МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«ІМПЛЕМЕНТАЦІЯ ДОСТУПУ ДО КОРПОРАТИВНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ НА ОСНОВІ ДОДАТКІВ ANDROID»**

Виконав: студент 2 курсу, групи 8.1219-з

спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)

освітньої програми інженерія програмного забезпечення
(назва освітньої програми)

Турейський А.С.
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.- м.н Горбенко В.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення

(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

Лісняк А.О.

(підпис)

« » 2020 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Турейському Андрію Сергійовичу

(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Імплементація доступу до корпоративної інформаційної системи на основі додатків Android

керівник роботи (проекту) Горбенко Віталій Іванович, доцент кафедри програмної інженерії к.ф.-м.н.

(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 20 » травня 2020 року № 577-с

2. Строк подання студентом роботи

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі
2. Основні теоретичні відомості

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація

6. Консультанти розділів роботи

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

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	15.06.20	
2.	Збір вихідних даних.	20.07.20	
3.	Обробка методичних та теоретичних	17.08.20	
	джерел.		
4.	Розробка першого розділу.	08.10.20	
5.	Розробка другого розділу.	02.11.20	
6.	Оформлення та нормоконтроль	01.12.20	
	кваліфікаційної роботи.		
7.	Захист кваліфікаційної роботи.	9.12.2020	

Студент _____ А.С. Турейський
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.І. Горбенко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.В. Кудін
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра « Імплементація доступу до корпоративної інформаційної системи на основі додатків Android »: 46 с., 29 рис., 20 джерел, 2 додатка.

JAVA, ANDROID, ІМПЛЕМЕНТАЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА.

Об’єкт дослідження – доступ до інформаційної системи завдяки android додатку.

Мета роботи: дослідити спосіб підключення android додатку до інформаційної системи.

Методи дослідження – аналіз, формалізація, синтез, моделювання.

Кваліфікаційна робота присвячена дослідженню реалізації Android додатку для інформаційної системи. В ході роботи було створено модельну інформаційну систему та розроблено Android додаток для роботи з нею. Для розробки серверу та Android додатку використовувалась мова програмування Java.

ABSTRACT

Master’s Qualification Thesis «Implementation of access to corporate information system based on Android applications»: 46 pages, 29 figures, 20 references, 2 supplements.

JAVA, ANDROID, IMPLEMENTATION, INFORMATION SYSTEM.

Object of the study – research is access to the information system through the android application.

Aim of the study: to explore how to connect the android application to the information system.

Method of research – analysis, formalization, synthesis, modeling.

Qualification work is devoted to the study of the implementation of the Android application for the information system. During the work, a model information system was created and an Android application was developed to work with it. The Java programming language was used to develop the server and Android application

ЗМІСТ

Завдання на кваліфікаційну роботу…… 2
Реферат…… 4
Abstract…… 5
Вступ…… 7
1 Теоретична частина ……..…… 8
1.1 Корпоративна інформаційна система …….. 8
1.2 Етапи впровадження КІС…… 14
2 Практична частина.…….…….. 17
3 Реалізація проекту та тестування……. 24
Висновки…….. 36
Перелік посилань…….....……... 37
Додаток А……. 39
Додаток Б……...……48

ВСТУП

Сьогодні основним фактором створення довгочасної конкуренто спроможної і більш інвестиційної привабливості організації стають оптимальні політики адміністрування бізнесом. Ефективне адміністрування - це такий же потенціал, як гроші або матеріальні цінності. Саме цей потенціал допомагає динамічно відповідати на постійно змінну ринкового становища, тримати під контролем всі сторони активності організації. Розвиваючи інформаційні системи (ІС) необхідно намагатися до виробничої сегменту бізнесу, створюючи потенціал не тільки примітивного набору інформації, розвитку бізнес процесів і інших характеристик впровадження, але задовольняти можливість аналітичної обробки інформації на рівні якості продукту, технологій, ресурсів і так далі.

Одним із рішень таких задач може послужити впровадження єдиного додатка (програмного забезпечення) для доступу до корпоративної інформаційної системи. Завдяки такому впровадженню з’явиться можливість швидкого доступу до інформації, її перегляду, редагуванню, тощо. Основні переваги які будуть отриманні при такому впроваджені:

— Підвищення продуктивності праці, інтегрування фінансової інформації, стандартизація інформації по персоналу.

— Для менеджерів різних рівній, надає можливість ефективно оброблювати інформацію, контролювати фінансові операції, інформативну систему звітів.

Фундаментальною основою для ефективного управління організацією це якісне управління.

Метою кваліфікаційної роботи полягає у можливості доступу до ІС за допомогою android додатку для підвищення швидкості обробки інформації менеджерів різних рівнів корпорації.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Корпоративна інформаційна система

Корпоративна інформаційна система (КІС) – це відкрита інтегрована автоматизована система реального часу по автоматизації бізнес-процесів компанії всіх рівнів, в тому числі, і бізнес-процесів прийняття управлінських рішень [1]. За своєю архітектурою КІС – це комплекс багатьох унікальних додатків, та спеціалізованих програмних платформ, які об’єднанні в єдину інформаційно-однорідну систему, яка вирішує унікальні задачі, конкретні в своїй галузі для кожного підприємства[2].

Класифікації КІС:

— На замовлення – це унікальні КІС яка не має аналогів,та створенні на замовлення для корпорації з унікальними характеристиками.

— Тиражовані – такі КІС проходять етапи адаптації до роботи корпорації.

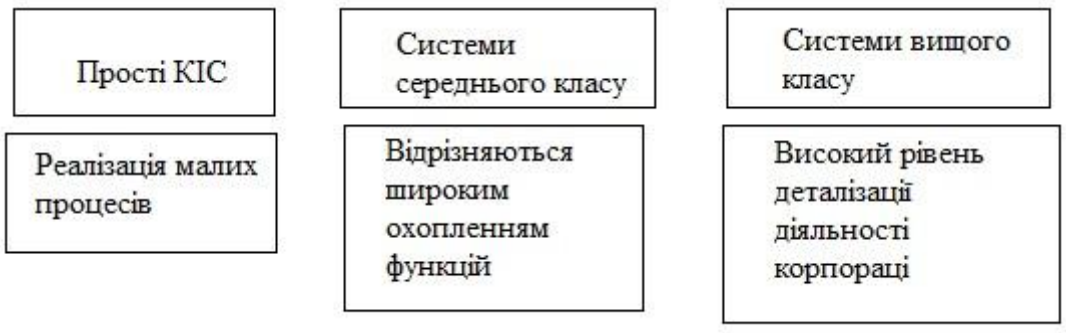


Рисунок 1.1 — Класи КІС.

Еволюція КІС зображена на рисунку 1.2.

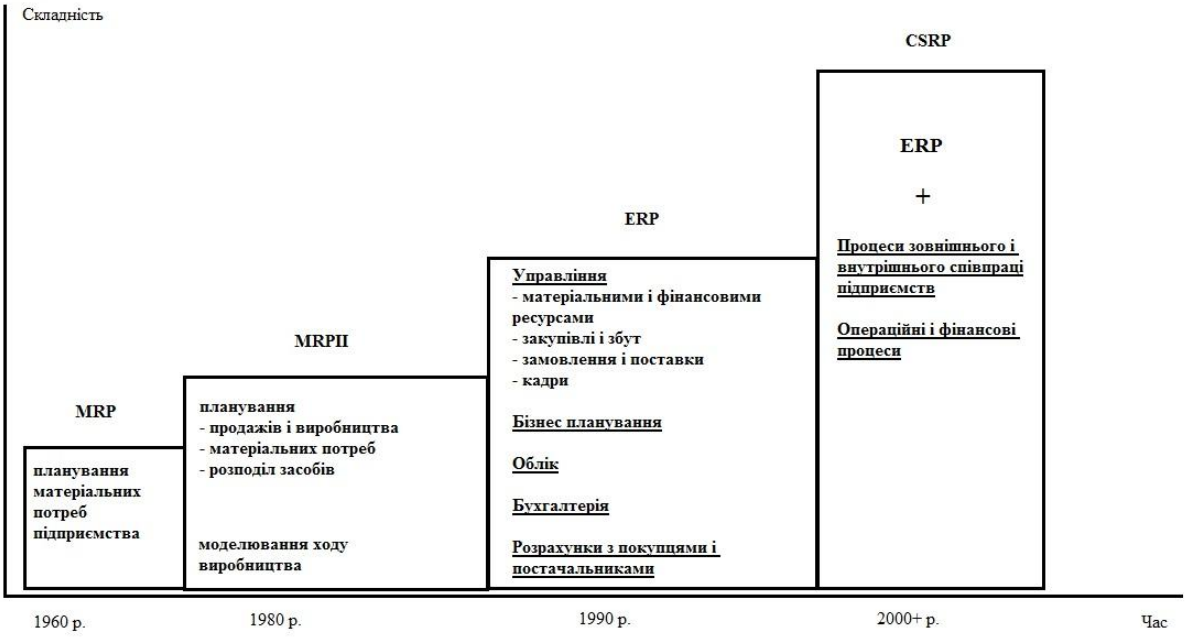


Рисунок 1.2. — Еволюція КІС

Планування потреб в матеріалах — (MRP – Material Requirements Planning) концепція мінімізації витрат запасів складу. Недоліками такої системи є значний об'єм даних. Структура системи приведена на рисунку 1.3

Рисунок 1.3 — Структура системи MRP.

Планування виробничих ресурсів — (Manufacturing Resource Planning – MRP2) концепція в якій ведеться облік та планування ресурсів корпорацій. Ієрархія виробничих планів в системі приведена на рисунку 1.4.

Рисунок 1.4 — Ієрархія виробничих планів в системі.

