ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ ПОКРАШЕННЯ РОБОТИ СФЕРИ ОБСЛУГОВУВАННЯ ПІД ОПЕРАЦІЙНУ СИСТЕМУ ANDROID**»

Виконав: студент 4 курсу, групи ПД-41

спеціальності

121 Інженерія програмного забезпечення

Нікончук П.І.

Керівник Жебка В.В.

Рецензент _____

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність - 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії програмного забезпечення

_____О.В. Негоденко

“____”____20___року

ЗАВДАННЯ

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

НІКОНЧУК ПАВЛО ІГОРОВИЧ

1. Тема роботи: «Розробка мобільного додатку для покращення роботи сфери обслуговування під операційну систему Android»
Керівник роботи Жевка Вікторія Вікторівна, доцент кафедри, кандидат технічних наук, доцент
затверджені наказом вищого навчального закладу від «12» березня 2021 року

№65.

2. Строк подання студентом роботи «01» червня 2021 року

3. Вихідні дані до роботи:
Методи розробки Android додатків;

Офіційна документація мови програмування Dart.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

4.1 Аналіз об’єкту програмування;
4.2 Опис програмних засобів;
4.3 Опис загальних відомостей про додаток;
4.4 Реалізація додатку;

5. Перелік графічного матеріалу

1. Титульний слайд;
2. Мета, об’єкт дослідження, предмет дослідження;
3. Аналіз існуючих аналогів;
4. Технічне завдання;
5. Засоби програмної реалізації;
6. Діаграма діяльності;
7. Розроблений додаток;
8. UML діаграма послідовностей для архітектури MVC

6. Дата видачі завдання «19» червня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з /п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми бакалаврської роботи	19.04-20.04	Виконано
2	Дослідження актуальності теми	20.04-26.04	Виконано
3	Розгляд аналогічних додатків	26.04-30.04	Виконано
4	Дослідження програмних засобів	01.05-05.05	Виконано
5	Розробка функціоналу додатку	05.05-07.05	Виконано
6	Розробка додатку з використанням Flutter фреймворку	10.05-15.05	Виконано
8	Розробка презентації	15.05-16.05	Виконано
9	Захист дипломної роботи		

Студент Нікончук П.І.

Керівник роботи Жебка В.В.

РЕФЕРАТ

Текстова частина бакалаврської роботи с. 50, рис. 26, джерел 10.

Об`єкт сфери дослідження - Покращення роботи сфери обслуговування.

Предмет дослідження - Мобільний додаток під операційну систему android.

Мета дослідження - Покращення роботи сфери обслуговування за рахунок розробки мобільного додатку під операційну систему Android.

Методи дослідження - З такими додатками ми зустрічаємося майже кожен день. Деякі мають свої недоліки, але без всі заклади ще мають свої додатки.

У дипломній роботі були проаналізовано актуальні схожі додатки, їх плюси та мінуси. Роздивилися програмні засоби за допомогою які використовуються при розробці додатку. Також була виконана розробка власного додатку для сфери обслуговування.

Розроблений додаток розрахований на те щоб зменшити можливість гостям закладу створювати черги та економити час користувачів додаткі. Додаток має надасти зручний інтерфейс та швидкий онлайн розрахунок..

Ключові слова: Flutter, Dart, FireBase, BLOC, Android, Android Studio.

ЗМІСТ

ВСТУП

З розвитком технологій використання смартфонів у повсякденному житті стає дедалі поширенішим. В наш час майже у кожної людини є свій смартфон. Різні мобільні додатки дозвоняють значно спростити наше життя. Вони мають застосування майже у всіх областях нашого життя. Сфера обслуговування завжди була та є і буде. Тому поява додатків які дозволяють отримувати їжу, речі і тому подібне було лише питанням часу. Зараз вже є дуже багато додатків від закладів харчування бо не один підприемець не хоче втрачати кошти.

Так як ми живемо в час пандемії, додатки сфер обслуговування стають все більше і більше популярні. Багато з них працюють на доставку та на самовивіз. В основному на доставку працюють заклади харчування. Велика кількість людей не виходять з дому щоб замовляють їжу. Але не всі заклади мають свої мобільні додатки, в деяких є Web-сайт, але і це не завжди рятує. По-перше не всі сайти мають мобільну версію, по-друге не у всіш зроблена нормальна мобільна версія. Але якщо користуватися з персонального настільного комп’ютера то це дуже зручно. Також маю бажання сказати про доатки які розробили супермаркети. Вони не завжди практичні, супермаркетів з нормальними додатками можу на пальцях перерахувати, в усіх інших є багато недопрацювань. Багато людей просто розуміють як ними користуватися. В наш час дуже розвиваються технології тому я впевнений що в ближайший час усі подібні додатки будуть все краще та краще. Але додаток який буде розроблятися трохи не про це. Додаток має нахил на економію часу для людей які поспішають на роботу, зустріч, тощо. Багато людей полюбляють каву. Кожен 3-й житель спального району зранку біжить на роботу або їде на роботу автомобілем і хоче кави. Але їм немає часу чекати поки їм зроблять каву. Додаток має надати можливість зробити замовлення завчасно. Наприклад виходить людина з дому, спускається в ліфті, або прогрівае автомобіль, також можливо ви з дівчиною вийшли прогулятися по парку, ви також можете замовити каву по дорозі до кав’ярні, в цей момент

відкрили додаток, замовили каву собі, можливо собі та другу або подрузі. Розрахунок може бути онлайн банківською картою, або готівкою в закладі. Отже людина як раз проходячи або роїзджаючи повз кавьярню, кава вже готова та чекає на замовника.

Об`єкт сфери дослідження - Покращення роботи сфери обслуговування.

Предмет дослідження - Мобільний додаток під операційну систему android.

Мета дослідження - Покращення роботи сфери обслуговування за рахунок розробки мобільного додатку під операційну систему Android.

Методи дослідження - З такими додатками ми зустрічаємося майже кожен день. Деякі мають свої недоліки, але без всі заклади ще мають свої додатки.

Новизна проекту – Задача проекту покладена на прискорення роботи так званого “fastfood” швидкої їжі. Спробувати зменшити черги. Знайти та розробити простий інтерфейс, в якому буде тільки найважливіше. Інтерфейс повинен бути простим з яким розбереться не тільки дитина, а і ще людина яка не дуже ладить з смартфоном.

У дипломній роботі були проаналізовано актуальні схожі додатки, їх плюси та мінуси. Роздивилися програмні засоби які використовували при розробці додатку. Також була виконана розробка власного додатку для сфери обслуговування.

Розроблений додаток розрахований на те щоб гості закладу не створювали велику чергу та економити час, але тільки для користувачів додатку. Додаток має надати зручний інтерфейс та швидкий онлайн розрахунок..

1 ТЕОРЕТИЧНА ЧАСТИНА

Останнім часом розвиток смартфонів набуває все більше обертів. С кожним днем операційні систеи стають досконалішими,мають більше функцій та розширюють можливості для реалізації додатків з використанням найновіших інструментів. Перші мобільні додатки почали з’являтися з операційними системами для мобільних пристроїв. Можливостей та функцій у перших додатків було досить мало. Але з кожною новою версією мобільної операційної системи зростала кількість інструментів для розробника. Це давало можливість захоплювати все більше областей для застосування створених додатків, а інструментарій став набагато більшим. Також сприяло створення додатків для використання у різних областях нашого життя.

На сьогоднішній день є багато категорій додатків, наприклад: відео плеєри, редактори, навігатори, онлайн магазини, додатки соціальних мереж. Вони використовуються повсюду. Майже кожен банк має власний клієнтський додаток, що значно спрощує взаємодію банка і клієнта. Є навіть банки які перейши в всесвітню мережу. В таких банках додаток стае не допоміжним, а основним інструментом взаемо дії кліента та банка. Ще одна область яка яка повністю перейша в онлайн та мобільні додатки, це пасажирські перевезення. Більшість служб таксі тають додатки через які клієнт може замовити автомобіль, спостерігати за маршрутом, за напрямком руху водія, а також оцінити сервіс. Також додатки мають великі мережі супермаркетів онлайн. Ці замовлення доставляють кур’єрські служби, які в свою чергу теж мають власний додаток. Область застосування одатків дуже велика та стає все більше і більше з кожним днем.

1.1 Мобільний додаток для покращення сфери обслуговування

1.1.1 Про додаток

Додаток матиме назву Eagle_Coffee. Стоїть задача надати можливість робити предзамовлення в кавьярні. Розробка буде під онду кавьярню, в тестовій формі.В додатку маємо намір зробити: Профіль для кожного користувача, меню з вибором товару, кошик для вибраного товару, по можливості додати персональні настройки по типу заднього фону. Додаток буде розробленый на Android-пристрої. Але розробка буде на фреймворку Flutter(детальніше сторінка 20.). Якщо коротко фреймворк дозволяє розробити аналогічний додаток для системи iOS. В намірах протестувати цей додаток у закладі (кавьярні), тільки після успішного тестування можливо будете почати розробку для iOS, а також зв’явиться можливість виводити додаток в маси. Перед тим як додаток побачить світ буде змінена назва на більш універсальну, та буде розроблена частина в якій буде вибір закладу. Доречі, не можна забувати про те що всі заклади різні, мова йде про:

1. Меню.
2. Ціни.
3. Знижки.
4. Оформлення.

Тому буде потреба розробити можливість змінівати все перечислене вище для кожного закладу індивідуально. Тоб-то щоб заклад який починає користуватися додатком, міг без труднощів змінювати вище перечислені пункти. А також буде потрібно зробити бота (так звана технічна підтримка) який буде допомагати розібратися з керуванням додатком.

1.1.2 Загальні функції

Додаток надає функціональність і простіть в UI

1. Профіль для кожного користувача - Профіль в якому є можливіть привьязати банківську карту, та продивитися минулі закази.

2. Детальний вибір товару - в виборі товару буде опис товару, калорії, протеіни і тому подібне.

3. Кошик для вибраного товару - як тільки виберете товар, вам будуть одразу відкрите вікно з добавками. Після вибору усіх добавок товар буде відправлений в кошик. В кошику під час оплати буде два варіан розрахунку готівковий, та без готівковий. Далі тільки забрати свій заказ в закладі.

4. Онлайн розрахунок - Моливість розрахуватися банківською картою.

1.1.3 Аналіз аналогів

Додаток “Сільпо”

Додаток супермаркету “Сільпо”

При відкритті додатку одразу попадаємо на головну сторінку.

В нижній частині дадатку можемо побачити такі сторінки:

“Головна сторінка” - як вище було сказано одразу відкривається при вході в додаток. На головній сторінці велика кількіть інформації, такої як бонуси та бали клієнта, персональні пропозиції, знижки в магазині та інше.

“Доставка” - одразу запитує адрес на який доставляти. Потім вибір найближчого магазину. Після цого вже можна набивати свій кошик. Але не всі товари будуть в наявності. Хоча якщо зайти одразу до магазину товар буде на полицях. Доставка товари коштує 40грн. Самовивіз я не знайшов.

“QR код” - використовується для сканування карти супермаркету на касі.

“Покупки” - арфів в якому можна знайти чеки з минулих покупок.

“Меню” - в меню знаходиться профіль клієнта.

Додаток зроблений в світлих тонах, приємний для ока, але дуже кидається в очі оранжевий колір.

Плюси:

1. Не треба ходити до супермаркету.

2. Можливіть не носити з собою карту супермаркету.

3. Дізнаватися про акції на товар через додаток.

Мінуси:

1. Немая самовивозу товару.

2. Забагато інформації на головній сторінці.

Додаток “Glovo”

Це додаток курьерської служби.

Коротко про нього простий додаток на головній сторінці немає нічого непотрібного. Тільки саме основне із того що курьери доставляють та архів заказів. Доставка недорога. Корисно для тих хто лінивий. Невиходячи с дому все доставляють. Додаток зроблений в жовтих тонах, дуже кидається в око.

Плюси:

1. Просто в користуванні.

2. Має банківський або готівковий розрахунок.

Мінуси:

1. Довго доставляють.

2. Не весь заказ може буду доставлений.

Додаток “Domino’s Pizza”

Додаток піцерії “Домінос Піцца”

При в ході в додаток одразу можемо знайти Акції та меню товарів. Є можливість відкрити мапу та подивитися усі піцерії в місті. Також архів заказів і меню додатку. Додаток зроблений в світлих тонах, приємний для ока. Додаток досить простий в користуванні.

Плюси:

1. Простий в користуванні.

2. Є розрахунок онлайн банківською картою.

3. Є можливість самовивозу.

Мінуси:

1. Довга доставка.

1.2 Архітектура

Розробники iOS та Android добре розбираються в Model-View-Controller (MVC) і використовують цей шаблон як вибір за замовчуванням при створенні програми. Модель і Вид розділені, контролер надсилає між ними сигнали.

