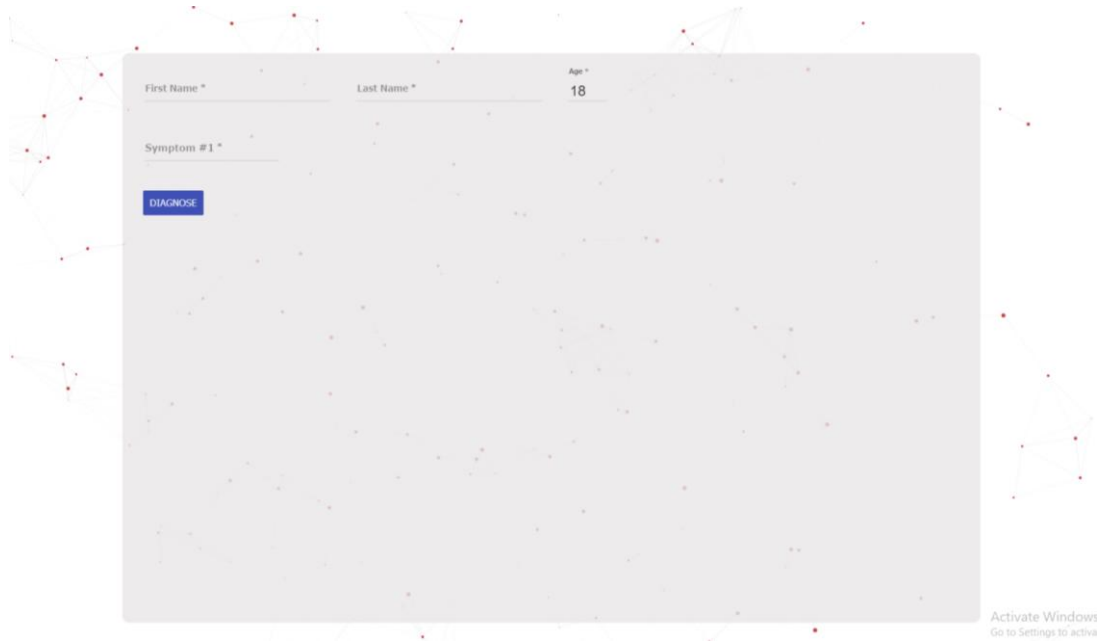

Рисунок 7.1 – Головна сторінка додатку

Рисунок 7.2 – Сповіщення про помилку

Як бачимо, відразу є можливість ввести необхідні дані та отримати результат.

Для цього введемо ім’я, прізвище, вік та оберемо симптоми. Як мінімум один симптом має бути заданий, інакше ми отримаємо сповіщення про недостатню кількість даних, як представлено на рисунку 7.2.

Теперь заповнимо усі необхідні поля, як показано на рисунку 7.3, та натиснемо «Diagnosis» для отримання результатів діагностування.

