Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної підготовки

випускна кваліфікаційна робота бакалавра

«ІНФОРМАЦІЙНА СИСТЕМА УПРАВЛІННЯ СЕЛЕКЦІЙНИМ КОНТЕНТОМ»

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проєкту

№	П.І.Б.	Група
1	Платов А. М.	КН-Д-191
2	Карпова А. М.	КН-П-192
3	Гуртовий М. В.	КН-П-191
4	Філатова О. С.	КН-П-191

Автор звіту

Карпова Аліна Максимівна

Одеса – 2023

АНОТАЦІЯ

УДК 004.9
К - 26

Карпова А.М. Інформаційна система управління селекційним контентом: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп'ютерні науки». - Одеса: ОТУШ, 2023 - 13 с.

Ця робота є дослідженням зосередженим на розробці і впровадженні інформаційної системи з метою покращення торговельних взаємовідносин між користувачами. Окрім цього система містить такі функциональності як: авторизація і автентифікація, механізми обмеження доступу і також можливість обмежені канали зв'язку між клієнтом та компанією-продавцем.

Важливість цієї роботи зумовлена поширеністю завдань, пов'язаних з торгівлею які базуються на фільтруванні результатів за різними критеріями. Ці критерії включають такі фактори, як категорія, оцінки якості і цінові категорії продукту. Розгляд цих критевіїв важливий для проведення комплексної оцінки волатильності результатів і прийняття обґрунтованих рішень. Важливо зазначити, що застосування цих критеріїв може зіткнутися з проблемами, такими як неточність даних або неповна інформація. Важливо зазначити, що застосування цих критеріїв може зіткнутися з проблемами неточності даних або неповна інформація. Отже, це сприяє постановці задач в умовах невизначеності або неповної визначеності вхідних даних. Крім того, актуальність цієї теми підкреслюється необхідністю впровадження ефективних мобільних інформаційних систем і посилення заходів захисту інформації, особливо коли йдеться про забезпечення конфіденційності та безпеки особистих даних.

Предметом дослідження було обрано впровадження інформаційної системи з метою покращення торговельних взаємовідносин між користувачами.У якості задач досліджень було встановлено наступне:

- Аналіз спрямований на визначення характерних особливостей і областей для вдосконалення в цих системах, приділяючи особливу увагу оптимізації функцій, пов'язаних з управління контентом.
- Оцінка програмних засобів та інтерфейсів, які забезпечують ефективну співпрацю між користувачами в інформаційній системі. Ця оцінка розглядає зручність використання, функціональності і сумісності визначених інструментів для вибраних цілей проєкту.
- Впровадити в систему результат попередніх досліджень, щоб полегшити взаємодію користувачів та оптимізувати процес взаємодії.
- Застосувати надійна заходи безпеки в інформаційній системі, щоб захистити дані користувачів і конфіденційність особистої інформації.

Методологія дослідження, яка використовується для досягнення поставлених цілей, передбачає поєднання системного аналізу, моделювання та когнітивних підходів. Методи дослідження перетинаються, щоб сформувати комплексну основу для проведення досліджень.

ЗМІСТ

ВСТУП

Розвиток і широке впровадження інформаційних технологій призвели до оцифрування та міграції численних послуг в онлайн-сферу. У стратегії розвитку інформаційного суспільства в Україні комунікаційні технології визнано необхідними інструментами соціально-економічного прогресу та ключовим каталізатором інноваційного економічного розвитку [1].

З поширенням сучасних мобільних пристроїв, таких як смартфони, планшети та ноутбуки, стали повсякденною нормою нашого життя. Ці пристрої зробили революцію в комунікації людей, сприяючи ефективним і персаналізованим обміном послуг. З появою платформ соціальних мереж, додатків для обміну повідомлень та онлайн-ринків, соціальний аспект спілкування між клієнтом і продавцем зазнав помітних змін.Це динамічне комунікаційне середовище змінило відносини клієнт-продавець, зробивши їх більш інтерактивними, прозорими та орієнтованими на клієнта.[2]

Однак це широке використання також вимагає винекненню потреб в постійному розвитку для просування торгівлі і комерції[3].Це вимагає постійної еволюції та вдосконалення галузі не лише для оптимізації існуючих процесів, але й для дослідження нових шляхів, які сприяють зростанню та розширенню торгівлі.[3]Отже, це підкреслює необхідність постійних досліджень і розробок у цій конкретній галузі, що слугує каталізатором розвитку торгівлі.

В якості предмету цього дослідження було обрано досить об'ємну систему взаємодії користувачів, яка охоплює отримання пропозицій від певних користувачів і пошук відповідних пропозицій від інших.

Крім того більшість подібних систем повинні включати можливість пошуку бажаних пропозицій, геопозиціювання пропозицій і дотримання торговельних узгоджень, що в свою чергу вимагають надійних заходів авторизації та автентифікації, для забезпечення доступу, а також запобіжних заходів для несанкційного доступу.