Флаттер, однак, приносить новий реактивний стиль, який не повністю сумісний з MVC. Варіант цього класичного візерунка з'явився у спільноті Флаттера під назвою BLoC pattern.

1.2.1 BLOC

Це штатна система управління Flutter, рекомендована розробниками Google. Це допомагає керувати станом та робити доступ до даних із центрального місця у вашому проекті.

Бібліотека управління станом, яка допомагає реалізувати шаблон дизайну BLoC таблиця 1.1.

Таблиця 1.1

Package	Pub
bloc	pub v7.0.0
bloc_test	pub v8.0.0
flutter_bloc	pub v7.0.0
angular_bloc	pub v6.0.1
hydrated_bloc	pub v7.0.0
replay_bloc	pub v0.0.1

Наведена діаграма рисунок 1.1

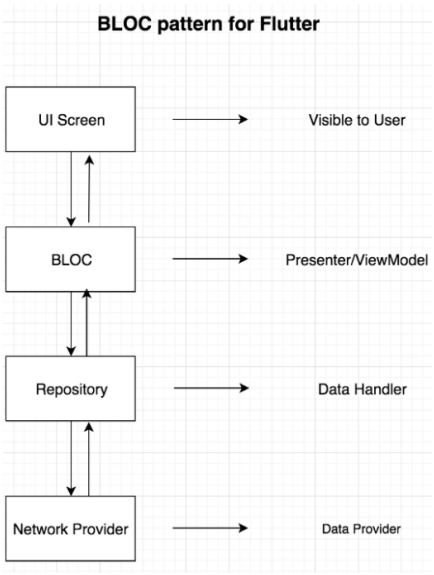


Рисунок 1.1 - Схема Архітектури BLOC

Ця діаграма показує, як перетікають дані від інтерфейсу користувача до рівня даних і навпаки. BLOC ніколи не матиме посилань на віджети на екрані інтерфейсу користувача. На екрані інтерфейсу спостерігатимуться лише зміни, що надходять із класу BLOC.

Що знаходиться під капотом BLOC?

Стрімкий або реактивний підхід. Загалом, дані будуть надходити від BLOC до UI або від UI до BLOC у формі потоків.

Чи можна співвіднести цю архітектуру з будь-якою іншою архітектурою?

Так, звісно. MVP та MVVM - кілька хороших прикладів. Зміниться лише те: BLOC буде замінено на ViewModel у MVVM.

Почнемо будувани проект із шаблоном BLOC

Спочатку створіть новий проект і очистіть весь код у файлі main.dart. Введіть нижче команду у своєму терміналі:

```
flutter create myProjectName
```

1. Запишіть код нижче у вашому файлі main.dart:

```
    import 'package:flutter/material.dart';
import 'src/app.dart';
void main(){
runApp(App());
}
```

Напевно, у другому рядку з'являється помилка. Ми вирішимо це на наступних етапах.

2. Створіть пакет **src** під пакетом **lib** . Всередині пакета src створіть файл і назвіть його як **app.dart** . Скопіюйте, вставте наведений нижче код у файл app.dart.

```
    import 'package:flutter/material.dart';
import 'ui/movie_list.dart';
class App extends StatelessWidget {
@override
Widget build(BuildContext context) {
// TODO: implement build
return MaterialApp(
theme: ThemeData.dark(),
home: Scaffold(
body: MovieList(),),);}}
```

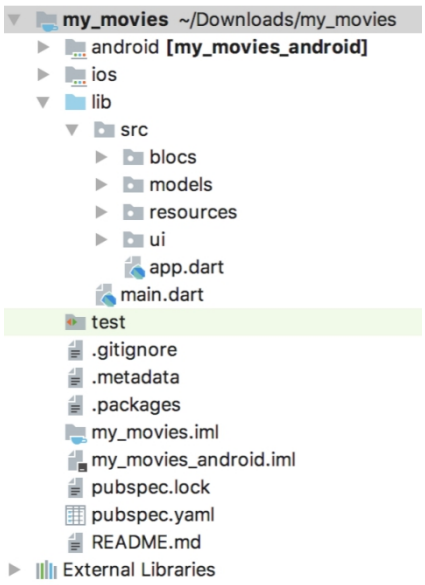


Рисунок 1.2 - Структура проекту.

3. Створіть новий пакет всередині пакета src та назвіть його **ресурсом.**

Тепер створіть кілька нових пакетів, тобто блоки, моделі, ресурси та інтерфейс, як показано на діаграмі вище рисукок 1.2, і тоді ми налаштовуємося на скелет проекту.

Blocs зберігатиме наші файли, пов'язані з реалізацією BLOC. Пакет **models** буде містити клас POJO або клас моделі відповіді JSON, який ми отримаємо від сервера. Пакет **resources** буде містити клас сховища та клас реалізованого мережевого виклику. пакет **ui** буде містити наші екрани, які будуть видимі користувачеві.

1.2.2 Model View Controller

Model-View-Controller рисунок 1.3 (MVC) являється архітектурним шаблоном, який відокремлює додаток на три основні логічні компоненти: модель , вид і контролер. Кожен із цих компонентів створений для обробки конкретних аспектів розробки програми. MVC – одна з найбільш часто використовуваних галузевих стандартів Flutter-розробки для створення масштабованих та розширюваних проектів.

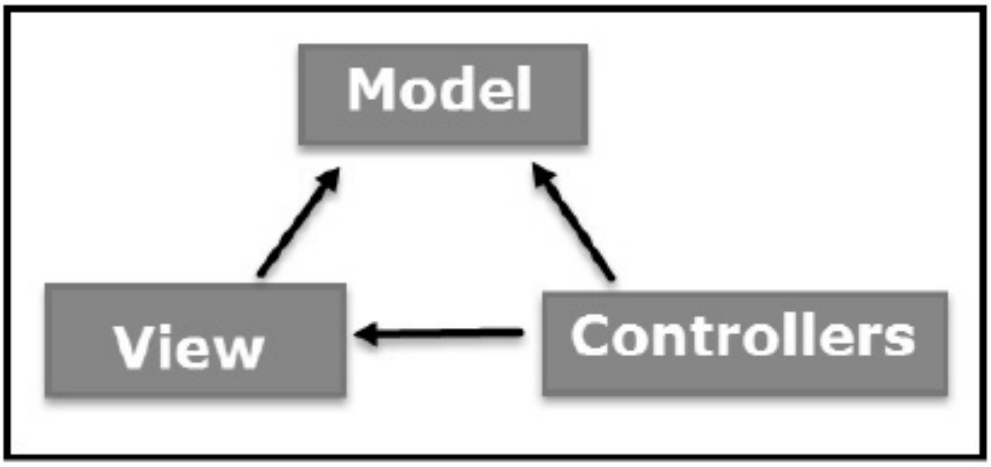


Рисунок 1.3 - Компоненти MVC

Модель

Компонент Model відповідає всій логікі даних, з якою працює користувач. Це може представляти або дані, що передаються між компонентами View та Controller, або будь-які інші дані, пов’язані з бізнес-логікою. Наприклад, об'єкт "Клієнт" буде отримувати інформацію про клієнта з бази даних, маніпулювати нею та оновлювати дані назад у базу даних або використовувати її для рендерингу даних.

Вид

Компонент View використовується для всієї логіки інтерфейсу програми. Наприклад, подання Клієнт включатиме всі компоненти інтерфейсу, такі як текстові поля, випадаючі меню тощо, з якими взаємодіє кінцевий користувач.

Контролер

Контролери виступають інтерфейсом між компонентами Model і View для обробки всієї бізнес-логіки та вхідних запитів, маніпулювання даними за допомогою компонента Model та взаємодії з поданнями для надання кінцевого результату. Наприклад, контролер замовника буде обробляти всі взаємодії та входи з перегляду замовника та оновлювати базу даних за допомогою моделі замовника. Той самий контролер буде використовуватися для перегляду даних Клієнта.

Flutter, однак, приносить новий реактивний стиль, який в повному обсязі сумісний з MVC. Варіація цього класичного патерну виникла із товариства Flutter, званого BLoC патерном.

2 ОПИС ПРОГРАМНИХ ЗАСОБІВ

Програмний продукт повинен бути побудований на мобільній платформі “Android”для використання на мобільних пристроях, що працюють на цій системі. В якості мови програмування було обрано мову програмування “Dart”. Для створення додатку було обрано інтегроване середовище розробки”Android Studio”.

2.1 Оцінка вимог

Для вирішення задачі необхідні наступні інструменти:

1. Персональний комп’ютер
2. Середовище розробки Android Studio
3. Мобільний пристрій на платформі Android

2.2 Платформа Android

Платформа “Android” є продуктом групи “Open Handset Alliance”, яка ставить перед собою ціль створити найбільш досконалу мобільну систему. З точки зору розробки програмного забезпечення “Android” знаходиться в самому центрі розробки відкритого програмного забезпечення. Перший пристрій який був випущений під управлінням саме цієї системи випустили у 2008 році і єдинй інструмен розробки додатків для нього були випуски пакета розробки “SDK”.

За шириною можливостей платформа “Android” не поступається місцем операційним системам персонатьних ком’ютерів. Це система яка має багато рівнів, та за основу тримає ядро “Linux” з широкими функціональними можливостями. В підсистему інтерфейсу користувача входять вікна, віджети та представлення. “Android” володіє широким спектром можливих підключень, такі як “Wi-Fi”, “Bluetooth” та протоколи передачі даних. В стек програмнрого

забезпечення “Android” входить підтримка сервісів, визначення місцеположення та аклелерометрів, але не всі пристрої на цій платформі мають необхідне обладнання. Також є явна підтримка фото та відеокамери. Історично мобільні системи відставали від систем в настільних персональних коп’ютерах двома областями: графікою та методом збереження даних. “Android” вирішив проблему графіки завдяки вбудованій підтримці графіки, включивши бібліотеку “OpenGl”. А задача збереження даних спростилася завдяки пополярної бази даних з відкритим кодом “Sqlite”.

Мобільна операційна система “Android” працює поверх ядра “Linux”. Додатки пишуться на мові програмування “Java” і виконуються в віртуальній машині. Але технології розвиваються та мов програмування для смартфонів з системою “Android” з’являється більше. Зараз можна написати додаток на мовах: Java, Dart, C#, C, C++, Kotlin, Python, та багато інших.

Компанія Google вклала багато ресурсів в зменшення платформи та в зростання її ефективності. Оптимізація в першу чергу направлена на зменшення розміру, збільшення швидкості роботи, економію заряду акамулятору та зменшення витрати пям’яті. Робота була проведена на делількох рівнях стеку. Віртуальна машина Dalvik була ретельно розроблена з метою задоволення вимог до продуктивності та має декілька незвичайних характеристик. На відміну від звичайного середовища Java, Dalvik заснований на регістрі, а не на стеку і не підтримує компіляцію JIT. Кожен додаток виконується в окремій віртуальній машині і пам’ять розподіляється між екземплярами для зменшення витрат. Для скорочення часу відкриття додатку, Dalvik має компонент який має назву Zygote, який ініціалізує екземляри віртуальних машин і розгалуджує їх, якщо в цьому є потреба.

Радикально інший підхід Android до розробки мобільних додатків, пропонує деякі унікальні переваги, але він також створює багато проблем для сторонніх розробників. Найбільшою перевагою є те що такий підхід забезпечує

високий рівень однорідності. Більша частина додатків Android можуть працювати на буль-якому пристрої на базі Android.

До складу операційної системи “Android” входить комплект базових додатків: клієнти електронної пошти, календар, карти, браузер, програми для керування контактами, тощо.

“Android” спокійно може використовувати свою потужність, яка використовується в додатках ядра. Архітектура побудована так, що буль який додаток може використовувати вже релізовані можливості іншого додатку, при умові, що той відкриє доступ до своєї функціональності. Так архітектура реалізує принцип багаторазового використання компонентів і додатків. На рисунок 2.1 наведено архітектуру системи.