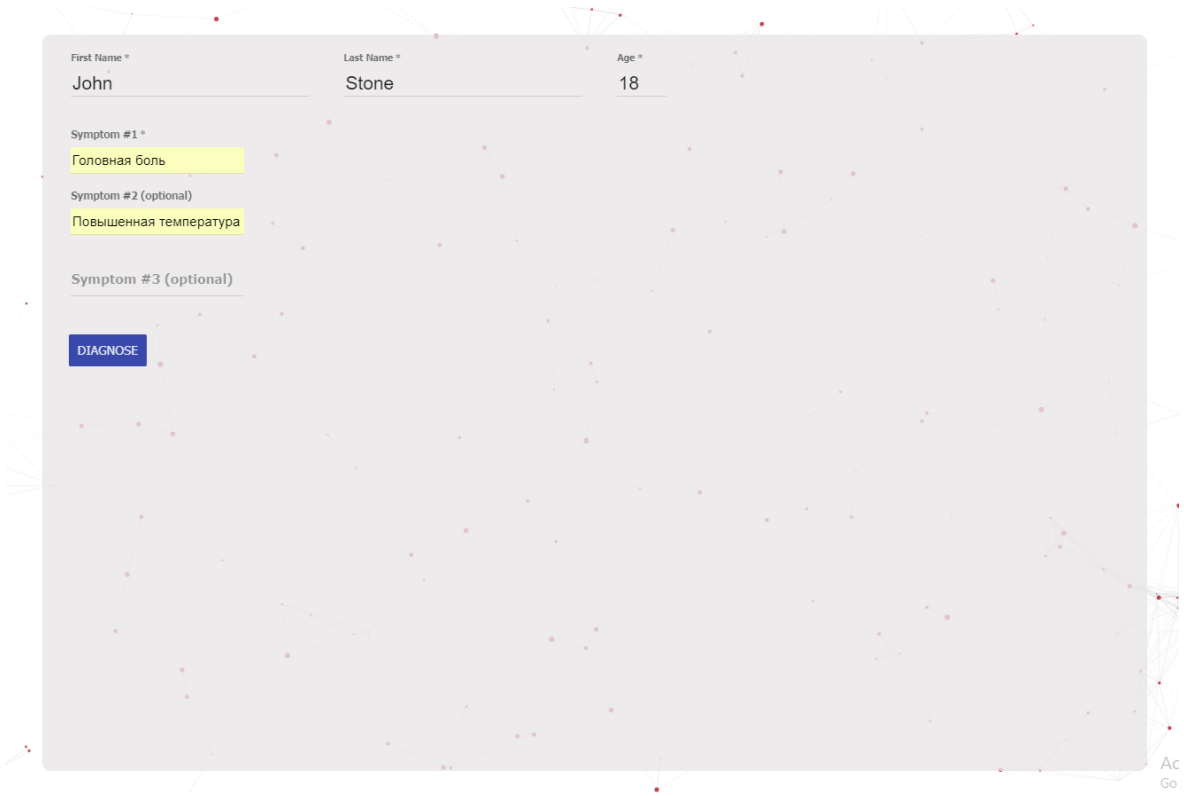

Рисунок 7.3 – Правильний запит

Тож усі дані були введено коректно, і запит було успішно відправлено на обробку. Після цього отримуємо наступний результат, як показано на рисунку 7.4.

Бачимо, що запит успішно виконаний, за його результатами можемо зробити припущення про своє захворювання за заданими симптомами. Також, ми отримали короткий запис про необхідну інформацію по діагностіці кожного з цих захворювань.

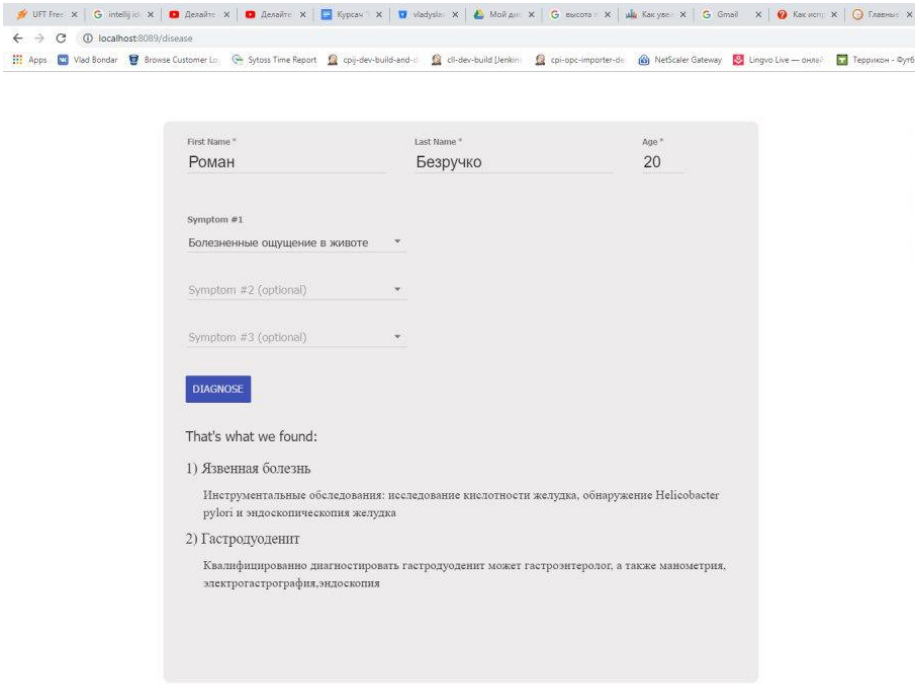

Рисунок 7.4 – Результат роботи додатку

Далі розглянемо можливості адміністратора. Для того, щоб дізнатися, яка ж хвороба була найбільш часто діагностована, необхідно виконати наступний SQL-запит, як показано на рисунку 7.5.

```
select name, COUNTER
from DISEASE
where COUNTER = (select max(COUNTER) from DISEASE);
```

Рисунок 7.5 – SQL Запит

Таким чином ми отримуємо результат, що сповіщаю адміністратора про те, що найчастіше була діагностована пневмонія (10 разів), як показано на рисунку 7.6, і можна розглядати варіанти розвитку системи з оглядом на отриманий результат.