Планування ресурсів корпорації — (Enterprise Resource Planning – ERP) в моделі закладено принцип створення сховища даних (репозиторію), інформація яка зберігається могла бути доступна всім працівникам організації. Структура такої системи приведена на рисунку 1.5

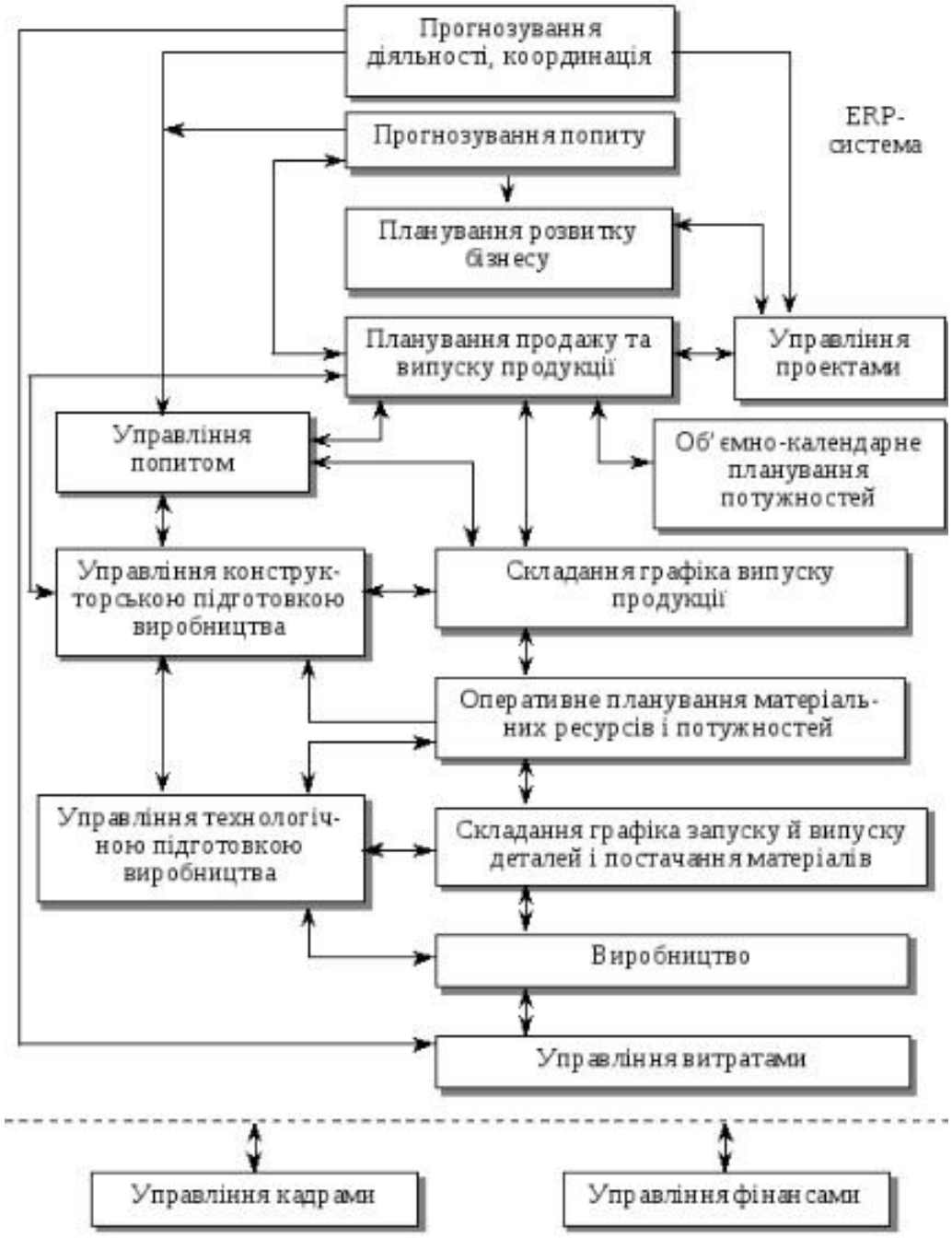


Рисунок 1.5 — Структурна схема ERP-системи.

Планування ресурсів, синхронізованих з покупцем — (CSRP – Customer Synchronized Resource Planning) ця модель використовує функціонал ERP та включає в себе повний цикл від побудови проекту, до після продажного обслуговування. На рисунку 1.6 приведено схему формування інформації на

підставі вимог покупця. На рисунку 1.7 схема планування ресурсів синхронізованого з покупцем [3][4].

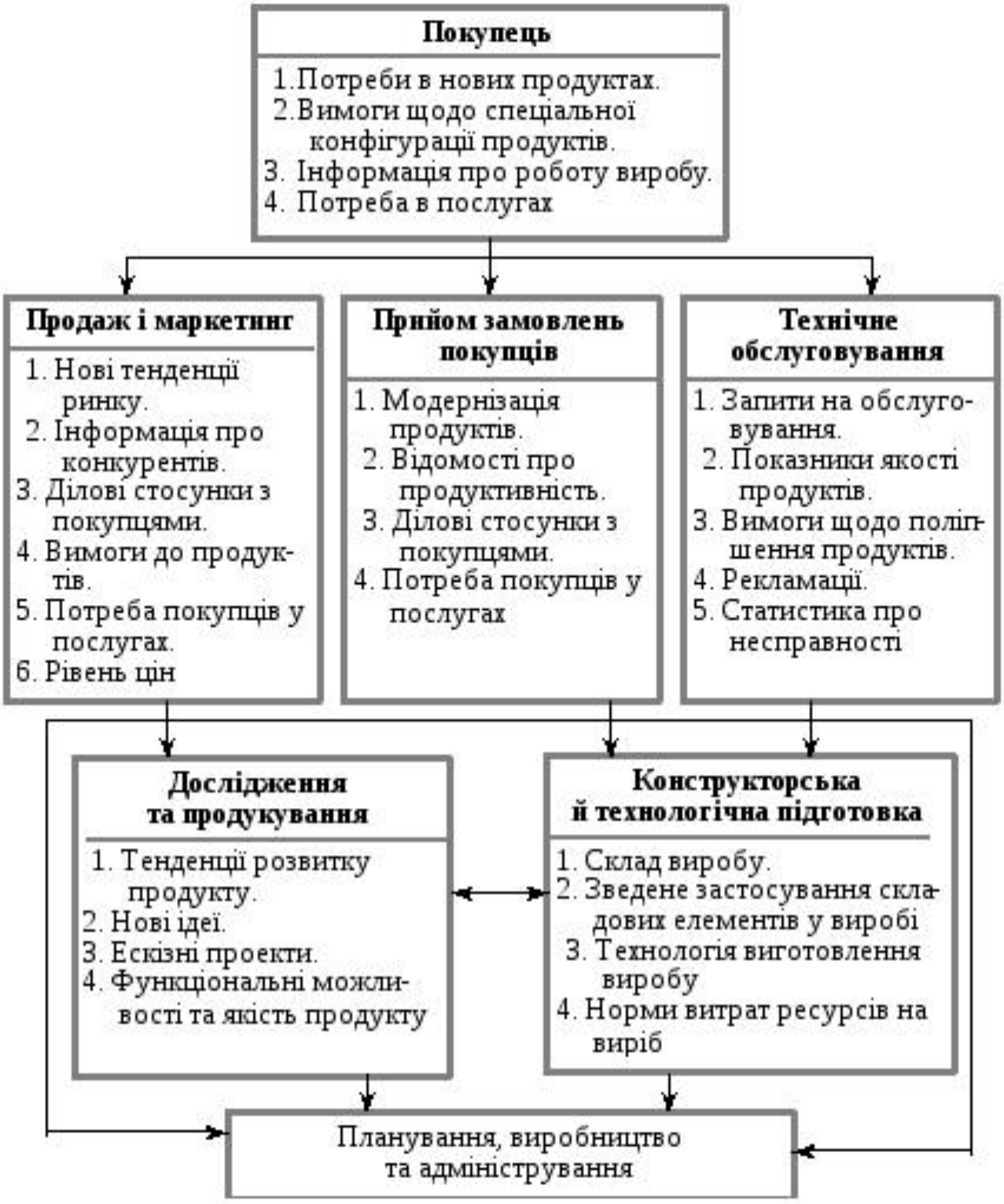


Рисунок 1.6 — Схема формування інформації на підставі вимог покупця.

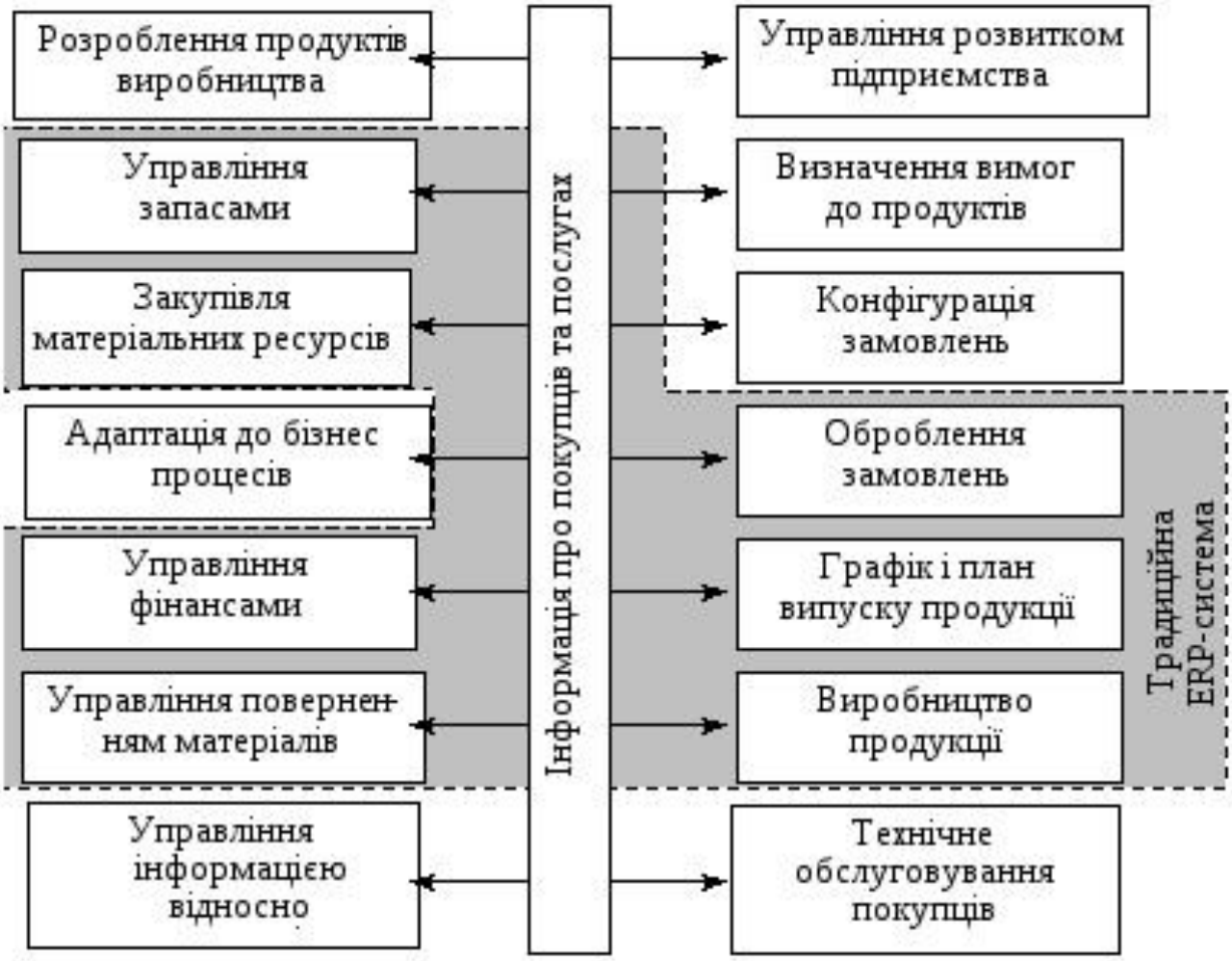


Рисунок 1.7 — Схема планування ресурсів синхронізованого з покупцем

Імплементація доступу до інформаційної системи – отримання інформації з ресурсів корпоративної системи. Фундаментальною основою для ефективного управління організацією це якісне управління. Під цим мають на увазі, головніше за все, чіткість в діях управління та адміністрування (топ-менеджмент цілісний тоді, коли використовує ідентичну аналітичну інформацією, ефективно та швидко відповідає на зміну обставин).