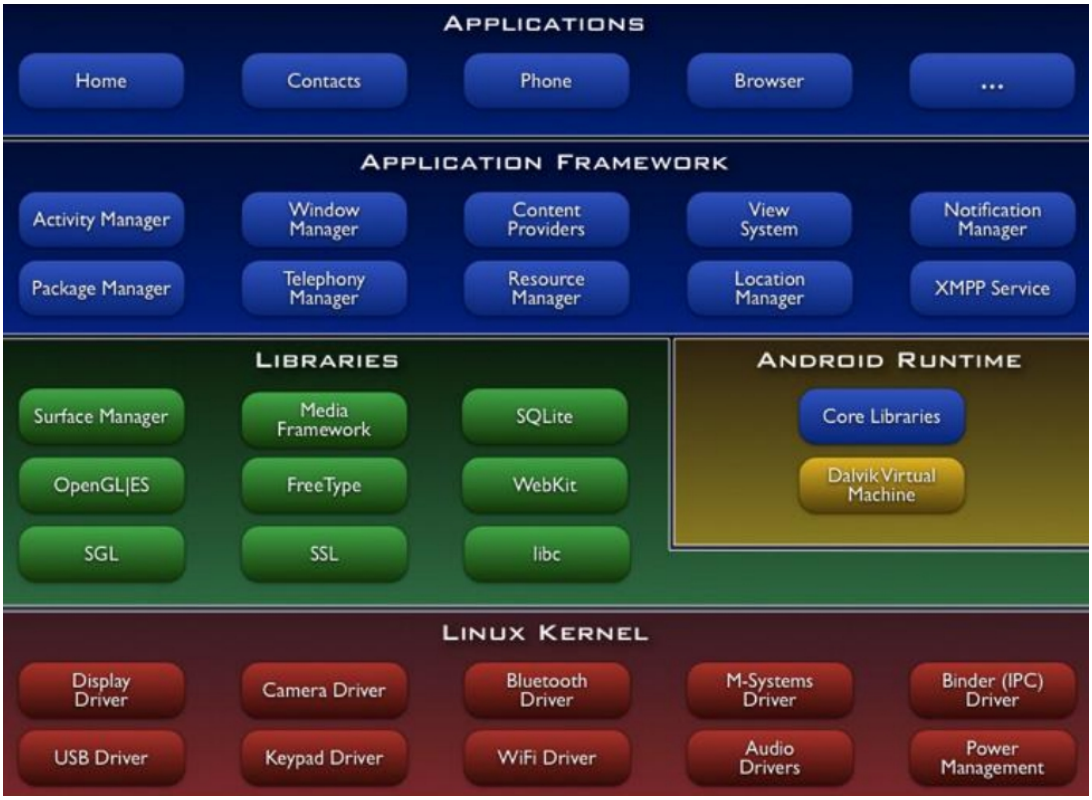


Рисунок. 2.1 - Схема архітектури системи.

Платформа Android надає добре організований асортимент високорівневих API інтерфейсів для створення додатків та використання функцій платформи.

Інтерфейси API забезпечують вісиким ріврнем абстракції, що робить іх відносно інтуїтивно зрозумілими і простими у використанні при процесі розробки додатків.

Сторонні додатки можуть працювати або взаємодіяти практично з усіма основними компонентами платформами. Наприклад: Android надає методи для отримання інформації з списку контактів користувача та методи для розширення системи контактів новими полями даних. Також нові інтерфейси дзвінка або реалізувати користкувальку поведінку для системи подій, таких як вхідні повідомлення. Зокрема, API дозволяють створювати додатки, які повнісю інтегруються з іншими платформами рисунок 2.2.

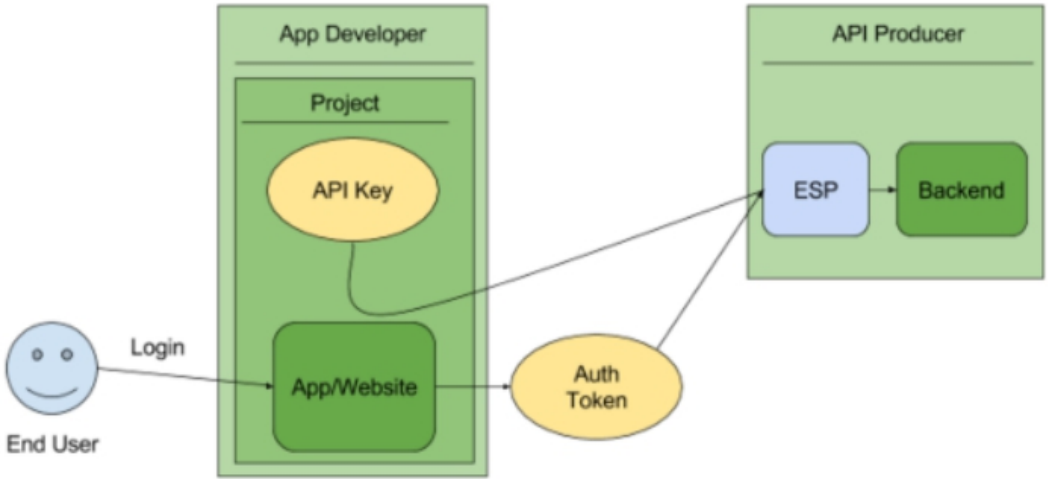


Рисунок 2.2 - Приклад використання API

Android має набір інструментів для віджетій, який дає велику кількість дуже корисних компонентів відразу. Окремі віджети розроблені спеціально для зручної взаємодії з пальцями. Для опису користувацького інтерфейсу на основу беруть мову XML для визначення макетів та атрибутів, при цьому описи завантажують в програму через систему ресурсів Android.

На окремі віджети в маркеті XML, можна посилатися по індетифікатору в програмі. Найбільш правильний спосіб створення макетів є написання описів

вручну. Пакет Android SDK має вбудований інструмент візуального макета, але він не завжди праціє як потрібно.

Додатки Android складені з провайдерів, приймачів, служб та активностей. Всі компоненти мають окрему задачу в стеку додатків Android. Спосіб, яким вони взаємодіють один з одним це те, що наповнює Android його модульністю.

Провайдери контенту слугують рівнем абстракції для взаємодії з джерелами даних і обміном постійними даними між додатками. Вони надають потрібну інформацію через стандартизований інтерфейс запитів. Запити описуються за допомогою синтаксису URI, розробники додатків використовують багаторівневі класи, які генерують редки запитів і керують ними. Какож можна підключити механізми моніторингу до провайдерів контенту, для того щоб ваш додаток міг отримати повідомлення про зміну даних.

Система провайдера контенту пропонує розробникам Android додатків декілька переваг. Найбільш явна переваго є те, що система забезпечує високий рівень ваємодії, дозволяє додаткам обмінюватися даними уніфікованим методом. Ключові джерела даних платформи включають список контактів та систему збереження мультимедіа, їх легко використовувати для сторонніх додатків тому що за замовчуванням вотни доступні через інтерфейси провайдерів контенту.

Важливою особливістю, яка відрізняє Android від інших платформ є те, що Android офіційно підтримує фонові процеси сторонніх додатків. Вони реалізовані через сервісний компонент. Сервіси - це операції без заголовку, які виконують в фоновому режимі протягом тривалого часу.

Система повідомлень Android дуже схожа з системою повідомлень Linux. Розробники вбудовують приймачі, які можуть визначит, коли з’являються визначені системні повідомлення чи події. Багато базових системних подій відслідковується за допомогою систем мовлення, тому відстежуються такі речі як: зміна стану акумулятора, отримання вхідних SMS.

Система Android Intents є ключем до того, як всі ці частини використовуються разом. Об’єкти Intents, які містять ідентифікатор дії та URI даних, використовуються для виклику дії, служб та отримувачів, ідентифікатор дії вказує на поведінку, а необов’язковий URI даних дає місце розташування цих даних, над якими має виконатися дія рисунок 2.3

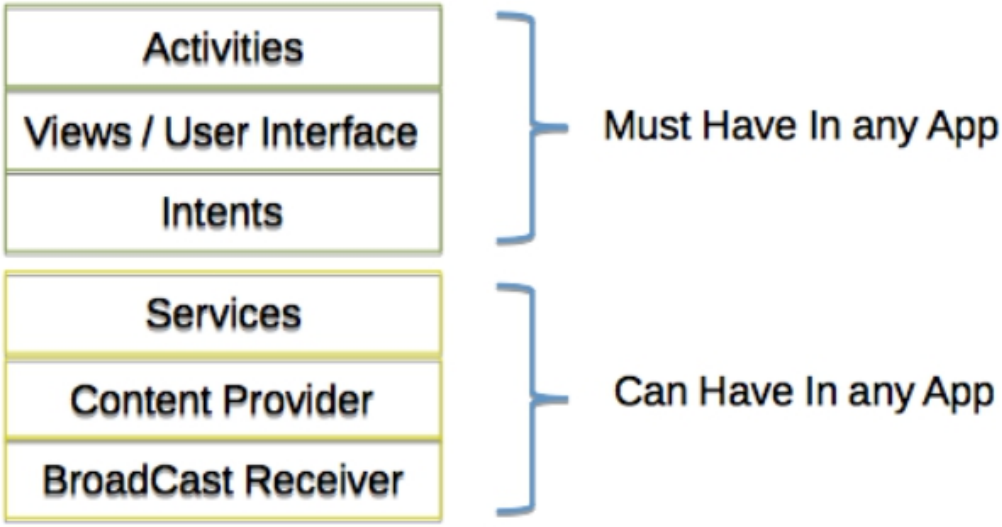


Рисунок 2.3 - Приклад використання API

Одна із головних переваг систем Android - її відкритість. Операційна система побудована на основі відкритого коду. Це дозволяє розробникам отримати доступ до вмсхідного коду і зрозуміти яким чином реалізовані функції додатків. Кожен користувач може прийняти участь в удосконаленні операційної системи.

2.3 Середовище розробки Android Studio

Загальна інформація

Середовище Android Studio - це офіційне інтегроване середовище розробки (IDE) для розробки додатків для Android, засноване на IntelliJ IDEA . На додаток до потужного редактора коду та інструментів розробника IntelliJ,

Android Studio пропонує ще більше функцій, що підвищують вашу продуктивність при створенні програм для Android, таких як:

1. Гнучка система побудови на основі Gradle
2. Швидкий і багатофункціональний емулятор
3. Єдине середовище, де ви можете розробляти всі пристрої Android
4. Застосовуйте зміни, щоб надсилати зміни коду та ресурсів до запущеної програми без перезапуску програми
5. Шаблони коду та інтеграція GitHub, щоб допомогти вам створити загальні функції програми та імпортувати зразок коду
6. Широкі інструменти та основи тестування
7. Інструменти Lint для виявлення продуктивності, зручності використання, сумісності версій та інших проблем
8. Підтримка C ++ та NDK
9. Вбудована підтримка Google Cloud Platform , що спрощує інтеграцію Google Cloud Messaging та App Engine

Вперше Android Studio було анонсовано на Google I/O в травні 2013 року, а перша стабільна збірка була випущена в грудні 2014 року. Середовище доступне для настільних платформ: Windows, Mac та Linux.

Android Studio являється часто використовуємою для Flutter-розробки, котра орієнтована на Flutter-додатки та інші види програм, які можна створювати за допомогою мови Dart. Android Studio надає функції автоматичного завершення конструкцій мови Dart в коді, інспектування коду, різні алгоритми рефакторинг та зручну і швидку навігацію по коду.

Реалізований в Android Studio графічний Dart-відладчик підтримує умовні точки зупину, відстеження значень і автоматизований вхід в налагодження окремих процедур. При редагуванні коду виділяються конструкції синтаксису, здійснюється розширене форматування конфігурації, виявлення помилок в режимі реального часу і завершення коду. Android Studio-редактор враховує коментарі до коду при його завершенні, автоматично вибираючи оптимальне

рішення проблеми. Dart-рефакторинг і редагування шаблонів гарантує зміна проекту в найкоротші терміни. Android Studio дозволяє візуалізувати код в ієрархічності вигляді і забезпечує швидку навігацію по всіх елементах.

Завдяки застосуванню Dart Unit-тестів можна швидко переглядати результати генерації коду окремих блоків або всього програми. Якщо тест був проведений невдало, продукт дозволяє переглядати окремі кодові рядки, в яких балу виявлена помилка. Android Studio забезпечує налагодження коду Dart і надає широкий діапазон можливостей: знаходження точки зупину, настройка параметрів точки зупину, тестування синтаксису коду в режимі реального часу і т. д.