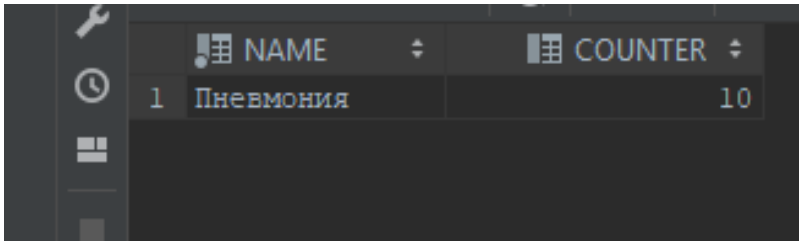

Рисунок 7.6 – Результат виконання запиту

Таким чином, був протестований увесь функціонал системи, усі дані були успішно додані та оброблені.

ВИСНОВКИ

У результаті виконання магістирської роботи була спроектована та розроблена база даних за допомогою СУБД Oracle, створений Java-додаток, що успішно використовує Hibernate для підключення до бази, а також розроблений фроненд за участю Angularjs та Angular Material.

Система отримує запити різного типу з фронтенд частини, обробляє їх за допомогою контролерів, сервісів та репозиторіїв та відсилає необхідні об’єкти або статуси у відповідь.

Робота веб-додатку була протестована, усі дані успішно додаються до бази, а також швидко оновлюються на фронтенд частині.

У подальшому можливе розширення системи, додавання нового функціоналу у разі успіху тестової версії веб-додатку та надмірного потоку пацієнтів. Є можливості до співпраці із багатьма лікарнями одного міста, та декількома типами захворювань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Д. Крёнке. Теория и практика построения баз данных, 8-е издание. – СПб.: Питер. – 2009. – 800 с.

2. К.Дж. Дейт. Введение в системы баз данных, 8-е издание.: Пер. С англ. – М. Изд. дом «Вильямс». – 2010. – 1328 с.

3. Косинський В.І. Сучасні інформаційні технології : навч. посіб. / В.І. Косинський, О.Ф. Швець. — К. : Знання, 2011. — 318 с.

4. М’якшило О.М. Моделювання баз даних засобами Case - технології ERWin: для студ. напряму 0804 “Комп‘ютерні науки” всіх форм навчання – К.: НУХТ, 2007 – 60 с.

5 Базы данных – Понятие базы данных // [Електронний ресурс]// Режим доступу: http://www.site-do.ru/db/db1.php (Дата звернення 23.05.19)

6 База даних // [Електронний ресурс]// Режим доступу: https://ru.wikipedia.org/wiki/База_даних (Дата звернення 23.05.19)

7 Основні відомості про бази даних // [Електроний ресурс]// Режим доступу: http://office.microsoft.com/uk-ua/access-help/HA010064450.aspx (Дата звернення 25.05.19)

8 Руководство по Hibernate // [Електроний ресурс]// Режим доступу: https://proselyte.net/tutorials/hibernate-tutorial/architecture/ (Дата звернення 25.05.19)

9 Основы SpringMVC Framework // [Електроний ресурс]// Режим доступу: https://proselyte.net/tutorials/spring-tutorial-full-version/spring-mvc-framework/ (Дата звернення 27.05.19)

10 Характеристика СУБД Oracle // [Електроний ресурс]// Режим доступу: http://www.omega.ru/oracleinfo.html (Дата звернення 28.05.19)

11 Intellij IDEA Guide // [Електроний ресурс]// Режим доступу: https://jetbrains.ru/products/idea/ (Дата звернення 27.05.19)

12 Номенклатура хвороб // [Електроний ресурс]// Режим доступу: http://medical-enc.com.ua/nomenklatura_bolezney.htm (Дата звернення 29.05.19)

13 СТАН МЕДИЧНОЇ СФЕРИ В УКРАЇНІ // [Електроний ресурс]// Режим доступу:http://ratinggroup.ua/research/ukraine/6a36be500ecc1cac19abf8680a0c51ac.html

14 МЕТОДИ ДІАГНОСТИКИ // [Електроний ресурс]// Режим доступу: http://www.likar.info/metodyi-diagnostiki/article-59038-metody-diagnostiki-stsintigrafiya/(Дата звернення 01.06.19)

15 А.В. Єпішина Пропедевтика внутрішніх захворювань Тернопіль.2001. 768