У сучасному положенні це можливо задовольнити тільки тоді, коли володіння корпоративної інформаційної системи, що єднає в собі всі

інформаційні засоби корпорації, і присуджує їх керуючому персоналу, для її щоденної і аналітичної роботи.

Будь-які значні зміни зовнішньої сфери вражає ті чи інші інтереси підприємства, та віддзеркалюються на її результатах. Сучасні корпоративні інформаційні системи, прогнозують інтенсивне оновлення показників, що надає здійснювати безупинний нагляд і контроль викривлення від цілеспрямованих результатів.

Намірами ініціювання КІС на підприємстві найчастіше є: виконування переходу на якісно сучасний розвиток прийняття оперативних і важливих адміністративних рішень, внаслідок існування повного та своєчасного інформування про активність організації в масі перетинів; встановлення прозорості, максимального інспектування і гнучкій діяльністю організації для власників і топ-менеджерів організації; підйом матеріальних прибутків компанії в наслідок наведення устрою в процедурах обліку і нагляду за фінансовими і матеріальними потоками [5].

1.2 Корпоративна інформаційна система

Етапи впровадження ІС[6]:

— підготовка;

— розморожування;

— зміни;

— заморожування.

На етапі підготовки формулюється сутність наступної зміни, проводиться аналіз його необхідності, а також виявляється різниця між поточним і бажаним станом справ на підприємстві. До основних положень першого етапу процесу впровадження ІС відносяться:

— визначення зміни: впровадження корпоративної ІС, що відповідає найбільш загальним для різних типів компаній функцій, плюс додаткове

програмне забезпечення, що відображає специфіку діяльності компанії за рахунок галузевих рішень;

— розуміння поточного і бажаного стану організації, ключовими компонентами цього розуміння є завдання, що виконуються ІС; організаційні структури і системи, що включають лінії підзвітності, посадові інструкції, механізми контролю та ін., організаційну культуру, а також облік необхідних навичок, знань і досвіду людей, яких торкнуться ці зміни;

— вивчення ринку корпоративних ІС: масштабів, вартості, застосовності для конкретного підприємства;

— вибір відповідної ІС, заснований на вартості окремих систем, порівняно змісту функцій, які виконуються компанією, з вмістом функціональних модулів пропонованих ІС.

На етапі розморожування здійснюється підготовка персоналу до майбутніх змін. Людей переконують в необхідності відмовитися від старих методів роботи і почати освоювати нові. Це необхідно у випадках опору змінам.

Поради керівника підприємства з менеджерами, відповідальними за впровадження і функціонування ІС, а також з менеджерами, які є користувачами ІС:

— спільні обговорення процесів впровадження виробників корпоративних ІС і співробітників компанії. Розуміння співробітниками суті впроваджуваної на підприємстві ІС зацікавлять їх більш швидким освоєнні системи;

— навчання співробітників компанії;

— участь і залучення;

— допомога та підтримка.

На етапі зміни необхідно обрати певну стратегію. Існує два основних підходу до вирішення завдання впровадження ІС:

— поетапна розробка корпоративних ІС власними силами (включаючи покупку готових комп'ютерних програм по окремих модулів системи);

— впровадження готової ІС корпоративного рівня. Великі корпорації зазвичай віддають перевагу готовим програмам.

На етапі заморожування зміна зазнає невдачі, якщо закріпленню зміни (заморожування) приділяється недостатньо уваги. Існує небезпека, що рутинні справи не дозволять успішно завершити процес заморожування. У цьому випадку необхідний ретельний контроль і управління етапом заморожування.

З технічної точки зору система має включати в себе два головних аспекти: це віддалений сервер та We -сервер, один із яких повинен бути БД. Клієнтська станція і сервер повинні бути підключені до однієї локальної мережі. СУБД повинна буди представлена однією з системою: MySQL, Oracle, або іншою. В свою чергу WE -сервіс повинен базуватися на: Apache, Personal Web-Server, або іншому середовищі яке включає підтримку мови розмітки гіпертекстів. Драйвер доступу до БД повинен буди JDBC, ODBS, та ін. Для прикладу на рисунку 1.8 показана трирівнева архітектура КІС[7].

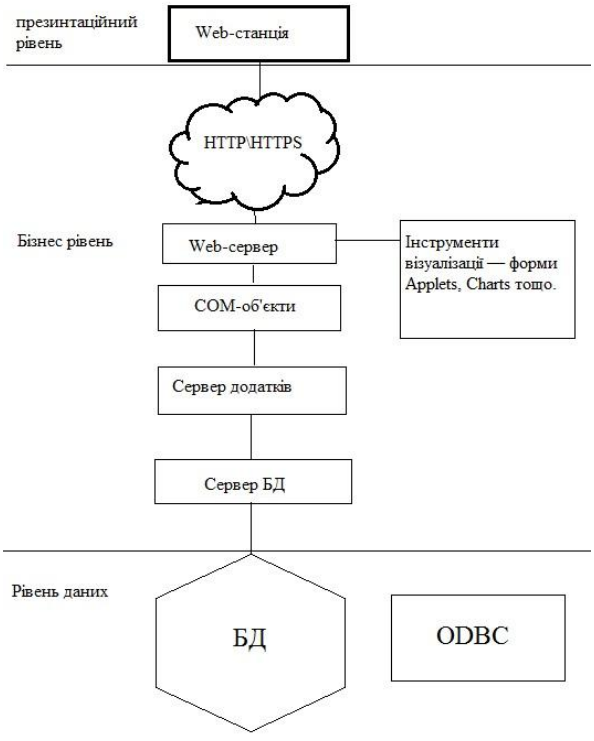


Рисунок 1.8 — Трирівнева архітектура КІС.

2 ПРАКТИЧНА ЧАСТИНА

Для вирішення задачі використовувалось наступне програмне забезпечення:

— мова програмування: Java;

— середовище розробки: IntelliJ IDEA, Android studio;

— система управління бази даних: PostgreSQL.

Java – строго-типизована об'єктно-орієнтована мова програмування. Програми на Java транслюються в байт-код Java, який виконується віртуальною машиною Java (JVM) - програмою, обробній байтовий код і передавальної інструкції обладнанню як інтерпретатор.

Розробка Java почалася в 1990 році, перша офіційна версія - Java 1.0, - була випущена лише 21 січня 1996 року. Усередині Java існує кілька основних сімейств технологій.

Java SE — Java Standard Edition, основне видання Java, містить компілятори, API, Java Runtime Environment; підходить для створення призначених для користувача додатків, в першу чергу - для настільних систем.

Java EE — Java Enterprise Edition, являє собою набір специфікацій для створення програмного забезпечення рівня підприємства. У 2017–му проект Java EE був переданий Eclipse Foundation, після чого був перейменований в Jakarta EE. Модулі Java EE видалені з Java SE, починаючи з 11-ї версії[8].

Java Development Kit (скорочено JDK) – безкоштовний продукт компанії Oracle Corporation (раніше Sun Microsystems) включає в себе компілятор Java (javac) ,та компоненти розробника додатків на мові Java. JDK не містить в собі середовище розробника Java, тому розробник, змушений використовувати інші редактори та розробляти свої програми, використовуючи утиліти командного рядка[9]. Склад JDK показаний на рисунку 1.9.

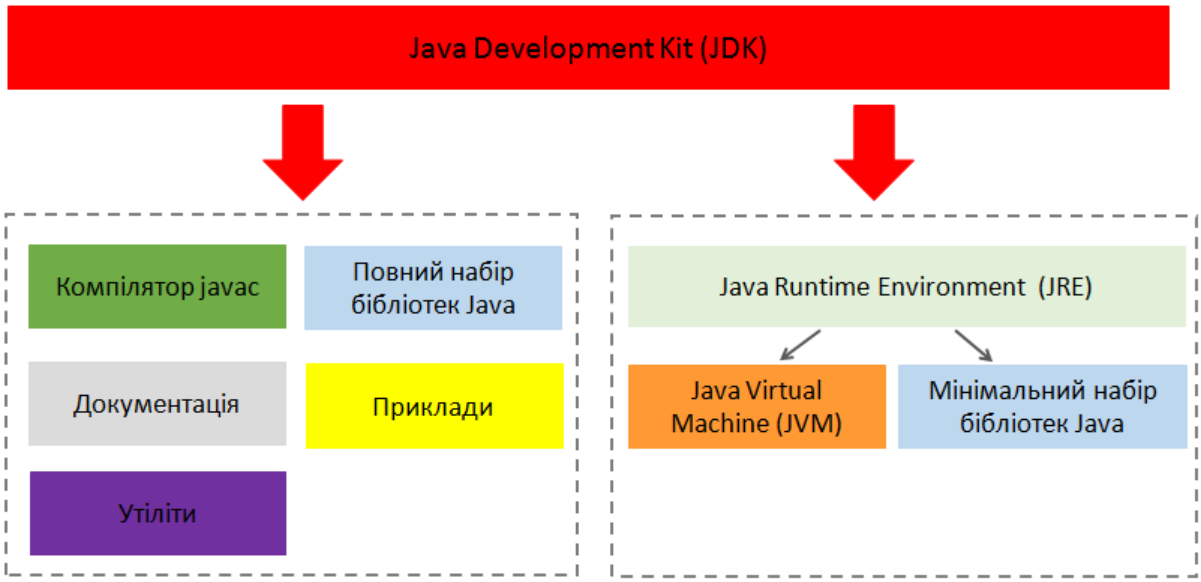


Рисунок 2.1 — Склад JDK.

Java Virtual Machine віртуальна машина Java (скорочено Java VM, JVM) – основна частина виконує системи Java, так званої Java Runtime Environment (JRE). JVM виконує байт-код Java, який створений з вихідного тексту Java-програми компілятором Java (javac). JVM може також запускати код для виконання програм, який написаний не на Java,наприклад код на мові Ada може бути скомпільовано в Java, який буде виконуватися JVM. Так як JVM доступна для багатьох апаратних і програмних платформ вона є ключовим компонентом Java. Java описують як « скомпільовано одного разу, запускається всюди» (compile once, run anywhere) [10]. Концепція роботи JVM показана на рисунку 1.10.