В якості середевища для розробки було обрано “Android Studio”. Це інтегроване середовище розробки виробництва компанії “Java”, за допомогою якого розробники мають доступ для сворення додатків. Середовище доступно для різних операційних систем. “Android Studio” безкоштовне середовище, тобто завантажити може любий. В середовищі присутні макети для створення користувацького інтерфейсу. Також є інструменти для створення додатків на смартфон, сланшет, наручні смарт годинники та автомобілі. Рисунок 2.4

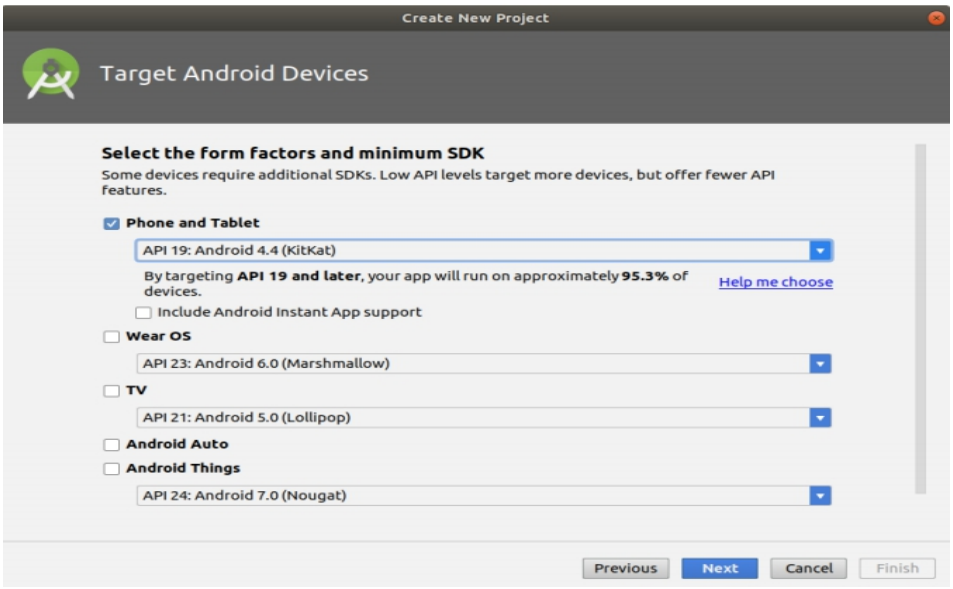


Рисунок 2.4 - Екран вибору форм фактору в Android Studio

Для початку розробки в Android Studio спочатку необхідно встановини безкоштовний пакет розробника JDK (Java Development Kit). Після встановлення та налаштування необхідно завантажити Android Studio з офіційного сайту. Длі необхідно запустити завантажений інсталятор та запустити процес встановлення. Коли відкриємо встановлений Android Studio побачимо налаштування. Потрібно налаштувати відповідно до вимог користувача. Після виконання усіх дій можна починати до створення додатку

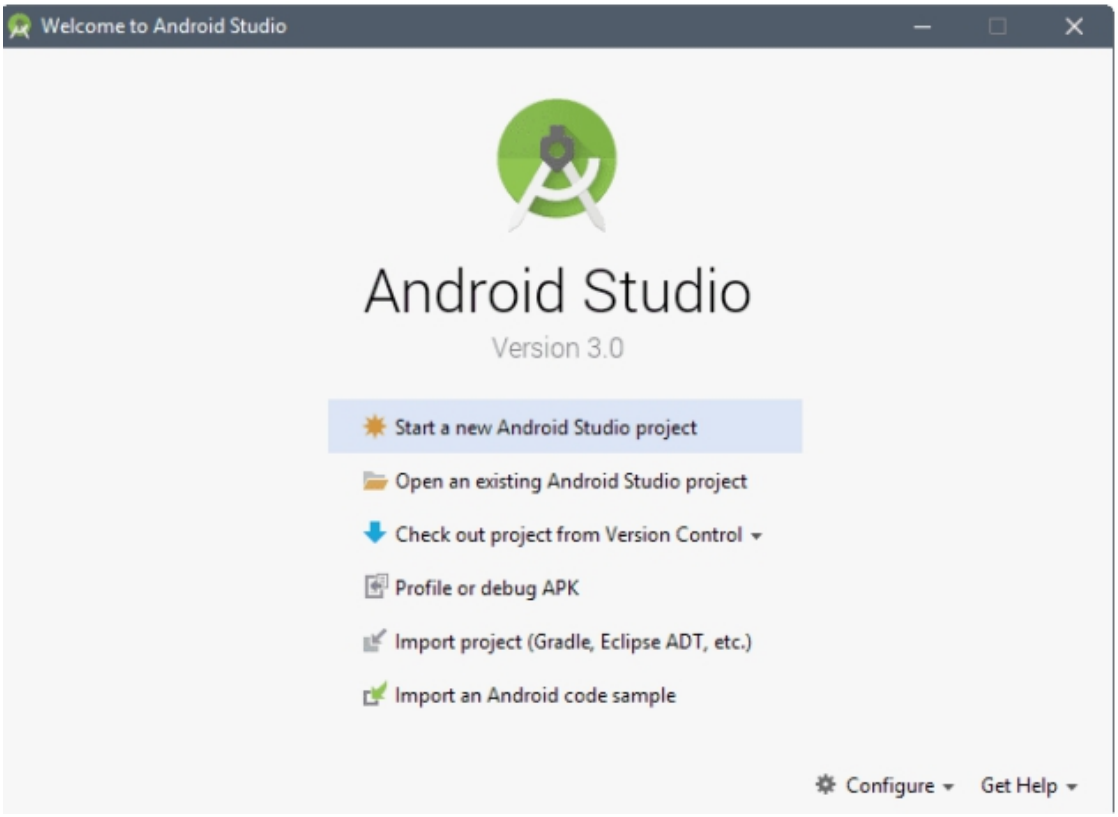


Рисунок 2.5 - Стартовий екран Android Studio

Середовище Android Studio є вікном. Воно максимально використовує простір екрану і щоб не перевантажувати людину якак разробляє Android Studio відображає лише не велику кількість допустимих вікон. Також середовище містить приховані вікна, які розробник може активувати або деактивувати з меню налаштувань. Найважливіша функція є навігація. Проекти зазвичай

маштабні та складаються з каталогів файлів і тому подібне, що організовуються між собою. Програмні додатки з багатьма файлами та ресурсами організовані в структурі проекту. В цілому файлова структура і називається проектом, а сама структура являється організаційним рішенням. На етапі збірки проекту створюються файли конфігурації та відбувається структуризація каталогів за ієрархією. Рисунок 2.6

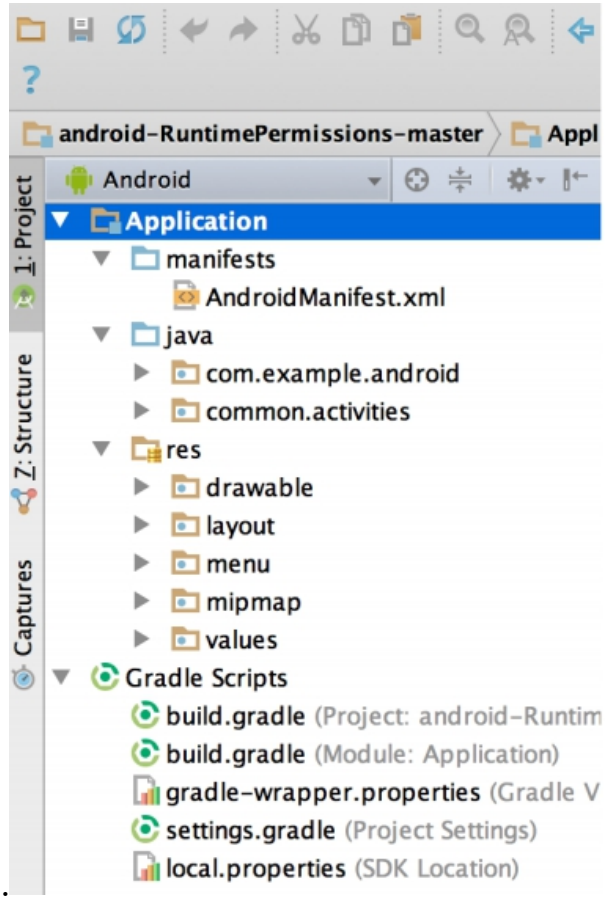


Рисунок 2.6 - Структура проекту в Android Studio

В процеці розробки додатків в інтегрованому середовищі розробки Android Studio необхідно буде компілювати та запускати додаток кілька разів. Розроблений одаток можна первірити, встановити та запустити на фізичному пристрої або на віртуальному пристрої Android (AVD - android virtual device), який називають емулятором. Емулятор надає вс’ можливості реального

пристрою Android. Тестувати на віртуальному пристрої зручніше та швидше, ніж на фізичному. Рисунок 2.7

Рисунок 2.7 - Віртуальний девайс в Android Studio

2.4 Flutter - фреймворк

2.4.1 Визначення

Flutter - це набір інструментів для інтерфейсу корустувача від Google для створення чудових, власно скомпільованих програм для мобільних пристроїв, Інтернету та робочого столу з однієї кодової бази.

Простіше кажучи, за допомогою Flutter можливо створити власний мобільний додаток з одним масивом коду.

Flutter - це кроссплатформенность

Кроссплатформені додатки — це давня мрія бодь-якого бізнесу, тому що окремі додатки для iOS та Android значно дорожче в розробці і підтримці. В 2018 році команда Surf серйозно взялась за розробку Кроссплатформенного напрямку. В данних умовах найкращим рішенням для рітейлу, фінтех і commerce став Flutter.

2.4.2 Загальні функції

Фреймворки надають функціональність у своєму коді або через розширення для завершення повсякденних операцій, необхідних для запуску Flutter-додатків. Висока продуктивність програм і швидкість розробки досягає за рахунок таких технік:

1. Не використовую JavaScript. У якості мови програмування використовується Dart.
2. Flutter не використовує нативні компоненти.
3. Для побудування UI під Flutter використовується декларативний підхід, натхненний веб-фреймворком ReactJS, на основі віджетів (в світі інтернету іменованих компонентами).
4. В фреймворк вбудований Hot-reload, такий звичний для вебу, і досі не було в нативних платформах.

Звичайний додаток був складений з чотирьох слоїв:

1. Data - слой роботи з даними. На цьому рівні описували роботу з API.
2. Domain - слой бізнес-логіки.
3. Internal - слой додатку. Нацьому рівні проходило дадавання залежностей.
4. Presentation - слой представлення. На цьому рівні описуемо UI додатку.

Слої не мають представлення звідки вони беруть дані, та куда їх передають, вони не знають дані і те як і хто буде їх використовувати.

2.4.3 Створення проекту

При вже встановленому Flutter, ми створюємо проект за допомогою інструментів IDE або командної строки. В останьому випадку вприсуємо комаду: *flutter create myapp.*

В результаті буде створено проект. Додаток буде з 4-х слоїв, тому створюємо потрібні папки.

Отримуємо таку структуру каталогів. Див. рисунок 2.8

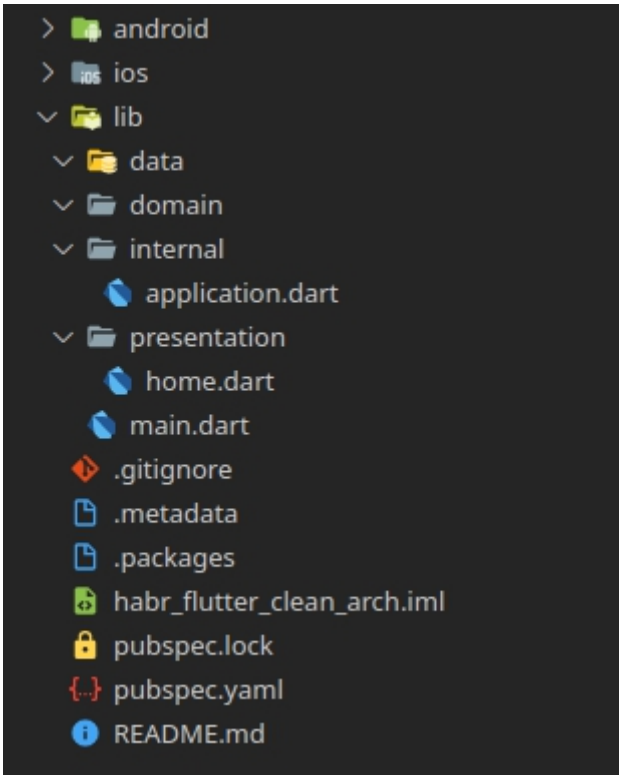


Рисунок 2.8 - Сткуртура каталогів.

Зміст файлів .dart можемо побачити в Додаток Б сторінка 49.

Далі готовимо domain

Кожен додаток живе, якщо у нього є данні, які перетікають з однієї його частини в іншу. Аби створили цей невеликий додаток нам треба буде використати безплатний сервіс Sunrise Sunset, який надасть інформацію о дні, та часі.

До сервіса ми будемо робити таку запроси:

1. Час сходу.
2. Час заходу.
3. Час, в яке наступає астрономічний полудень.
4. Тривалість дня.

Тепер ми можемо зробити першу модель. Додаємо в папку domain директорію model, в котрій створимо файл з іменем day.dart. Опишемо в цьому файлы модель:

```
import 'package:meta/meta.dart';
class Day {
final DateTime sunrise;
final DateTime sunset;
final DateTime solarNoon;
final int dayLength; Day({
@required this.sunrise,
@required this.sunset,
@required this.solarNoon,
@required this.dayLength, });
}
```

Тут ми визначили конструктор з іменованими аргументами, а анотація @required говорить нам про те, що всі аргументи є обов'язковими.

2.5 Мова програмування Dart

Dart - це оптимізована клієнтом мова для розробки швидких додатків на будь-якій платформі. Його мета - запропонувати найпродуктивнішу мову програмування для розробки мультиплатформ у поєднанні з гнучкою платформою виконання для фреймворків програм.