Нижче ви знайдете технічне завдання з детальним описом до проєкту. Цей звіт зосереджується на кліенській частині, який стосується мобільної програми, призначеної для користувачів Android систем. Звіт складається з трьох розділів: загальна характеристика системи, деталі створення клієнтської частини та аналіз результатів впровадження та тестування. Висновки містять колективні результати, задокументовані усіма учасниками проєкту.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЄКТУ

Загальний опис.
Програмне забезпечення (ПЗ) призначене для створення інформаційної системи управління селекційним контентом. Головна функція ПЗ полягає в наданні користувачу можливості замовити доставку продукції популярних закладів свого міста. Додаткова функція ПЗ має дозволити залишати відгуки та оцінки щодо якості доставки і продукції закладів.

Призначення:
ПЗ орієнтується на різні види закладів за видом послуг: ресторани, кафе, магазини тощо. ПЗ створюється за прикладом сервісу доставки, водночас має бути якомога якіснішим, щоб задовольнити потреби користувача.

Вимоги до функціоналу:
Реєстрація користувачів:
Для автентифікації користувачу потрібно створити обліковий запис. Має бути можливість це зробити як за допомогою авторизаційних даних (номера телефону/паролю), так і за допомогою облікових записів популярних соціальних мереж (Google, Facebook, тощо). Кожному користувачу потрібно вказати своє ім’я та/або прізвище для підтвердження своєї особистості кур’єру та спілкування з персоналом за потребою.

Платформа для розміщення закладів:
Основною функцією ПЗ має бути розміщення та перегляд більшості популярних закладів свого міста для кожного користувача. Має забезпечити усі необхідні засоби для

- перегляду асортименту послуг
- замовлення послуги закладу
- складання рейтингу на основі оцінок усіх користувачів

Асортимент товарів:
Кожен заклад, розташований на платформі, повинен мати свій асортимент контенту(товарів/послуг). ПЗ має забезпечити можливість створювати, видаляти та повертати категорії, підкатегорії послуг, робити вибірку по категоріях.

Онлайн кошик для покупок:
Для зберігання обраних послуг/товарів ПЗ повинне для кожного закладу реалізувати функції кошика:

- додавання товару
- видалення товару
- підрахунок загальної вартості всіх доданих товарів

Кошик для конкретного закладу має зберігати інформацію при перегляді ПЗ на різних пристроях (веб-додаток та мобільний додаток).

Оформлення замовлення:
Після додавання товарів у кошик користувачу надається можливість замовити послугу доставки за вказаною адресою. ПЗ має розрахувати повну вартість з урахуванням вартості доставки від окремого закладу до адреси користувача.

Зворотній зв'язок
Після завершення замовлення і успішного надання послуги доставки користувачу пропонується надати відгук та скласти оцінку якості роботи кур'єрів для покращення роботи сервісу. Оцінки та відгуки мають відображатися через історію попередніх замовлень.

Архітектура
ПЗ має бути спроєктоване за клієнт-серверною архітектурою з розподіленими вузлами:
Backend - серверне програмне забезпечення, що забезпечує збереження та оброблення поточних даних, взаємодію з базою даних. Надає програмний інтерфейс взаємодії (API) з іншими вузлами.
Web App - додаток, доступний з веб-простору (сайт), що забезпечує візуальний інтерфейс користувачів усіх категорій.
Mobile App - додаток, що встановлюється на мобільних пристроях (смартфон, планшет), додатково до Web App забезпечує моніторинг замовлень у фоновому режимі, дозволяє накопичувати дані при відсутності мережного з'єднання (офлайн) з подальшою синхронізацією з Backend.

Технології
Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контроля версій (Git, за узгодженням групи проєкту). Структура файлів та каталогів повинна відбивати архітектуру ПЗ :

- Backend,
- Web,
- Mobile

Крім того, кожен із модулів повинен розміщуватися в окремому репозиторії.

Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології створення:

Backend - JS, TS або JS фреймворки, зокрема Nest JS, Express.js
Web - JS, HTML, CSS,або JS фреймворки, зокрема ReactJS, AngularJS
Mobile - Java, або Kotlin, або Flutter

Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU http://www.gnu.org/prep/standards/, у тому числі з додержанням вимог до коментування та документування коду.

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

Система розроблена згідно з технічним завданням (ТЗ) і реалізована з використанням розподіленої архітектури клієнт-сервер. У системі було відокремлено окремі архітектурні компоненти, а саме:

- «Mockup»
- «Frontend»
- «Backend»

Графічна частина «Mockup» являє собою набір моделей та описів поведінки користувача. Модель подає веб-інтерфейси та мобільні інтерфейси взаємодії користувачів різного типу з інформаційною системою. При створенні інтерфейсів застосовано алгоритми розробки карти застосунку, прототипування, створення UI китів, принципи із підходами UI/UX, висновки колористики та ергономіки мобільних пристроїв. Частина розміщена за адресою https://www.figma.com/file/9xXtbafRd6wiTVc0YZaZEK/HERMES-Delivery?type=design&node-id=0-1&mode=design&t=R7JnLO2P93lMPhHZ-0

Серверна частина «Backend» реалізована як централізований вузол збереження та оброблення даних, що створюються та модифікуються протягом життєвого циклу інформаційної системи. Дана частина включає у себе віртуальний приватний сервер (VPS), базу даних та програмний інтерфейс (API) доступу до основних функцій бізнес-логіки, зокрема, автентифікація, перегляд контенту закладів, категорій та товарів, управління онлайн-кошиком, оформлення замовлення.