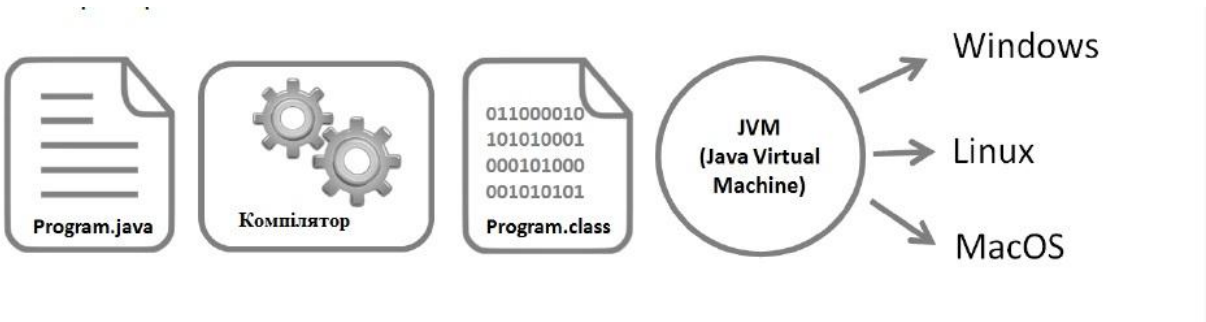


Рисунок 2.2 — Концепція роботи JVM.

IntelliJ IDEA – інтегроване середовище розробки програмам для різних мов програмування, таких як: Java, JavaScript, Python, розроблена компанією JetBrains. Починаючи з версії 9.0, середовище представлено в двох видах: Ultimate Edition та Community Edition . Community Edition є вільною версією, яка працює під ліцензією Apache 2.0, в ній реалізована повна підтримка Java SE, Kotlin, Groovy, Scala, та інтеграція з популярними системами управління версіями[11].

Android Studio – інтегроване середовище розробки (IDE) для платформи Android. Засобами цього додатку можливо створення продукту для смарфонів та різноманітних планшетів пристроїв що побудовані на операційної системі Android Wear, автомобільної мультімедії (Android Auto), окулярів Google Glass та телевізійних пристроїв що побудовані на TV версії Android. Під час розробки на платформі Android існують типові задачі, під яке інструментарій Android Studio вже адаптовано. У склад середовища входять інструменти які направлені на покращення процесу тестування програмних продуктів на роботу з різноманітними версіями операційної системи та можливості для створення додатків, що надають змогу програмним продуктам працювати на екранах пристроїв різних гаджетів, такі як годинники, різноманітні планшети,сучасні телефони, ноутбуки тощо).

Окрім наявних в IntelliJ IDEA інструментів, в Android Studio має в арсеналі декілька спеціальних функцій, одні з них це спеціальна система складання та тестування, що базується на можливостях Gradle, та підтримує засоби для інтеграції.

Нові функції з'являються з кожною новою версією Android Studio. На даний момент доступні наступні функції.

Розширений редактор макетів: WYSIWYG, здатність працювати з UI компонентами за допомогою Drag-and-Drop, функція попереднього перегляду макета на декількох конфігураціях екрану.

Збірка додатків, заснована на Gradle. Різні види збірок і генерація кількох .apk файлів, рефакторинг коду.

Статичний аналізатор коду (Lint), що дозволяє знаходити проблеми продуктивності, несумісності версій і інше.

Вбудований ProGuard і утиліта для підписування додатків. Шаблони основних макетів і компонентів Android. Підтримка розробки додатків для Android Wear і Android TV.

Вбудована підтримка Google Cloud Platform, яка включає в себе інтеграцію з сервісами Google Cloud Messaging і App Engine[12].

PostgreSQL – вільна об'єктно-реляційна система управління базами даних (СКБД). Сильними сторонами PostgreSQL вважаються:

— високопродуктивні і надійні механізми транзакцій і реплікації;

спадкування;

— можливість індексування геометричних об'єктів і наявність базується на ній розширення PostGIS;

— вбудована підтримка слабоструктурованих даних в форматі JSON з можливістю їх індексації;

— розширюваність (можливість створювати нові типи даних, типи індексів, мови програмування, модулі розширення, підключати будь-які зовнішні джерела даних).

PostgreSQL також може модифікувати БД декількома користувачами завдяки механізму Multiversion Concurrency Control (MVCC) [13].

Spring Framework (або коротко Spring) – універсальний фреймворк з відкритим вихідним кодом для Java-платформи. Компанія Apache у червні 2003 року випустила цей фреймворк вперше. Та вже в березні 2004 побачила світ перша стабільна версія 1.0.

Фреймворк вийшов та набув поширення в товаристві Java розробників як альтернатива та заміна Enterprise JavaBeans, хоча він не забезпечував якусь конкретну модель програмування, за рахунок великої свободи у створенні

програмних продуктів. Розробникам надається засоби, за допомогою яких можливе швидке та легке вирішення проблем, які трапляються під час створення продукту, особливо великого масштабу.

Розробники та фірми, при створенні програмного продукту, що базується на платформі Java, стикаються з багатьма завданнями, які Spring з легкістю вирішує. Фреймворк має дуже широку функціональність, тому важко виділити найважливіші структурні частини. Одна з причин популярності Spring є його серйозна інтеграція з платформою Java Enterprise, хоча даний фреймворк не має тісного зв’язку з нею.

Spring надає свій шар доступу до баз даних за допомогою JDBC. Крім того, він підтримує всі популярні ORM: Hibernate, JPA, JDO, EclipseLink, iBatis, Apache OJB, Apache Cayenne і т.п.

Для всіх цих фреймворків, Spring надає такі особливості:

— Управління ресурсами – автоматичне отримання і звільнення ресурсів бази даних;

— обробка винятків – переклад винятків при доступі до даних в виключення Spring-a;

— транзакційність – прозорі транзакції в операціях з даними;

— розпакування ресурсів – отримання об'єктів бази даних з пулу з'єднань;

— абстракція для обробки BLOB і CLOB[14].

Java Persistence API (JPA) – специфікація API Java EE, надає можливість зберігати в зручному вигляді Java-об'єкти в базі даних.

Існує кілька реалізацій цього інтерфейсу, одна з найпопулярніших використовує для цього Hibernate. JPA реалізує концепцію ORM.

Entity (Сутність) – POJO-клас, пов'язаний з БД за допомогою анотації (@Entity) або через XML[15].

Hibernate – найпопулярніша реалізація специфікації JPA, призначена для вирішення завдань об'єктно-реляційного відображення (ORM). Поширюється вільно на умовах GNU Lesser General Public License.

Метою Hibernate є звільнення розробника від значного обсягу порівняно низкорівневого програмування при роботі в об'єктно-орієнтованих засобах в реляційної базі даних. Розробник може використовувати Hibernate як в процесі проектування системи класів і таблиць « з нуля », так і для роботи з уже існуючою базою даних.

Hibernate забезпечує використання SQL-подібного мови Hibernate Query Language (HQL), який дозволяє виконувати SQL-подібні запити, записані поруч з об'єктами даних Hibernate. Запити критеріїв надаються як Об'єктно-орієнтована альтернатива до HQL [16].

Apache Maven – фреймворк для автоматизованого збирання проектів на основі їх структури в файлах на мові POM (англ. Project Object Model), що є підмножиною XML. Проект Maven випускається співтовариством Apache Software Foundation, та формально відноситься до Jakarta Project.

Maven забезпечує декларативну збірку проекту, в той час Apache Ant використовує імперативну автоматизації збирання. Специфікація знаходиться у файлах опису проекту. Специфікація описує в собі всі завдання по обробці файлів [17].

Thymeleaf – це шаблонизатор Java XML / XHTML / HTML5, який може працювати як в веб-середовищі, так і поза нею. Він краще підходить для обслуговування XHTML / HTML5 на рівні уявлення веб-додатків на основі MVC, але він може обробляти будь-який файл XML навіть в автономному режимі. Thymeleaf пропонує набір інтеграцій Spring, які дозволяють використовувати його як повнофункціональну заміну JSP в додатках Spring MVC.

Ці інтеграції дозволять:

— Зробити зіставлення методів в об'єктах Spring MVC Controller шаблонів, керованих Thymeleaf, точно так же, як робиться це з JSP;

— використовувати Spring Expression Language (Spring EL) замість OGNL в шаблонах;

— створювати форми в шаблонах, які повністю інтегровані з компонентами (bean) підтримки форм і прив'язками результатів, включаючи використання редакторів властивостей, служб перетворення і обробки помилок перевірки;

— відображати повідомлення інтернаціоналізації з файлів повідомлень, керованих Spring (через звичайні об'єкти MessageSource) ;

— знаходити шаблони, використовуючи власні механізми вирішення ресурсів Spring [18].

3 РЕАЛІЗАЦІЯ ПРОЕКТУ ТА ТЕСТУВАННЯ

В цілях роботи вирішення завдання було обрано створити модель КІС та підключити по мережі wi-fi android додаток на основі WebView.

Для створення Maven проекту було обрано сайт https://start.spring.io/ рисунок 3.1. Сайт призначений для створення Maven/Gradel проекту, за допомогою сайту можливо вибрати конфігурації та залежності які потрібні для проекту. Вихідний проект має всі необхідні файли для початку роботи. За основу проекту було взято керівництво Spring boot та Thymleaf з джерела [19], та модернізовано під задачу диплома. Код проекту приведено в додатку А.

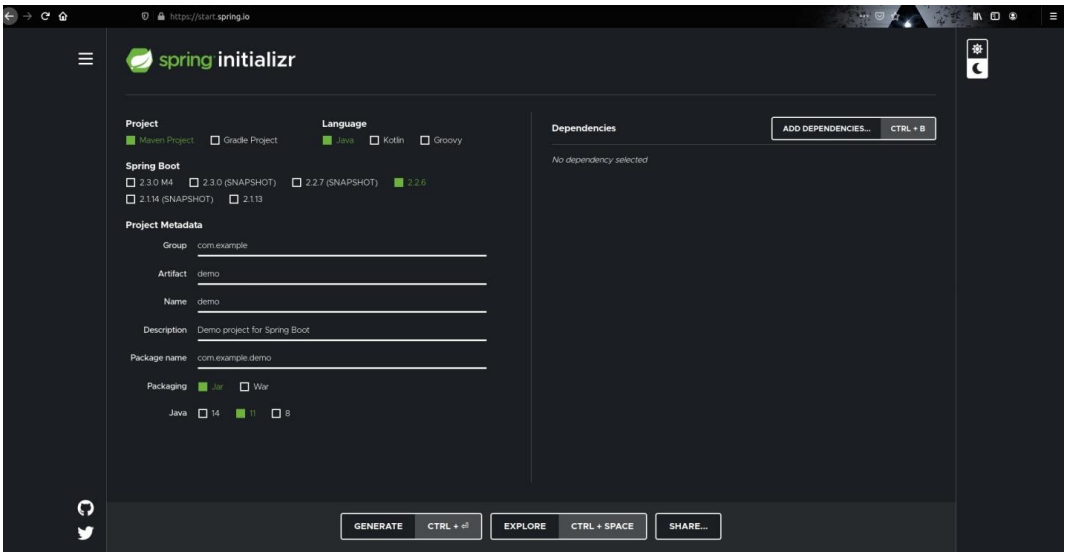


Рисунок 3.1 — Spring initializ.