Мови визначаються їх технічною оболонкою - рішеннями, зробленими під час розвитку, що формують можливості та сильні сторони мови. Dart розроблений для технічного пакету, який особливо підходить для розробки

клієнтів, надаючи пріоритет як розробці (допоміжна гаряча перезавантаження), так і високоякісному виробничому досвіду для широкого спектру цілей компіляції (Інтернет, мобільні та настільні комп'ютери).

Дартс також складає основу Flutter . Dart надає мову та час роботи, які працюють із програмами Flutter, але Dart також підтримує багато основних завдань розробника, таких як форматування, аналіз та тестування коду.

Мова Дарт безпечна для друку; він використовує перевірку статичного типу, щоб значення змінної завжди відповідало статичному типу змінної. Іноді це називають набором звуку. Хоча типи є обов'язковими, анотації типів є необов'язковими через висновок типу. Система набору тексту Dart також гнучка, дозволяючи використовувати dynamicтип, що поєднується з перевірками часу виконання, що може бути корисним під час експериментів або для коду, який повинен бути особливо динамічним.

Наведений зразок коду демонструє декілька функцій мови Dart, включаючи бібліотеки, асинхронні виклики, типи, що дозволяють обнуляти та не допускають дозволу, синтаксис стрілок, генератори, потоки та геттери. Дивитися додаток А сторінка 47. Синтаксис Dart дуже схожий на синтаксис Java, C# та Java Script. Це не випадковість - при створенні Dart ставилась задача розробити знайому мову. Ось невеликий скрипт на Dart, складений із однієї функції main Рисунок 2.9.

```
main() {

  var d = "Dart";
  String w = "World";
  print("Hello ${d} ${w}");
}
```

Рисунок 2.9 - Привіт Світ

Функція main() - точка вхід, з котрої починається виповнення скрипта

Var d = “Dart”; - Типизація необов’язкова (тип не вказаний)

String w = “World”; - Анотація типу(заданий тип String)

Print(“Hello ${d} ${w}”); - Для виводу на консоль браузера або на stdout проводиться підстановка в рядок.

Цей скрипт можна добавити в HTML - сторінку за допомогою тегу <script type="application/dart"> і виповнити в браузері Dartium(браузер випущений Google Chrome, преднаписаний для розробки на Dart).

Щоб знайти приклади використання додаткових функцій Dart, дивиться нижче. Щоб дізнатись більше про мову, відвідайте мовний тур Dart .

```
var name = 'Voyager I';
var year = 1977;
var antennaDiameter = 3.7;
var flybyObjects = ['Jupiter', 'Saturn', 'Uranus', 'Neptune'];
var image = {
  'tags': ['saturn'],
  'url': '//path/to/saturn.jpg'
};
```

2.5.1 Dart: The libraries

Dart має багатий набір основних бібліотек , що забезпечують необхідне для багатьох повсякденних завдань програмування:

1. Вбудовані типи, колекції та інші основні функції для кожної програми Dart (dart:core).
2. Багатіші типи колекцій, такі як черги, зв’язані списки, хеш-карти та двійкові дерева (dart:collection).
3. Кодери та декодери для перетворення між різними представленнями даних, включаючи JSON та UTF-8 (dart:convert).

4. Математичні константи та функції, а також генерація випадкових чисел (dart:math).

5. Файл, сокет, HTTP та інша підтримка вводу-виводу для не веб-програм (dart:io).

6. Підтримка асинхронного програмування з такими класами, як Futurei Stream (dart:async).

7. Списки, які ефективно обробляють дані фіксованого розміру (наприклад, беззнакові 8-байтові цілі числа) та числові типи SIMD (dart:typed_data).

8. Інтерфейси зовнішніх функцій для сумісності з іншим кодом, який представляє інтерфейс у стилі C (dart:ffi).

9. Одночасне програмування з використанням ізолятів - незалежних працівників, які схожі на потоки, але не діляться пам'яттю, спілкуючись лише за допомогою повідомлень (dart:isolate).

10. Елементи HTML та інші ресурси для веб-додатків, які повинні взаємодіяти з браузером та об'єктною моделлю документа (DOM) (dart:html)

Крім основних бібліотек, багато API надаються за допомогою повного набору пакетів. Команда Dart публікує багато корисних додаткових пакетів, таких як:

- characters
- intl
- http
- crypto
- markdown

Крім того, сторонні видавці та широке співтовариство публікують тисячі пакетів із підтримкою таких функцій:

- XML
- Windows integration
- SQLite

- compression

2.5.2 Dart: The platforms

Технологія компілятора Dart дозволяє запускати код по-різному:

1. **Власна платформа** : для додатків, орієнтованих на мобільні та настільні пристрої, Dart включає в себе як Dart VM із компіляцією "точно в час" (JIT), так і компілятор "AOT" для створення машинного коду.

2. **Веб-платформа** : для програм, націлених на Інтернет, Dart включає як компілятор часу розробки (dartdevc), так і компілятор робочого часу (dart2js). Обидва компілятори перекладають Dart в JavaScript.

Flutter-фреймворк є популярним, мультиплатформенний UI інструментарій , який працює на платформі Dart, і що забезпечує інструментальні та бібліотек призначеного для користувача інтерфейсу для досвіду побудови призначеного для користувача інтерфейсу , які працюють на IOS, Android, MacOS, Windows, Linux, і в Інтернеті. Дивитися рисунок 2.10

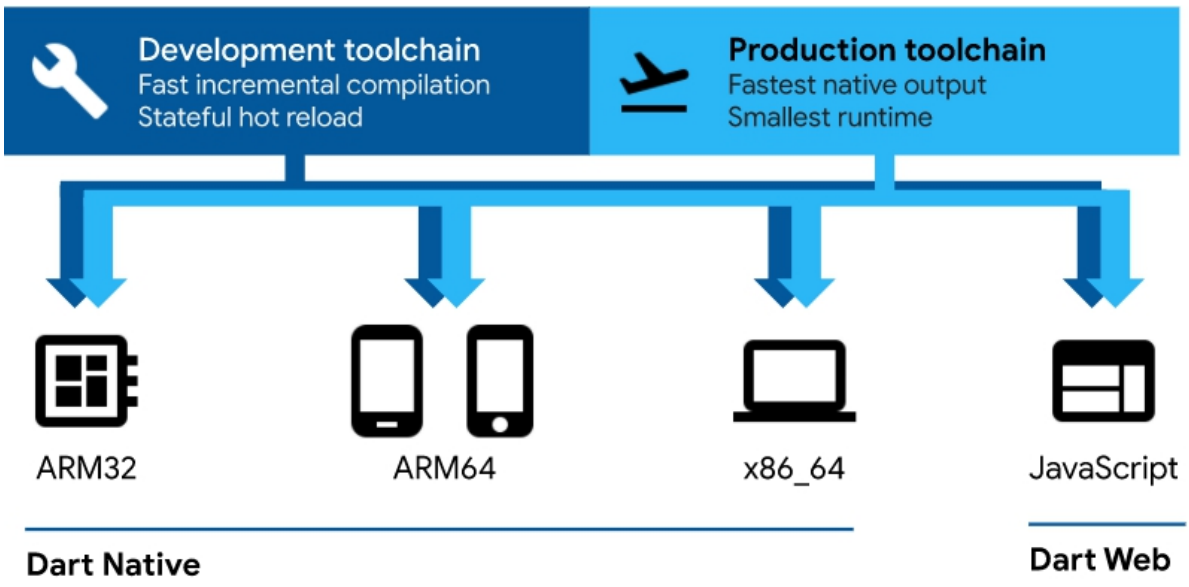


Рисунок 2.10 - Платформи на яких є можливіть працювати.

2.6 Firebase

Firebase - це Google платформа для розробки мобільних додатків, яка допомагає вам створювати, вдосконалювати та розвивати свій додаток.

Firebase - це набір інструментів для «побудови, вдосконалення та розвитку додатка», а інструменти, які він надає, охоплюють значну частину служб, які розробники зазвичай повинні будувати самі , але насправді не хочуть будувати, тому що вони воліють зосередитись на самому досвіді програми. Сюди входять такі речі, як аналітика, автентифікація, бази даних, конфігурація, зберігання файлів, push-повідомлення, і список можна продовжувати. Сервіси розміщуються у хмарі та масштабуються, практично не вимагаючи зусиль з боку розробника.

Клієнтський SDKs наданого Firebase взаємодіє з цими серверними послугами безпосередньо , без необхідності створення якого - або проміжного рівня між додатком і сервісом рисунок 2.11 Отже, якщо ви використовуєте один із параметрів бази даних Firebase, ви зазвичай пишете код для запиту бази даних у вашому клієнтському додатку.

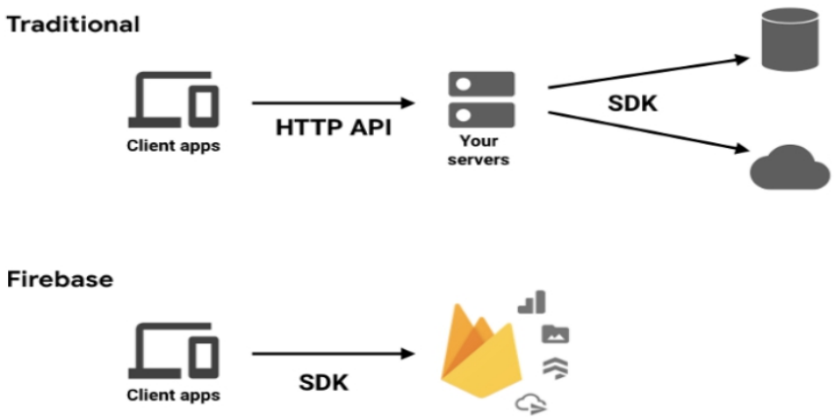


Рисунок 2.11 - Робота баз даних.

Це відрізняється від традиційної розробки додатків, яка, як правило, передбачає написання як інтерфейсного, так і внутрішнього програмного забезпечення. Код зовнішнього інтерфейсу просто викликає кінцеві точки API, виставлені серверною базою, і внутрішній код фактично виконує роботу. Однак у продуктах Firebase традиційний бекенд обходить, покладаючи роботу на клієнта. Адміністративний доступ до кожного з цих продуктів забезпечує консоль Firebase .

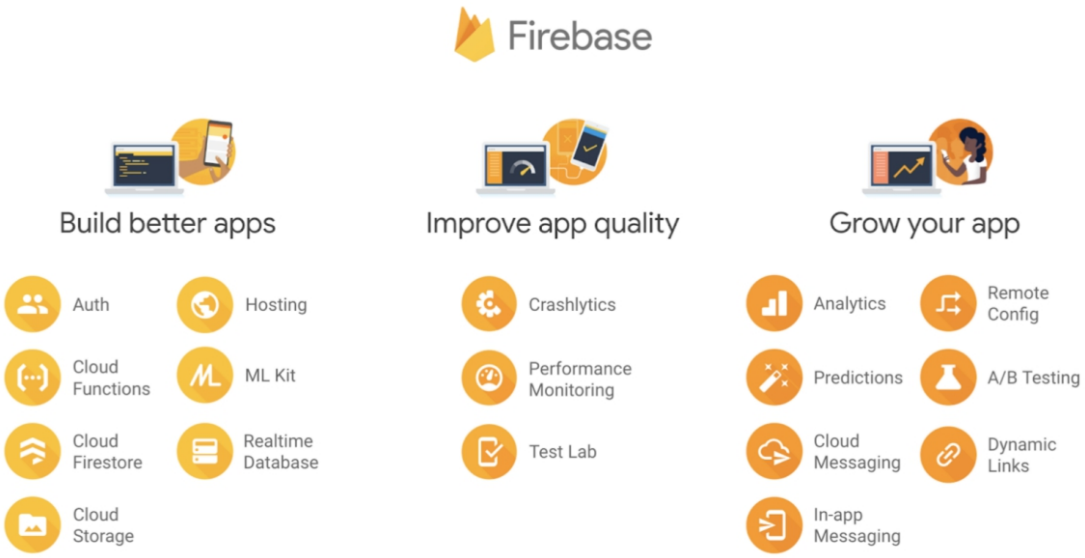


Рисунок 2.12 - Продукти та набори

На даний момент нараховується 17 окремих продуктів у наборі Firebase рисунок 2.12.

Створіть свій додаток - створіть “організм”

В групою продуктів “побудова” є такі:

В группі збірка є такі частини:

1. Аутентифікація - вхід та ідентифікація користувача.

2. База даних у реальному часі - в реальному часі, розміщена у хмарі, база даних NoSQL.

3. Cloud Firestore - в реального часу, розміщена у хмарі, база даних NoSQL.

4. Cloud Storage - широко масштабоване сховище файлів.
5. Cloud Functions - „безсерверне”, керований подіями backend.
6. Firebase - глобальний веб-хостинг.
7. ML Набір - SDK для загальних завдань ML.

2.6.1 Firebase Authentication

Він дбає про те, щоб ваші користувачі увійшли в систему та ідентифікували їх. Цей продукт необхідний для правильного налаштування деяких інших продуктів, особливо якщо вам потрібно обмежити доступ до даних для кожного користувача (що хоче зробити майже кожна програма).

“ об’єднана ідентифікація ” означає, що ви можете пов’язати облікові записи користувачів від різних постачальників ідентифікаційних даних (Facebook, Twitter, Google, GitHub) в єдиний обліковий запис у службі аутентифікації Firebase.

2.6.2 Firebase Realtime Database та Cloud Firestore

Вони надають послуги баз даних. Я перерахував їх обох як "базу даних NoSQL у режимі реального часу, розміщену у хмарі". Вони мають індивідуальні сильні та слабкі сторони, і, можливо, вам доведеться провести дослідження, щоб з’ясувати, який з них найкращий для ваших потреб. Підказка: починайте з Cloud Firestore, оскільки він, ймовірно, відповідає більшій кількості ваших потреб (і це також масштабовано). Ви можете використовувати один або обидва разом, якщо це відповідає вашому додатку

2.6.3 Cloud Storage

Має задачу забезпечувати масштабоване зберігання файлів. Технічно це також Google Cloud , а не продукт Firebase. За допомогою Cloud Storage *для Firebase* ви отримуєте клієнтські SDK для використання у вашому додатку, які

дозволяють завантажувати та завантажувати файли безпосередньо у хмарного сховища та з нього.

2.6.4 Cloud Functions

Це ще один Google Cloud, який добре працює з іншими продуктами Firebase та Cloud. Використовуючи Firebase SDK для хмарних функцій, ви можете писати та розгортати код, що працює на безсерверній інфраструктурі Google, який автоматично реагує на події, що надходять з інших продуктів Firebase. Правильно, це без серверів!

2.6.5 Firebase Hosting

Безпечний глобальний веб-хостинг CDN (Content Delivery Network). Це дійсно добре швидко доставляти статичний вміст (HTML, CSS, JS, зображення) за допомогою серверів, близьких до ваших користувачів. Ви можете швидко налаштувати його, із власним доменом або без нього, а також із наданим сертифікатом SSL, який вам нічого не коштує.

2.6.6 ML Kit для Firebase

Дозволяє скористатися багатим досвідом машинного навчання від Google, не знаючи нічого про ML. Це чудово для мене, бо я нічого не знаю про ML! Але те, що я отримую від ML Kit, - це здатність розпізнавати речі, які фіксує камера мого пристрою, такі як текст, обличчя та орієнтири. І це може працювати на моєму мобільному пристрої з дуже обмеженими обчислювальними можливостями. Для тих, хто має більш досконале розуміння ML (знову ж таки, не я), ви можете завантажити модель TensorFlow для більш складних випадків використання. Дорожня карта продуктів машинного навчання у Firebase буде повністю "об'єднаною":

2.7 Висновки до розділу

В даному розділі було виконано аналіз та огляд програмних технологій та використаних засобів у процесі розробки програмного продукту.Також обгунтований вибір використаних засобів при розробці.

Система завчасного замовлення буде складатися з декількох вікон керування. Головним буде вікно яке одразу буде відкриватися при відкритті додатку так звана “Головна сторінка”. На ній буде профіль користувача, пропозиції та знижки та панель з вибором кави. Система матиме декілька допоміжних вікон для взаємодії програмного продукту та користувача. Обрана структура буде простою та інтуїтивно зрозумілоб для користувача за рахунок чіткої та зрозумілої організації модулів.

3.1 Опис функціональності системи

Система дистанційного навчання містить у собі два актора, які взяємодіють із системою а також один з одним.

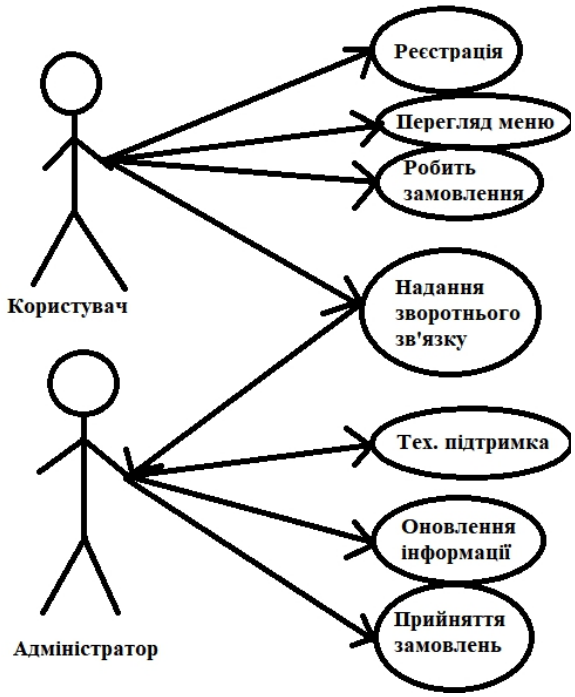
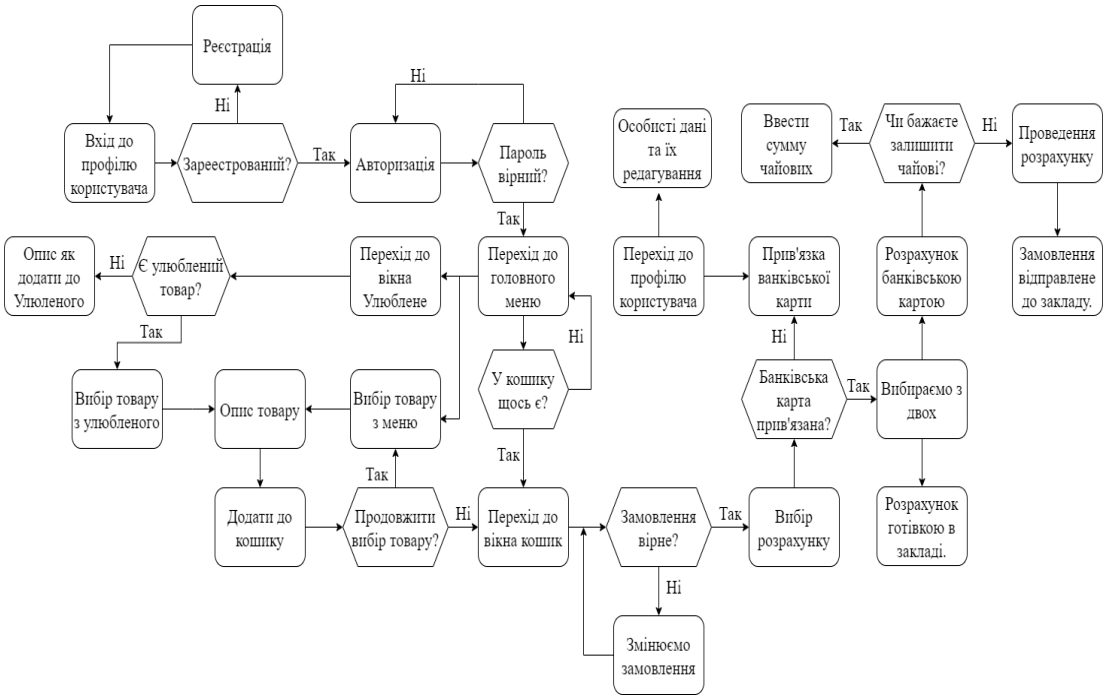


Рисунок 3.1 - Діаграма прецедентів системи

На Рисунок 3.1 представлена діаграма прецедентів, яка описує функції та дії акторів у системі дистанційного навчання.

Перший актор - це користувач який взаємодіє із системою додатку за допомогою смартфона. Другий актор - це адміністратор додатку який наповнює систему контентом, приймае закази та надає необхідну технічну підтримку користувачу.

Блок-схема роботи додатку зображена на рисунку 3.2



Ристунок 3.2 - Блок-сехма роботи додатку

Діаграма станів діяльності дивитися рисунок 3.3

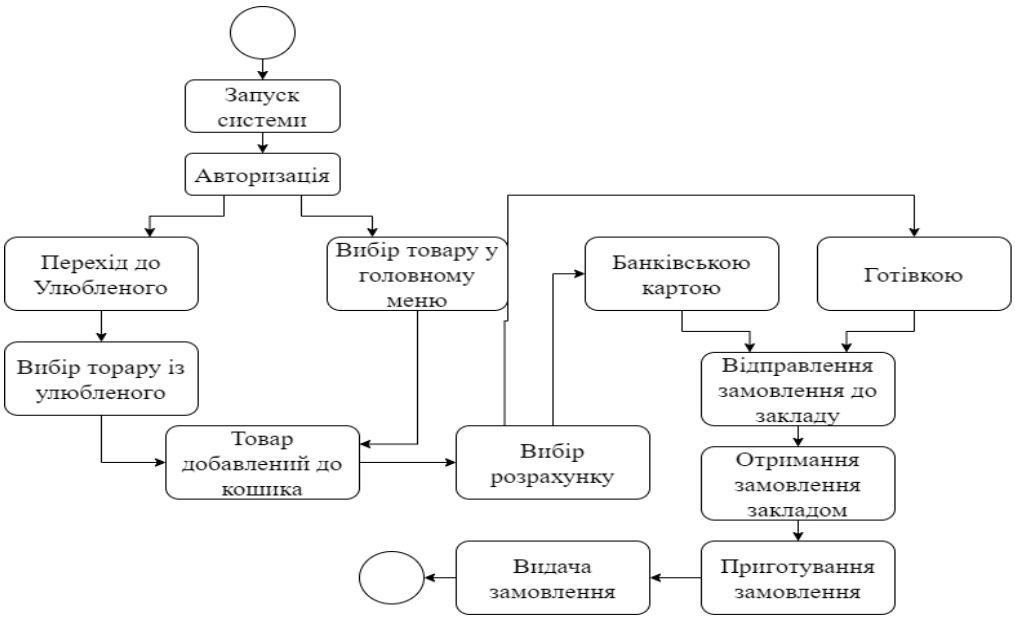


Рисунок 3.3 - Діаграма станів діяльності

3.2 Розробка інтерфейсу користувача

Графічний інтерфейс користувача має відповідати наступним вимогвам: якість, відповідність, виразність, послідовність. Для забезпечення цих вимог буде розроблений інтерфейс програмного продукту. Він має в собі декілька модулів. Це головне меню, перехід до профілю користувача, контакти та меню закладу. Все меню розташоване по порядку. Після того як користувач обирає щось з меню, він переходить у вікно добавок, в якому відображається все що можливо добавити в той чи інший товар.

3.2.1 Вікно головного меню

Дане вікно слугує для виведення на екран користувача всього самого головного і вибору користувачем того що йому потрібно. Маються такі єлементи:

1. Привітання користувача у верхній панелі та перехід в особистий профіль.
2. Меню в якому користувач вибирає той чи інший товар.
3. Контакти закладу.

4. Кошик з покупками.

3.2.2 Вікно профілю користувача

Дане вікно призване для виведення інформації користувача про нього самого. А також матиме такі функції:

1. Зв’язок з тех. Підтримкою.
2. Можливість прив’язати банківську картку.
3. Можливість прив’язати мобільний телефон.
4. Редагування профілю.

3.2.3 Вікно добавок

Дане вікно має призначення більш точно дізнатися замовлення користувача, додаток не надасть можливість додати щось до кошику минув це вікно. Якщо користувач вибрав щось із меню в що не входять добавки вікно не з’явиться, а товар буде одразу у кошику. У вікні буде:

1. Перецічені усі добавки для цього товару.
2. Кількість товару яку потрібно додати до кошику.
3. Можливість додати товар без добавок.

3.2.4 Вікно кошик з покупками

Дане вікно має виводити вікно з заказом, але якщо в кошику нічого немая воно не відкриється. У вікні буде можливість редагувати свій заказ. Та інші функції:

1. Видалити з кошику товар.
2. Змінити кількість
3. Вибір розрахунку, готівкою або банківською картою.
4. Сума замовлення.
5. Знижка якщо так мається.
6. Кнопка оплата товару.

7. Залишити чайові.

Можливість залишити чайові - це можливаість для користувачів які розраховуються онлайн банкістькою картою. Тобто у людини є сумма замовлення наприклад п’ятдесят чотири гривні, у строчні “чайові” користувач може ввести наприклад п’ять гривень і з його картки буде списано не п’ятдесят чотири гривні, а сумму з чайовими це п’ятдесят дев’ять гривень.

3.3 Розробка інтерфейсу адміністратора

Графічний інтерфейс адміністратора має відповідати наступним вимогвам: якість, відповідність, виразність. Для забезпечення цих вимог буде розроблений інтерфейс програмного продукту. Він має в собі декілька модулів. Це головне меню, управління знижками, приймання замовлень, управління профілями користувачів. Все в головному меню розташоване по порядку.