В проекті було створено такі класи:

— MainControlletr.java;
— StaffFrom.java;
— Staff.java;
— StaffRepository.java;
— Style.css;

— AddStaff.html;

— index.html;

— ListStaff.html;

Древо проекту показано на рисунку 3.2.

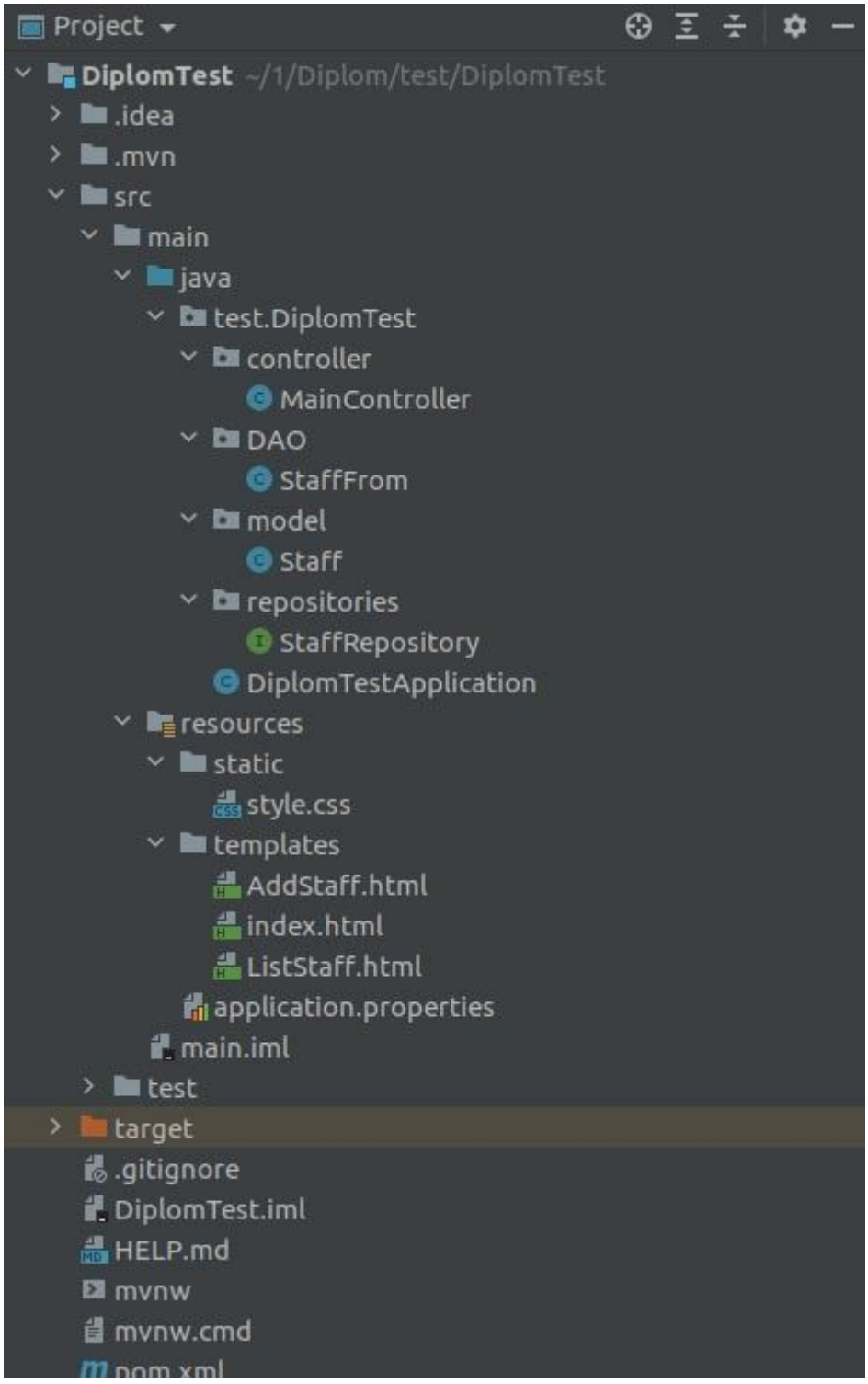


Рисунок 3.2 — Древо проекту.

Для роботи проекту використовувалась БД Postgres , та СУБД PostgreSQL, Підключення зображено на рисунок 3.3.

```
spring.datasource.url=jdbc:postgresql://localhost:5432/test
spring.datasource.username=test
spring.datasource.password=test
spring.jpa.generate-ddl=true
spring.jpa.show_sql=true
spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.properties.jdbc.lob.non_contextual_creation=true

spring.thymeleaf.cache=false
```

Рисунок 3.3 — Application.properties.

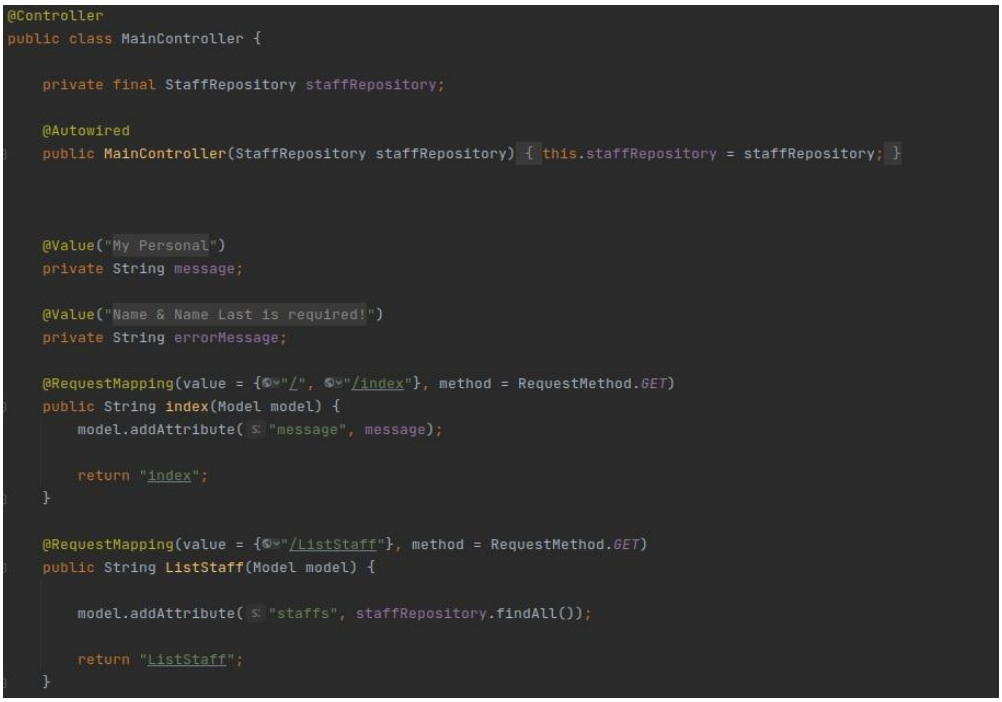
Роз’яснення класів в проекті:

MainController.java – це клас, призначений для безпосередньої обробки запитів від клієнта і повернення результатів. Основне завдання методів контролера - визначити необхідну дію, коректно прийняти дані, коректно повернути результат.

Клас позначається анотацією @Controller яка вказує що даний клас виконує роль контролера. Завдяки такій анотації, клас відпадає необхідність у спадкуванні стандартних класів або використання Servlet API. Ця анотація назначає клас як шаблон що вказує на його роль в проекті.

Анотація @RequestMapping вказує методам які написані в контролері на те, по якім адресам будуть вони доступні на клієнті. В даній анотації використовується параметр Value, котрий вказує на адрес, а також методи RequestMethod.GET, RequestMethod.POST. Ці методи використовуються для обробки запиту, в даному випадку це Get і Post запроси.

Анотація @Value вказує на значення за замовчуванням. В даному випадку використовується для виводу повідомлення рисунок 3.4.

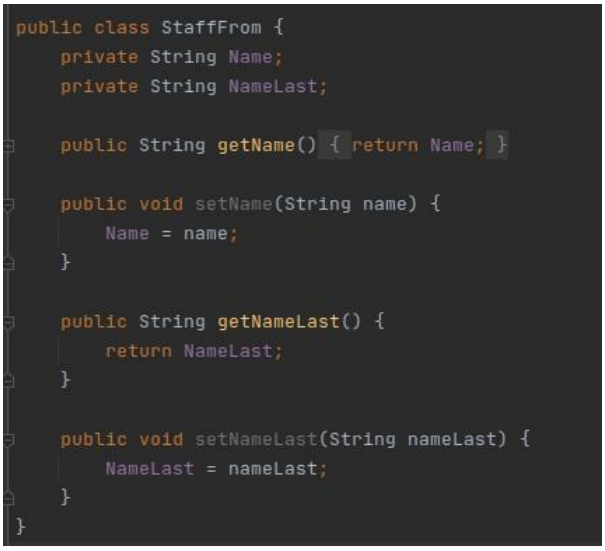


```
@Controller
public class MainController {

    private final StaffRepository staffRepository;

    @Autowired
    public MainController(StaffRepository staffRepository) { this.staffRepository = staffRepository; }



    @Value("My Personal")
    private String message;

    @Value("Name & Name Last is required!")
    private String errorMessage;

    @RequestMapping(value = {"/", "/index"}, method = RequestMethod.GET)
    public String index(Model model) {
        model.addAttribute( s: "message", message);

        return "index";
    }

    @RequestMapping(value = {"/ListStaff"}, method = RequestMethod.GET)
    public String ListStaff(Model model) {

        model.addAttribute( s: "staffs", staffRepository.findAll());

        return "ListStaff";
    }
```

Рисунок 3.4 — Частина коду класу MainController.

StaffFrom.java – клас-сутність, який відповідає за передачу запитів до БД і обробку отриманих від неї відповідей. У загальному випадку, визначення Data Access Object описує його як прошарок між БД і системою.

В класі створюється сутність з полями Name і NameLast, а також методами Get і Set для передачі або отримання цих полів.



```
public class StaffFrom {
    private String Name;
    private String NameLast;

    public String getName() { return Name; }

    public void setName(String name) {
        Name = name;
    }

    public String getNameLast() {
        return NameLast;
    }

    public void setNameLast(String nameLast) {
        NameLast = nameLast;
    }
}
```

Рисунок 3.5 — Частина коду класу StaffFrom.

Staff.java — клас-сутність пов'язаний з БД за допомогою анотації @Entity або через XML.

Клас позначений анотацією @Entity, вона вказує Hibernate на те що даний клас-сутність. Клас має пустий конструктор. Анотацією @Id вказуємо на первинний ключ цього класу. @GeneratedValue (strategy = GenerationType.AUTO) робить генерацію ключей автоматизовано рисунок 3.6

```
@Entity
public class Staff {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private long id;
    private String Name;
    private String NameLast;

    public Staff() {

    }

    public Staff(String Name, String NameLast) {
        this.Name = Name;
        this.NameLast = NameLast;
    }

    public long getId() {
        return id;
    }

    public void setId(long id) {
        this.id = id;
    }

    public String getName() {
        return Name;
    }
```

Рисунок 3.6 — Частина коду класу Staff.

StaffRepository.java — клас-репозиторій, являє собою всі об'єкти певного типу у вигляді концептуальної безлічі. Його поведінка схоже на поведінку колекцій, за винятком більш розвинених можливостей для побудови запитів

Файли: AddStaff.html, index.html, ListStaff.html – являють собою веб-сторінку. Style.css — стиль який буде використаній на веб-сторінках.

Тестування проекту. Запуск проекту та інтерфейс PGAdmin зображено на рисунках 3.7 – 3.8.

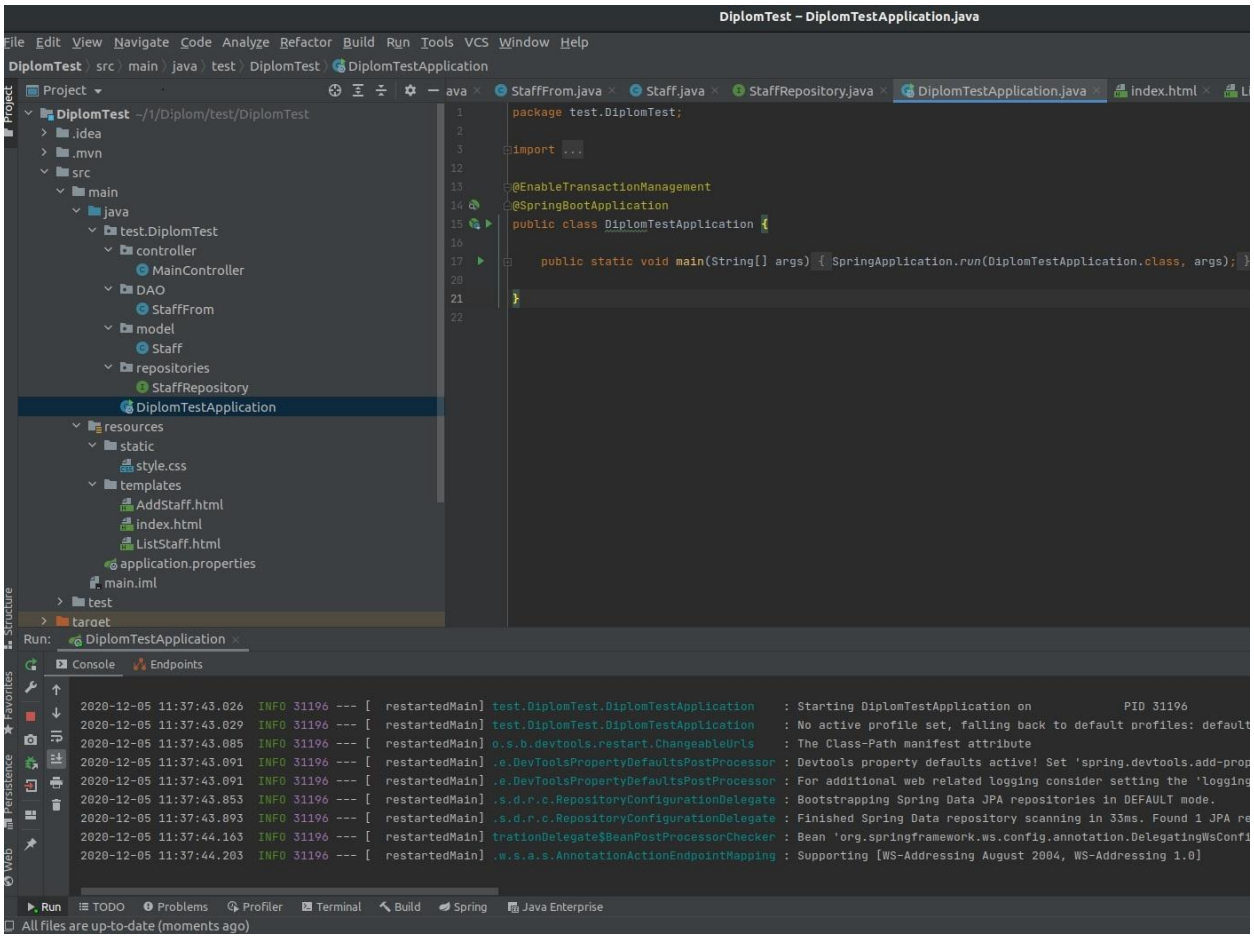


Рисунок 3.7 — Запуск проекту.

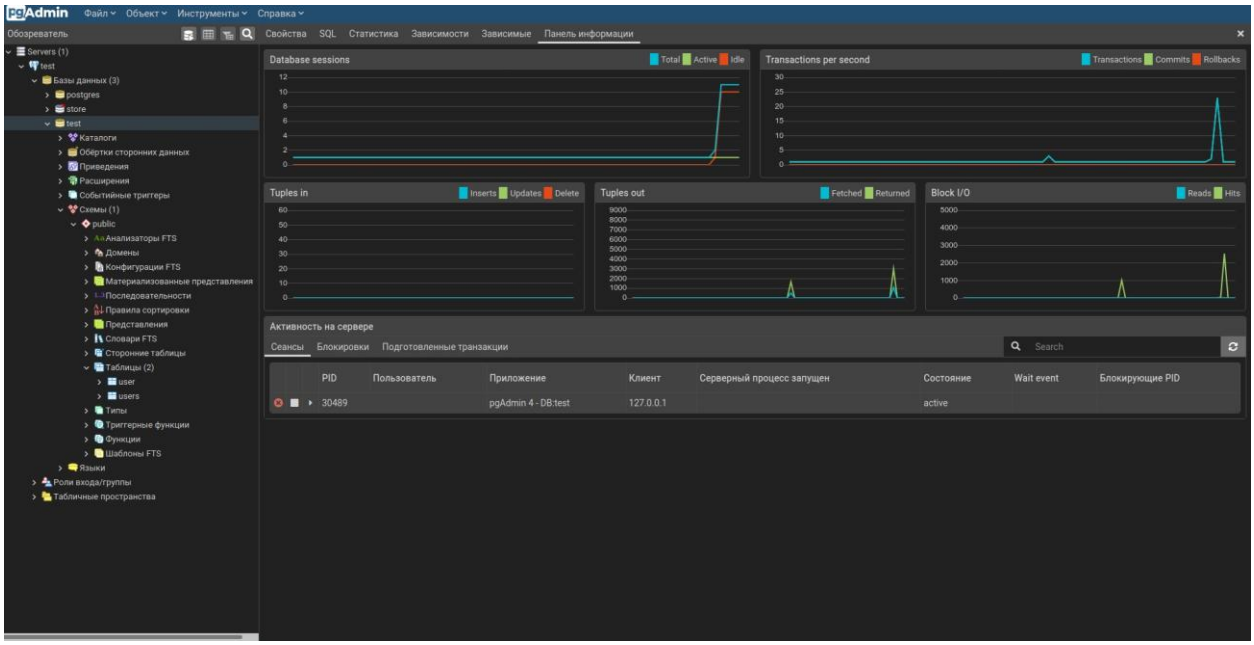


Рисунок 3.8 — Інтерфейс PGAdmin.

При створенні, база даних оновлюється, т.к задані такі умови в файлі application.properties. Статус підключення БД видно з графіку сесії.
При переході на сторінку localhast:8080 відображається сторінка , index.html, рисунок 3.9.Список персоналу відображається на сторінці ListStaff.html,так як БД не заповнена список пустий, рисунок 3.10.

При додаванні нової персони, йде перехід на сторінку AddStaff.html, рисунок 3.11. Після додавання нової персони відбувається перехід на сторінку ListStaff.html рисунок 3.12. В PGAdmin, з’являється нова таблиця Staff рисунок 3.13.

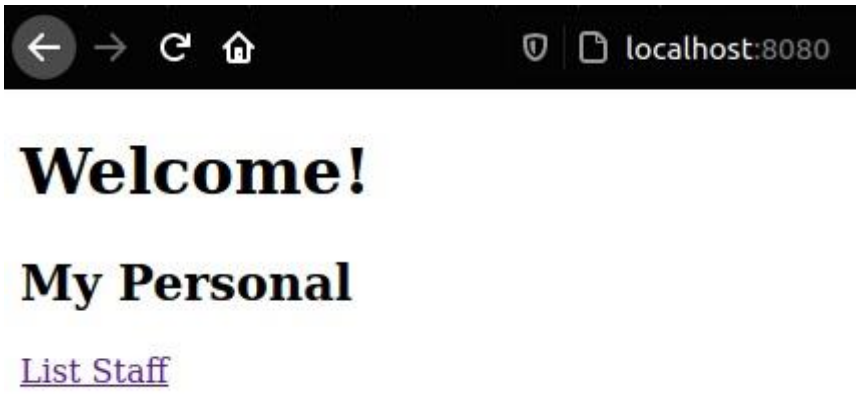


Рисунок 3.9 — Сторінка index.html.

Рисунок 3.10 — Сторінка ListStaff.html.

Рисунок 3.11 — Сторінка AddStaff.html.

Рисунок 3.12 — Сторінка ListStaff.html. з тестовою персоною.

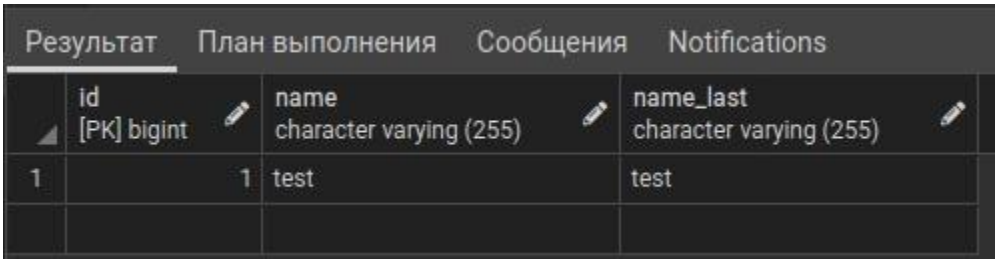


Рисунок 3.13 — Сторінка PGAdmin таблиця з тестовою персоною.

Для реалізація Android додатку був створений проект на основі WebView з офіційного джерела [20], та потім модернізований під задачу диплому. Древо проекту приведено на рисунку 3.14. Код проекту приведений в додатку Б.