3.3.1 Вікно головного меню

Це вікно відповідає за навігацію в адмін панелі. У вікні розташований перехід між іншими вікнами, а саме:

1. Перехід до замовлень користувачів.
2. Перехід до знижок та пропозицій.
3. Перехід до профілів користуівачів

3.3.2 Вікно замовлень користувачів

Дане вінко дає можливість побачити все що замовив той чи інший користувач.

У вікні можна буде побачити наступне:

1. Розгорнуті замовлення користувачів.
2. Час в який було зроблене замовлення.
3. Яким методом був або буде розрахунок.

3.3.3 Вікно знижок на пропозицій

У цьому вікні адміністратор має бачити знижки. Знижки виставляє сам адміністратор. Знижки будуть виставлятися на пейний час, це може буди день, а може буди тиждень, або навіть декілька годин. Пропозиції будуть працювати по типу “три по ціні двох” або візьми цей товар з цим і це буде дешевше.

1. Перелік товару на який можливо зробити знижку.
2. Пропозиції в які можливо додати товар.

3.3.4 Вікно управління профілями користувачів.

Дане вікно слугує для виведення на єкран адміністратора інформації про користувачів, а також можливість змінити певні пункти:

1. Знижка для користувача.
2. Видалити користувача якщо це потребується.

Знижка користувача працює таким чином: якщо користувач приходить до закладу кожен день, за ним є можливість зафіксувати постійку знижку “Постійного гостя”. Ця знижка має можливість автоматично робити знижку на товари в кошику користувача.

3.4 Висновки до розділу

У розділі було з’ясовано роль розроблюваного програмно-апаратного складу у системі завчасного замовлення кави. Приведена архітектура, опис основних змінних і функцій, схема збору апаратної частини розроблюваної системи.

4 РОБОТА КОРИСТУВАЧА В СИСТЕМІ

Система предзамовлення в закладі харчування розроблена для використання на мобільних пристроях (смартфони та планшени) під управлінням операційної системи “Android”. Додаток використовує підключення до мережі інтернет за допомогою Wi-fi на плашеті та смарфоні або мобільних даних на смартфоні та планшеті(якщо є така можливість) для взаємодії з інтерактивними елементами додатку.

4.1 Інсталяція та системні вимоги

Користувач повинен завантажити додаток. Додаток встановлюється на пристрій шляхом встановлення файлу “apk” та автоматично налаштовується. Так як додаток був розроблений на платформу “Android”, пристрій користувача повинен працювати саме на цій операційній системі. Для коректної роботи додатку на пристрої користувача повенна були операційна система не нижче версії 4.0.0. А також користувач повинен мати доступ до інтернету.

4.2 Інструкція з використання програмного продукту

При вході в додаток перед користувачем з’явиться меню входу в користувацький профіль. Після того як користувач зареєструється та виконає вхід всистему, йому більше не буде відкриватися це вікно. Рисунок 4.1.

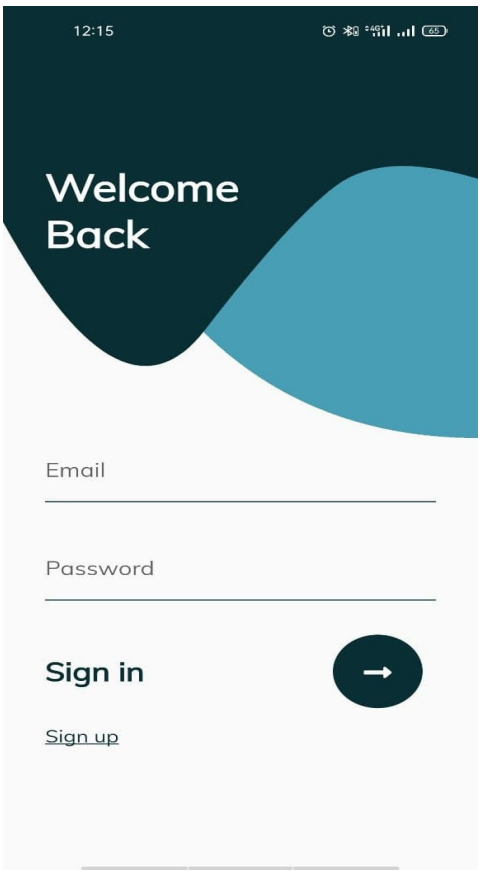


Рисунок 4.1 - Вікно Авторизації

При вході в додаток одразу відкриватиметься головне меню. Головне меню в собі має панель управління та меню. Див Рисунок 4.2.

В панелі управління є декілька сторінок.

1. Home - прехід на головну сторінку.

2. Order - Кошик користувача в якому знаходиться все що замовив користувач, тобто замовлення.

3. Favorites - Користувач може туди дотати улюблене, щоб не шукати весь час у меню.

4. Profile - Профіль користувача.

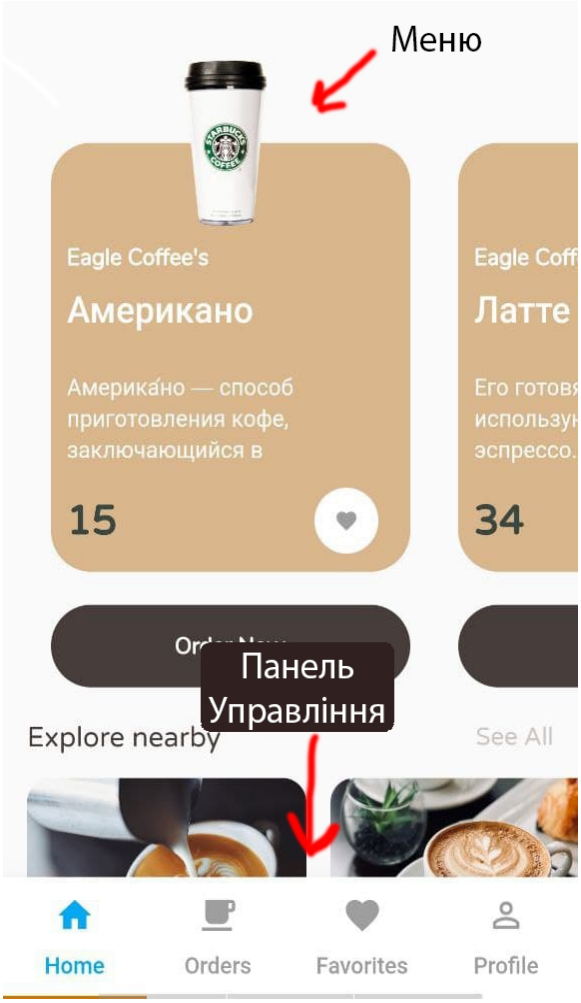


Рисунок 4.2 - Головне меню

Після того як користувач обрав один із запропонованих товарів, він натискає на “Order Now”. Перед ним з’яляється вікно товару. рисунок 4.3 Користувач отримує інформацію про товар який вибрав. В інформацю про товар входить

1. Опис товару.
2. Інградієтни.
3. Фото товару.

Фото є невеликуе але дозволяє побачити як виглядає цей товар.

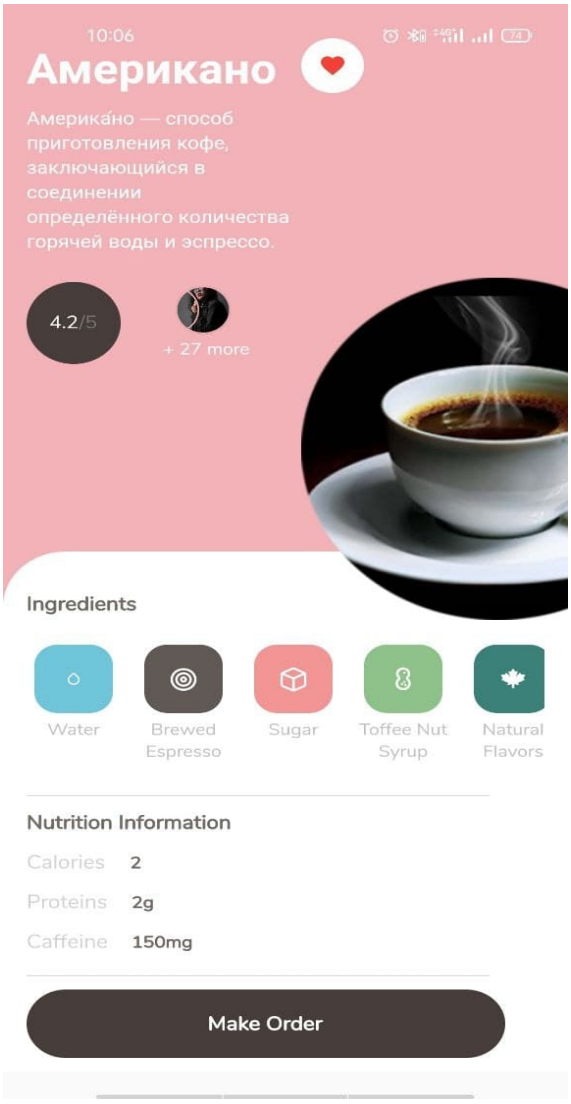


Рисунок 4.3 - Вікно товару

Після того як був вибраний товар, користувач натискає “Make Order”. Товар буде добавлений до кошику.

4.2.1 Вікно “Order“

Це місце де підбивається замовлення та йде розрахунок. Рисунок 4.4

В кошику є можливість змінити кількість вибраного товару та те скільки він коштує. Далі йде вибір розрахунку: готівкою або онлайн банківською картою. Також є можливість дати чайові. Після того як усі дії були виконані потрібно натиснути “Відправити замовлення”.

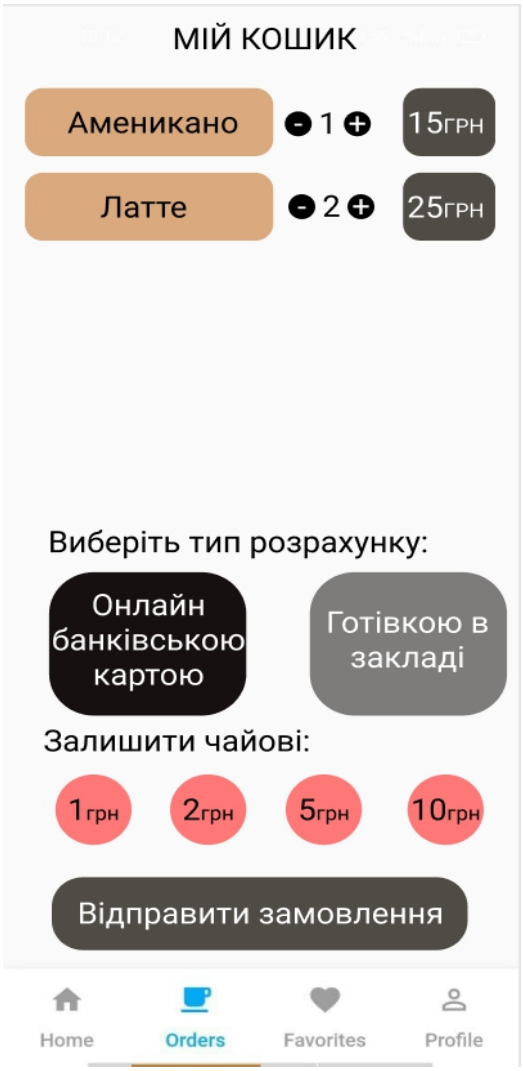


Рисунок 4.4 - Вікно “Order”

Далі замовлення відпривиться в заклад, де його зроблять до вашого приходу.

Після того як замовлення відправлене до закладу з кошика вото буде видалено.

4.2.2. Вікно “Profile”

Це вікно виводить на єкран користувача інформацію про кристувача та його дані. На рисуноку 4.5 користувач зможе побачимо повну інформацію про себе, а саме:

1. Прізвище та Ім’я.
2. Електронну пошту користувача.
3. Номер телефону.
4. Фотографію якщо коистувач її буде вважати за потрібне встановити.
5. Також бачимо можливість прив’язати банківстьку карту.

Рисунок 4.5 - Вікно “Profile”

Інформація про користувача є для того щоб ідентифікувати користувача який зробив замовлення та віддати йому саме його замовлення. Фото також є елементом ідентифікації бо переплутати людину по фото дуже тяжко. Також є номер телефону по якому є можливість перевірити деякі моменти замовлення. Банківську картку можна прив’язати для більш швидшого розрахунку. Карткою користувач може розраховуватися в вікні кошик.

4.3 Висновки до розділу

Зроблений інтерфейс зручний у користуванні, стійкий до внесення користувачем невірних даних та помилкових дій, при виникнетті помилок роботи системи виводиться повідомлення про способи їх усунення, а також має приємне оформлення в приємній кольоровій гамі.

ВИСНОВКИ

В данному дипломному проекті було досліджено та проаналізовано доцільність створення додатку предзамовлення для закладу сферу обслуговування.