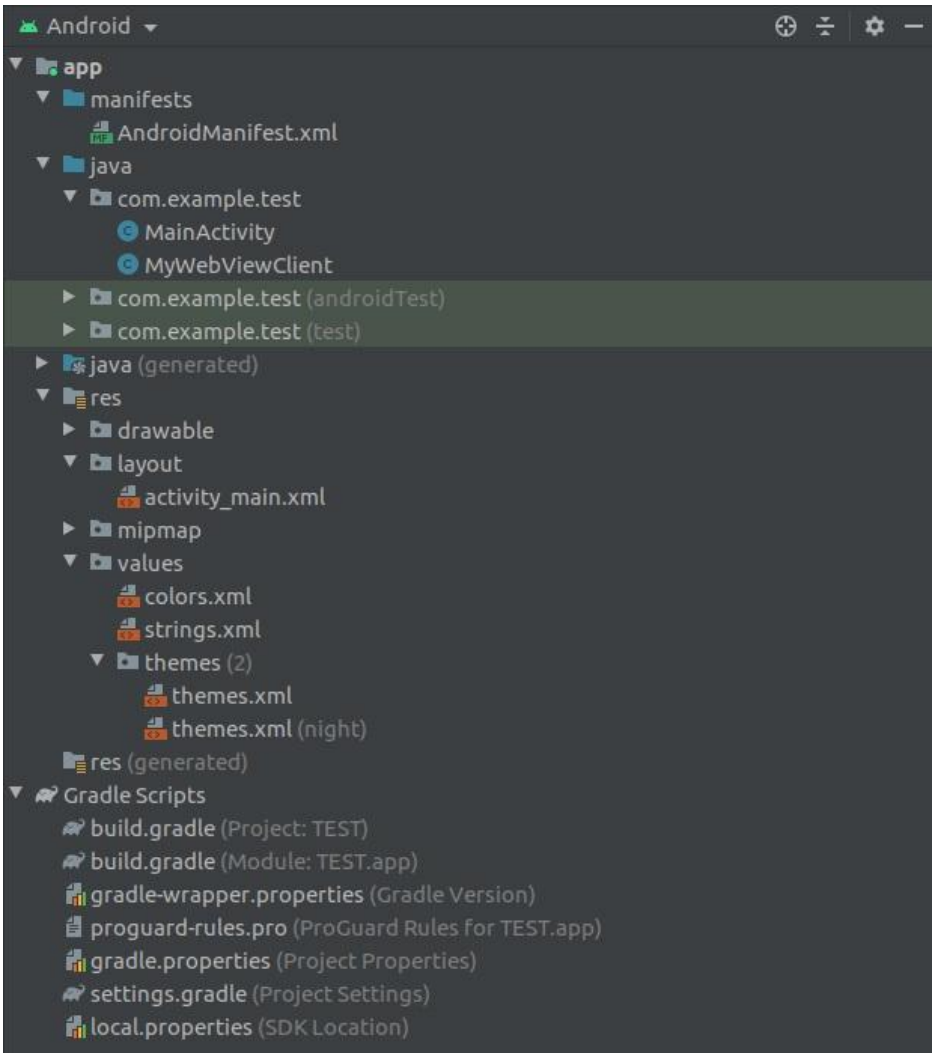


Рисунок 3.14 — Древо проекту android.

В проекті створений клас MyWebViewClient, це клас-екземпляр потрібний для обробки навігації WebView.

AndroidManifest.xml. – файл маніфесту інкапсулює всю архітектуру Android-додатку, його функціональні можливості і конфігурацію.

Клас MainActivity.java – клас за замовчуванням. Цей клас наслідує AppCompatActivity, це надає змогу отримати окрему сторінку. Метод

setContentView() передає ресурс графічного інтерфейсу. Анотацієя @Override перевизначає клас функціями з суперкласу до дочірнього. За допомогою методу loadUrl йде підключення до мережі моделі КІС.

```
public class MainActivity extends AppCompatActivity {


    private Object MyWebViewClient;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        WebView myWebView = (WebView) findViewById(R.id.webview);
        myWebView.loadUrl("http://192.168.0.103:8080");
        WebSettings webSettings = myWebView.getSettings();



        myWebView.setWebViewClient(new MyWebViewClient());


    }
```

Рисунок 3.15 — Клас MainActivity.

Android додаток тестувався за допомогою wifi-мережі. Так як тестування здійснювалось на ПК під управлінням системи Ubuntu 18.04 LTS, головною умовою працездатності додатку було в відключенні між мережевого екрану для доступу до мережі. Відключення між мережевого екрану здійснюється в терміналі системи командою sudo ufw disable.

Алгоритм тестування android додатку такий же як і на ПК. На рисунках 3.16-3.19 зображено працездатність додатку.

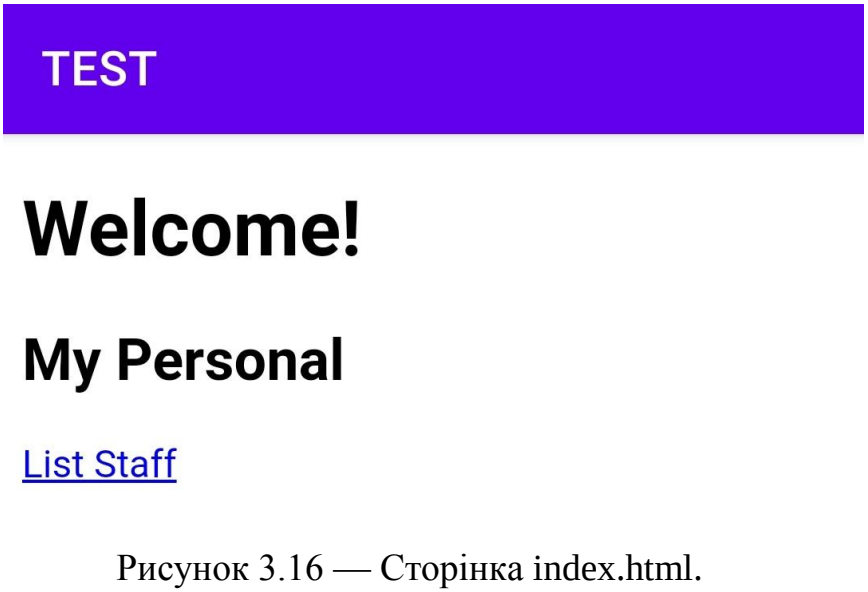


Рисунок 3.16 — Сторінка index.html.

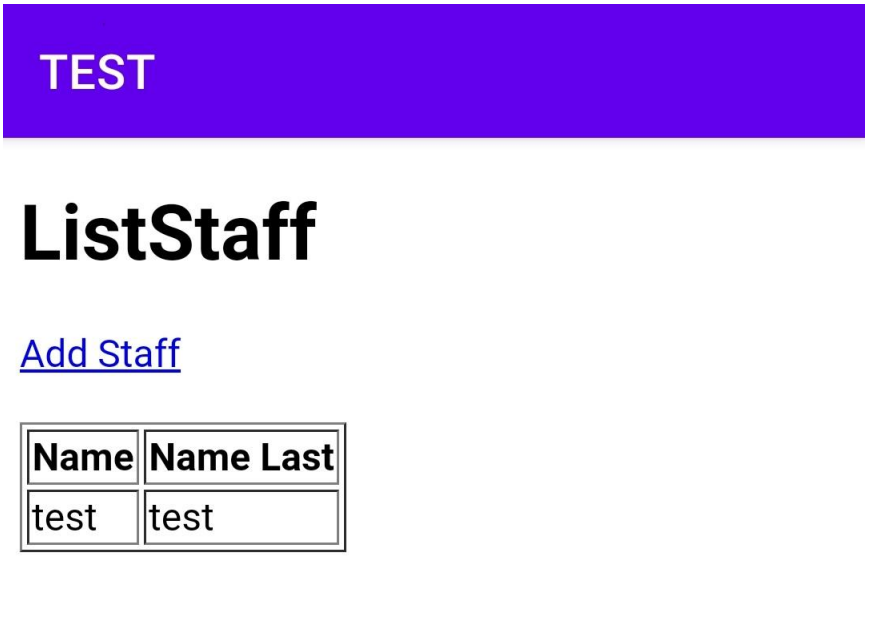


Рисунок 3.17 — Сторінка ListStaff.html.

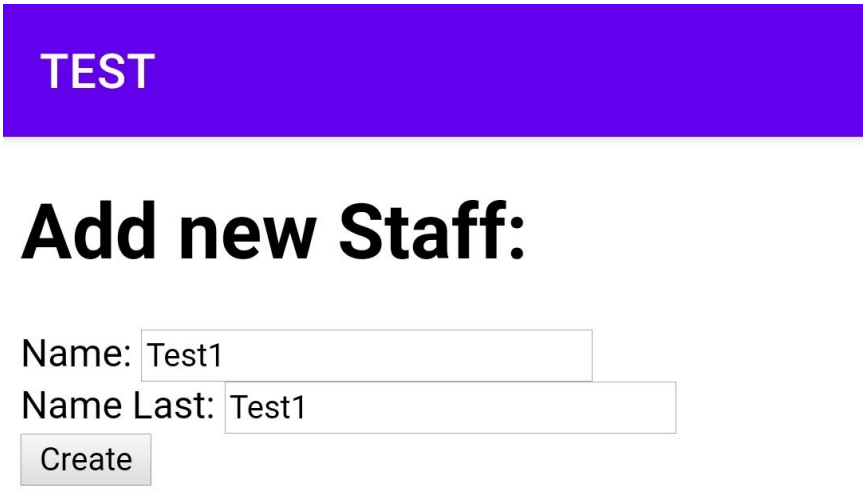


Рисунок 3.18 — Сторінка AddStaff.html

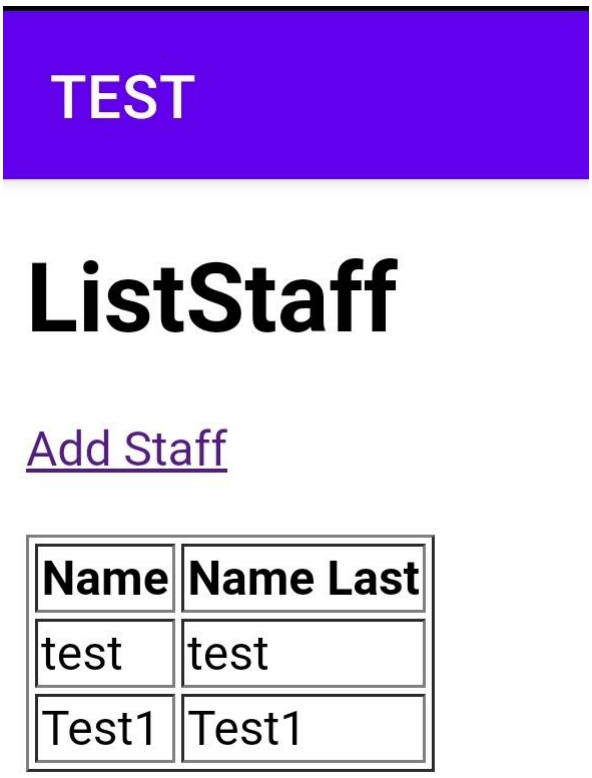


Рисунок 3.19 — Сторінка ListStaff.html. з тестовою персоною.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було розроблено модель корпоративної інформаційної системи, та додаток android. В роботі були виконані наступні етапи проекту:

аналіз проекту;

обрана модель побудови проекту;

створена модель корпоративної інформаційної системи за допомогою мови програмування Java;

розроблений додаток android;

створення доступу до КІС.

За результатами проекту було створено працездатну модель КІС та імплементований додаток android розроблений за допомогою компонента WebView. Основною перевагою такої імплементації до КІС є можливість доступу до неї завдяки android додатку який підключений до корпоративної wi-fi мережі без необхідності у використані персонального комп’ютера.

ПЕРЕЛІК ПОСИЛАНЬ

1. База знаний по бизнес-анализу URL: https://analytics.infozone.pro/korporativnye-informacionnye-sistemy-teoriya-ogranichenij/#i-19
2. Корпоративные информационные системы URL: http://iablov.narod.ru/igupit/kislec.htm
3. Стаття: " Що таке MRP, ERP, CSRP" URL: https://inteltech.com.ua/uk/blogs/shcho-take-mrp-erp-csrp
4. Файловый архив студентов StudFiles URL: https://studfile.net/preview/3284110/page:6/
5. Учебные материалы: Корпоративные информационные системы URL: https://works.doklad.ru/view/H7KRY9jnkPY.html
6. Л.И. Бушуева " Проблемы внедрения корпоративных информационных систем." Корпоративное управление и инновационное развитие экономики Севера. Вестник Научно-исследовательского центра корпоративного права, управления и венчурного инвестирования Сыктывкарского государственного университета URL: http://koet.syktsu.ru/vestnik/2005/2005-3/10.htm
7. Реферат: Корпоративні інформаційні системи КІС URL: https://www.bestreferat.ru/referat-375137.html
8. URL: https://ru.wikipedia.org/wiki/Java
9. URL: https://ru.wikipedia.org/wiki/Java_Development_Kit
10. URL: https://ru.wikipedia.org/wiki/Java_Virtual_Machine
11. URL: https://ru.wikipedia.org/wiki/IntelliJ_IDEA
12. URL: https://ru.wikipedia.org/wiki/Android_Studio
13. URL: https://ru.wikipedia.org/wiki/PostgreSQL

14. URL: https://ru.wikipedia.org/wiki/Spring_Framework
15. URL: https://ru.wikipedia.org/wiki/Java_Persistence_API
16. URL:https://ru.wikipedia.org/wiki/Hibernate_(%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0)
17. URL:https://ru.wikipedia.org/wiki/Apache_Maven
18. Руководство: Thymeleaf + Spring. Часть 1 URL: https://habr.com/ru/post/435062/
19. Руководство Spring boot и Thymeleaf URL: https://o7planning.org/ru/11545/spring-boot-and-thymeleaf-tutorial
20. Офіційний сайт Android URL: https://developer.android.com/guide/webapps.

Додаток А

Клас DiplomTestApplication:

```
package test.DiplomTest;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
org.springframework.transaction.annotation.EnableTransactionManagement
;

@EnableTransactionManagement
@SpringBootApplication
public class DiplomTestApplication {

	public static void main(String[] args) {
		SpringApplication.run(DiplomTestApplication.class, args);
	}

}
```

Клас MainController:

```
package test.DiplomTest.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import test.DiplomTest.DAO.StaffFrom;
import test.DiplomTest.model.Staff;
import test.DiplomTest.repositories.StaffRepository;


@Controller
public class MainController {

    private final StaffRepository staffRepository;

    @Autowired
    public MainController(StaffRepository staffRepository){
        this.staffRepository = staffRepository;
    }
```

```
    @Value("${welcome.message}")
    private String message;

    @Value("${error.message}")
    private String errorMessage;

    @RequestMapping(value = {"/", "/index"}, method =
RequestMethod.GET)
    public String index(Model model) {
        model.addAttribute("message", message);

        return "index";
    }

    @RequestMapping(value = {"/ListStaff"}, method =
RequestMethod.GET)
    public String ListStaff(Model model) {

        model.addAttribute("staffs", staffRepository.findAll());

        return "ListStaff";
    }
    @RequestMapping(value = {"/AddStaff"}, method = RequestMethod.GET)
    public String showAddStaffPage(Model model) {

        StaffFrom staffFrom = new StaffFrom();
        model.addAttribute("StaffFrom", staffFrom);

        return "AddStaff";
    }

    @RequestMapping(value = {"/AddStaff"}, method =
RequestMethod.POST)
    public String SaveStaff (Model model, //
                             @ModelAttribute("StaffRepository")
StaffFrom staffFrom) {

        String Name = staffFrom.getName();
        String NameLast = staffFrom.getNameLast();


           if (Name != null && Name.length() > 0 //
             && NameLast != null && NameLast.length() > 0) {
          Staff newStaff = new Staff(Name, NameLast);
          staffRepository.save(newStaff);

          return "redirect:/ListStaff";
      }

       model.addAttribute("errorMessage", errorMessage);
        return "AddStaff";
    }
```

```
}
```

Клас StaffFrom:

```
package test.DiplomTest.DAO;

public class StaffFrom {
    private String Name;
    private String NameLast;

    public String getName() {
        return Name;
    }

    public void setName(String name) {
        Name = name;
    }

    public String getNameLast() {
        return NameLast;
    }

    public void setNameLast(String nameLast) {
        NameLast = nameLast;
    }
}
```

Клас Staff:

```
package test.DiplomTest.model;

import javax.persistence.*;


@Entity
public class Staff {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private long id;
    private String Name;
    private String NameLast;

    public Staff() {

    }

    public Staff(String Name, String NameLast) {
        this.Name = Name;
        this.NameLast = NameLast;
    }

    public long getId() {
        return id;
```

```
    }

    public void setId(long id) {
        this.id = id;
    }

    public String getName() {
        return Name;
    }

    public void setName(String name) {
        Name = name;
    }

    public String getNameLast() {
        return NameLast;
    }

    public void setNameLast(String nameLast) {
        NameLast = nameLast;
    }

    @Override
    public String toString() {
        return "Staff{" +
                "id=" + id +
                ", Name='" + Name + '\'' +
                ", NameLast='" + NameLast + '\'' +
                '}';
    }
}
```

Клас StaffRepository:

```
package test.DiplomTest.repositories;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;
import test.DiplomTest.model.Staff;

@Repository
public interface StaffRepository extends CrudRepository<Staff, Long> {


}
```

Файл pom.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
- <parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.2.5.RELEASE</version>
  <relativePath />
  - <!--
   lookup parent from repository
   --> 
   </parent>
  <groupId>test</groupId>
  <artifactId>DiplomTest</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>DiplomTest</name>
  <description>Demo project for Spring Boot</description>
- <properties>
  <java.version>1.8</java.version>
    </properties>
- <dependencies>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
```

```
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
- <exclusions>
- <exclusion>
  <artifactId>hibernate-entitymanager</artifactId>
  <groupId>org.hibernate</groupId>
    </exclusion>
    </exclusions>
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-validation</artifactId>
    </dependency>
- <dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-core</artifactId>
  <version>5.4.12.Final</version>
    </dependency>
- <dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
    </dependency>
- <dependency>
  <groupId>org.postgresql</groupId>
  <artifactId>postgresql</artifactId>
  <scope>runtime</scope>
```

```
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-hateoas</artifactId>
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web-services</artifactId>
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-devtools</artifactId>
  <scope>runtime</scope>
  <optional>true</optional>
    </dependency>
- <dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
    </dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
- <exclusions>
- <exclusion>
  <groupId>org.junit.vintage</groupId>
  <artifactId>junit-vintage-engine</artifactId>
```

```
        </exclusion>
        </exclusions>
        </dependency>
        </dependencies>
- <build>
    <finalName>spring-boot-crud</finalName>
- <resources>
- <resource>
    <directory>src/main/resources</directory>
    <filtering>true</filtering>
        </resource>
        </resources>
- <plugins>
- <plugin>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
        </plugins>
        </build>
- <repositories>
- <repository>
    <id>spring-milestones</id>
    <name>Spring Milestones</name>
    <url>https://repo.spring.io/milestone</url>
        </repository>
- <repository>
    <id>spring-snapshots</id>
    <name>Spring Snapshots</name>
```

```
<url>https://repo.spring.io/snapshot</url>
- <snapshots>
<enabled>true</enabled>
</snapshots>
</repository>
</repositories>
- <pluginRepositories>
- <pluginRepository>
<id>spring-milestones</id>
<name>Spring Milestones</name>
<url>https://repo.spring.io/milestone</url>
</pluginRepository>
- <pluginRepository>
<id>spring-snapshots</id>
<name>Spring Snapshots</name>
<url>https://repo.spring.io/snapshot</url>
- <snapshots>
<enabled>true</enabled>
</snapshots>
</pluginRepository>
</pluginRepositories>
</project>
```

Додаток Б

Файл AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8" ?>
- <manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.test">
  <uses-permission android:name="android.permission.INTERNET" />
- <application android:allowBackup="true" android:usesCleartextTraffic="true"
    android:icon="@mipmap/ic_launcher" android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round" android:supportsRtl="true"
    android:theme="@style/Theme.TEST">
- <activity android:name=".MainActivity">
- <intent-filter>
  <action android:name="android.intent.action.MAIN" />
  <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    </activity>
    </application>
    </manifest>
```

Клас MainActivity.java

```
package com.example.test;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Context;
import android.os.Bundle;
import android.webkit.JavascriptInterface;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {


    private Object MyWebViewClient;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        WebView myWebView = (WebView) findViewById(R.id.webview);
        myWebView.loadUrl("http://192.168.0.103:8080");
        WebSettings webSettings = myWebView.getSettings();
        webSettings.setJavaScriptEnabled(true);



        myWebView.setWebViewClient(new MyWebViewClient());
    }


}
```

Клас MainActivity.java

```
package com.example.test;

import android.annotation.TargetApi;
import android.os.Build;
import android.webkit.WebResourceRequest;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MyWebViewClient extends WebViewClient {
    @TargetApi(Build.VERSION_CODES.N)
    @Override
    public boolean shouldOverrideUrlLoading(WebView view,
WebResourceRequest request) {
        view.loadUrl(request.getUrl().toString());
        return true;
    }
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url)
{
        view.loadUrl(url);
        return true;
    }


}
```

Файл activity_main.xml:

```
<?xml version="1.0" encoding="utf-8" ?>
- <androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent" android:layout_height="match_parent"
    tools:context=".MainActivity">

  <WebView android:id="@+id/webview" android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  </androidx.constraintlayout.widget.ConstraintLayout>
```