Для зручного представлення даних користувачу було створено мобільний додаток для пристроїв, що керується операційною системою Android. Розроблений програмний продукт, дозволяє зробити предзамовлення в закладі який користується цим додатком. Користувач робить замовлення з дому, або десь поблизу закладу. Замовлення відправляється до закладу. Адміністратор бачить це замовлення та робить його. Також була представлена інструкція користування додатком для користувача.

Будо описано методи та засоби для розробки програмногої системи. Обгрунтовано створення додатку та використання за допомогою мобільного пристрою. Використані засоби надають ряд переваг, таких як: гнучність, зручність у використанні та можливість переносу додатку на iOS.

Загалом в дипломному проекті вирішена поставлена задача, розглянуто супутні завдання та вирішено проблеми, які виникали в ході виконання завдання.

ПЕРЕЛІК ПОСИЛАНЬ

1. К. Бакетт. / Dart in action / - 2013 / - С. 30-33.
2. E. Windmill. / Flutter in Action / - 2020 / - С. 29-37. С.213.
3. What is android [Електронний ресурс]. / - 2008 / - Режим доступу: www.android.com.
4. Use cases [Електронний ресурс]. / - 2014 / - Режим доступу: firebase.google.com
5. User Guid [Електронний ресурс]. / - 2013 / - Режим доступу: developer.android.com
6. J. F. DiMarzio / Beginning Android Programming with Android Studio / - 2016 / - С. 29-37. С.213.
7. Кеті Вальрат, Сет Ледд / Dart: Up and Running / - 2012 / - С. 29-37. С.213.
8. Лоуренс Мороні / The Definitive Guide to Firebase / - 2017 / - С. 29-37. С.213.
9. Rap Payne / Beginning App Development with Flutter: / - 2019 / - С. 29-37. С.213.
10. Кеті Вальрат, Сет Ледд / Dart: Up and Running / - 2020 / - С. 29-37. С.213.

Додаток А

```
import 'dart:math' show Random;
void main() async {
  print('Compute π using the Monte Carlo method.');
  await for (final estimate in computePi().take(100)) {
    print('π ≅  $estimate');
  }
}
/// Generates a stream of increasingly accurate estimates of π.
Stream<double> computePi({int batch = 100000}) async* {
  var total = 0; // inferred to be of type int
  var count = 0;
  while (true) {
    final points = generateRandom().take(batch);
    final inside = points.where((p) => p.isInsideUnitCircle);
    total += batch;
    count += inside.length;
    final ratio = count / total;
    // Area of a circle is A = π · r², therefore π = A/r².
    // So, when given random points with x ∈ <0,1>,
    // y ∈ <0,1>, the ratio of those inside a unit circle
    // should approach π / 4. Therefore, the value of π
```

```
    // should be:
    yield ratio * 4;
  }
}
Iterable<Point> generateRandom([int? seed]) sync* {
  final random = Random(seed);
  while (true) {
    yield Point(random.nextDouble(), random.nextDouble());
  }
}
class Point {
  const Point(this.x, this.y);
  final double x, y;
  bool get isInsideUnitCircle => x * x + y * y <= 1;
}
```

Додаток Б

main.dart

```
import 'package:flutter/material.dart';
import 'internal/application.dart';
void main() {
 runApp(Application());
}
```

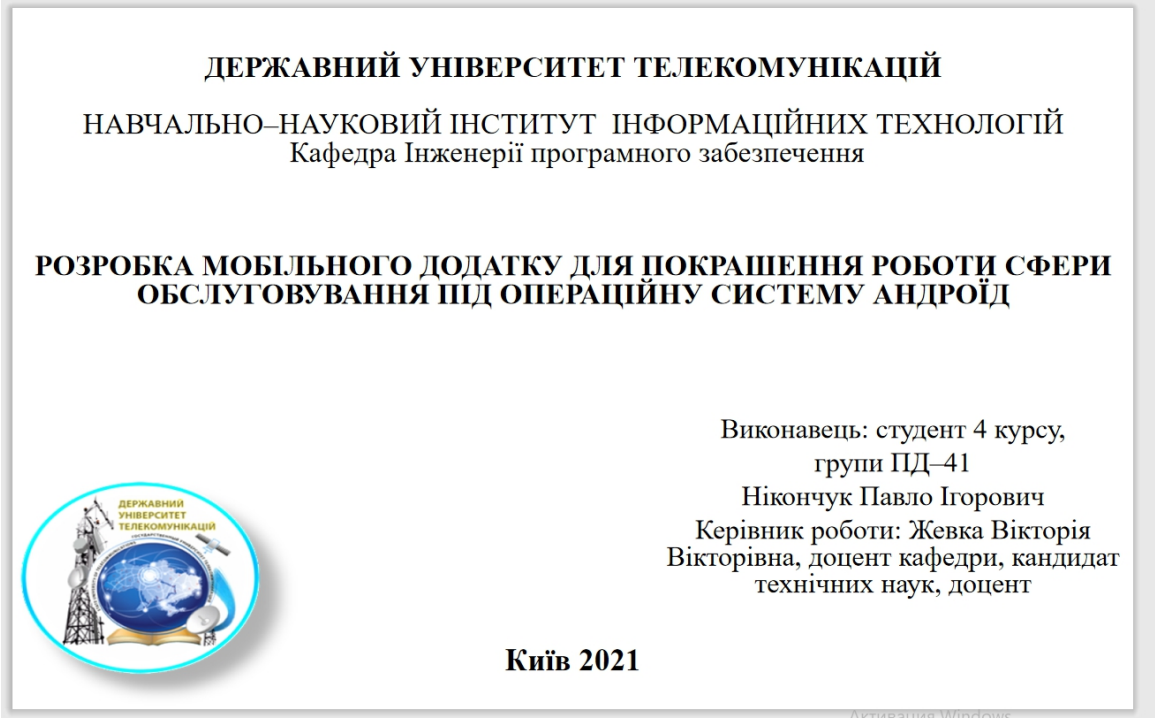
application.dart

```
import 'package:flutter/material.dart';import 'package:habr_flutter_clean_arch/presentation/home.dart';
class Application extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
    title: 'Flutter Demo',
    theme: ThemeData(
      primarySwatch: Colors.blue,
      visualDensity: VisualDensity.adaptivePlatformDensity,
    ),
    home: Home(),
  );
 }
}
```

home.dart

```
import 'package:flutter/material.dart';
class Home extends StatefulWidget {
  @override
    HomeState createState() =>   HomeState();
}
class _HomeState extends State<Home> {
  @override
  Widget build(BuildContext context) {
    return Scaffold();
  }
}
```

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)



ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра Інженерії програмного забезпечення

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ ПОКРАЩЕННЯ РОБОТИ СФЕРИ ОБСЛУГОВУВАННЯ ПІД ОПЕРАЦІЙНУ СИСТЕМУ АНДРОЇД

Виконавець: студент 4 курсу,
групи ПД–41
Нікончук Павло Ігорович
Керівник роботи: Жевка Вікторія
Вікторівна, доцент кафедри, кандидат
технічних наук, доцент

Київ 2021

Мета, об’єкт та предмет роботи

- ***Мета роботи*** – Зменшити час очікування клієнтом замовлення, знайти найзручніший інтерфейс для клієнта.
- ***Об'єкт дослідження*** – Мобільні додатки які використовуються в сфері обслуговування та швидкість обслуговування в закладах харчування, супермаркетах та інше.
- ***Предмет дослідження*** – Додаток в закладах в сфері обслуговування.

Аналіз аналогів

Переваги

- Простий у використанні.
- Має арфів покопок.
- Є інформації про знижки та акції.
- Можливість не носити картку супермаркету.

Недоліки

- Навантаження на обчислювальні модулі.
- Не побачив самовивозу товару одразу при оплаті.

Технічне завдання

- Система реєстрації.
- Оптимізація категорій.
- Кошик для замовлення.
- Система онлайн розрахунку банківською картою.
- Профіль користувача.

Спеціалізовані технології

- Платформа розробки додатку - Android.
- Мови програмування – Dart.
- Фреймворк - Flatter.
- Середовище розробки - Android Studio.
- Середовище Бази даних - Firebase

Архітектура Flutter

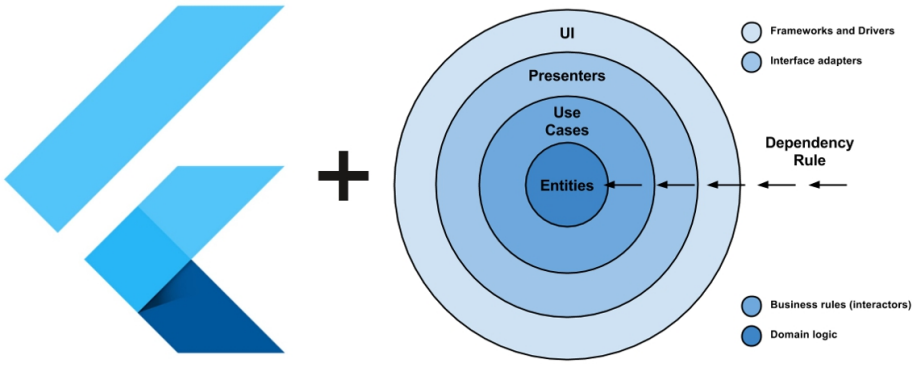
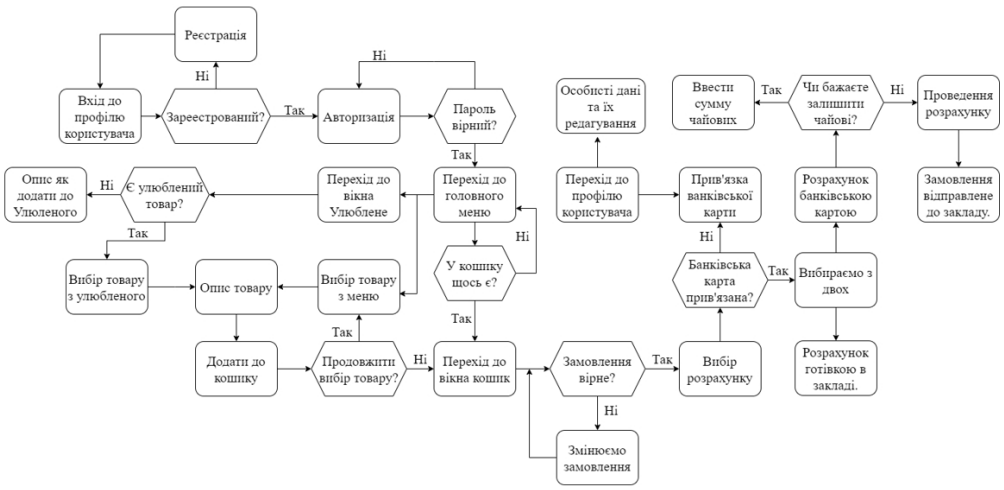
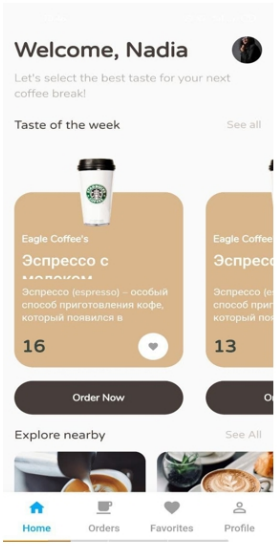
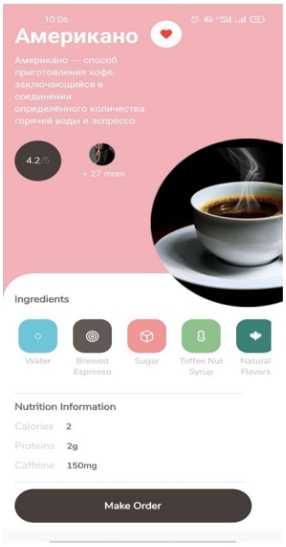


Схема роботи системи



Практичне застосування





Наукова новизна та практична значимість

Наукова новизна дослідження покладена на прискорення роботи так званого “fastfood” швидкої їжі. Спробувати зменшити черги. Знайти та розробити простий інтерфейс, в якому буде тільки найважливіше. Інтерфейс повинен бути простим з яким розбереться не тільки дитина, а і ще людина яка не дуже ладить з смартфоном.

Практична значимість дослідження полягає в створенні продукту на платформу Android, який дозволить замовити собі каву, будучи при цьому ефективним, компактним, легким в використанні.

Висновки

Створено бюджетну та ефективну систему пред. замовлення для закладу сфери обслуговування на платформі Android, що підтверджую досягнення мети дипломного проекту.

Проект CoffeeShop має наступні переваги:

1. Відкриття додатку займає декілька секунд.
2. Система передає дані від користувача до адміністратора за секунди.
3. Додаток потребує досить мало оперативної памьяті, тому пряцює досить швидко.
4. Реалізовано безпечний та просту систему кодування файлів.
5. Бюджет проекту становить 0$. Так як все розроблялося своїми силами.

Є наміри протестувати проект в реальному житті.

Дякую за увагу!