Частина «Backend» розміщена на віртуальному сервері Amazon Linux, виконана на базі програмного забезпечення Node.js та СУБД MongoDB. Додатково використані платформа для створення серверних додатків NestJS та програмні пакети Express, Mongoose.

Вихідний код частини розміщено у репозиторії https://github.com/hermes-delivery-app/hermes_delivery_android_version. Подальший звіт присвячений детальному опису цієї частини.

Частина «Frontend» є шлюзом для доступу користувачів до функціональних можливостей системи через візуальний інтерфейс. Ця система не тільки забезпечує безперебійну взаємодію, але й містить функції для локального зберігання даних, що дозволяє частково використовувати систему за відсутності мережі передачі даних.

Вихідні коди та ресурси даної частини розміщені на репозиторії https://github.com/hermes-delivery-app/hermes-backend-api.git. Подальший звіт присвячений детальному опису цієї частини.

Компонент інтерфейсу зв’язується з сервером для отримання даних, надсилання запитів і представлення інформації користувачам у зручний спосіб.
Розподілена архітектура клієнт-сервер системи GLOVO забезпечує ефективний зв’язок і обмін даними між різними архітектурними частинами. Макет визначає візуальне представлення системи, у той час як серверна частина обслуговує основні функції системи, а зовнішня частина забезпечує безперебійну роботу користувача.

РОЗДІЛ 2

РЕАЛІЗАЦІЯ СЕРВЕРНОЇ ЧАСТИНИ

Клієнтська частина мобільного додатку реалізована з використанням наступних технологій:

- Операційна система: Android, гарантується підтримка систем з версіями API від 23 до 33 (поточна на час початку створення проєкту).
- Засоби розроблення: мова програмування Java (OpenJDK 64-Bit Server VM by Oracle Corporation) та середовище Android Studio

Клієнтська частина мобільного додатку пропонує та включає в себе допоміжний функціонал, такий як:

- Реєстрація та авторизація користувачів
- Огляд і пошук запропонованих продуктів
- Можливість налаштовувати свої замовлення
- Можливість залишати і дивитись оцінку закладів
- Керування особистою інформацією у профілі

Після початку взаємодії з додатком, користувачі повинні завершити процес реєстрації, таким чином стаючи клієнтом інформаційної системи.

Під час реєстрації користувачам пропонується вказати номер телефону, який слугує засобами авторизації, отримання сповіщень щодо відповідних оновлень і змін у їх особистому прості. Для забезпечення точності даних введена інформація проходить попередню перевідку на серверній частині додатку для забезпечення гнучкості системи:

```
private void PostOKHttp(){
    String string_name = nameET.getText().toString();
    String string_phone_number = phoneET.getText().toString();
    String string_password = passwordET.getText().toString();

    if( string_name.isEmpty() ) {
        Toast.makeText( context: this, text: "Enter name", Toast.LENGTH_SHORT ).show();
        nameET.requestFocus();
        return;
    }
    new *
    new Thread(new Runnable() {
        new *
        @Override
        public void run() {
            RequestBody postBody = new FormBody.Builder()
                    .add( name: "name", string_name)
                    .add( name: "password", string_password)
                    .add( name: "phoneNumber", string_phone_number)
                    .build();
            Request request = new Request.Builder()
                    .url(url)
                    .post(postBody)
                    .build();
            OkHttpClient client = new OkHttpClient();
            OkHttpClient.Builder builder = new OkHttpClient.Builder();
            client = builder.build();

            try(okhttp3.Response response = client.newCall(request).execute()){
                //response = call.execute();
```

```
            try(okhttp3.Response response = client.newCall(request).execute()){
                String serverResponse = response.body().toString();
                new *
                runOnUiThread(new Runnable() {
                    new *
                    @Override
                    public void run() {
                        testTV.setText(serverResponse);
                    }
                });
            }catch(IOException error){
                error.printStackTrace();
            }
        }
    }).start();
}
1 usage   • Renesiat
```

Основним компонентом взаємодії користувача в програмі є робота з даними на стороні сервера. Взаємодіючи з інтерфейсом, користувачі можуть отримувати доступ і досліджувати інформацію, яка їх цікавить.

```
Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(url)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
myApi = retrofit.create(MyApi.class);
Call<JSONResponse> call = myApi.itemCall();

 Renesiat
call.enqueue(new Callback<JSONResponse>() {
     Renesiat
    @Override
    public void onResponse(Call<JSONResponse> call, retrofit2.Response<JSONResponse> response) {
        JSONResponse jsonResponse = response.body();
        items = new ArrayList<>(Arrays.asList(jsonResponse.getItems()));

        PutDataIntoRecyclerView(items);
    }

     Renesiat
    @Override
    public void onFailure(Call<JSONResponse> call, Throwable t) {

    }
});
```

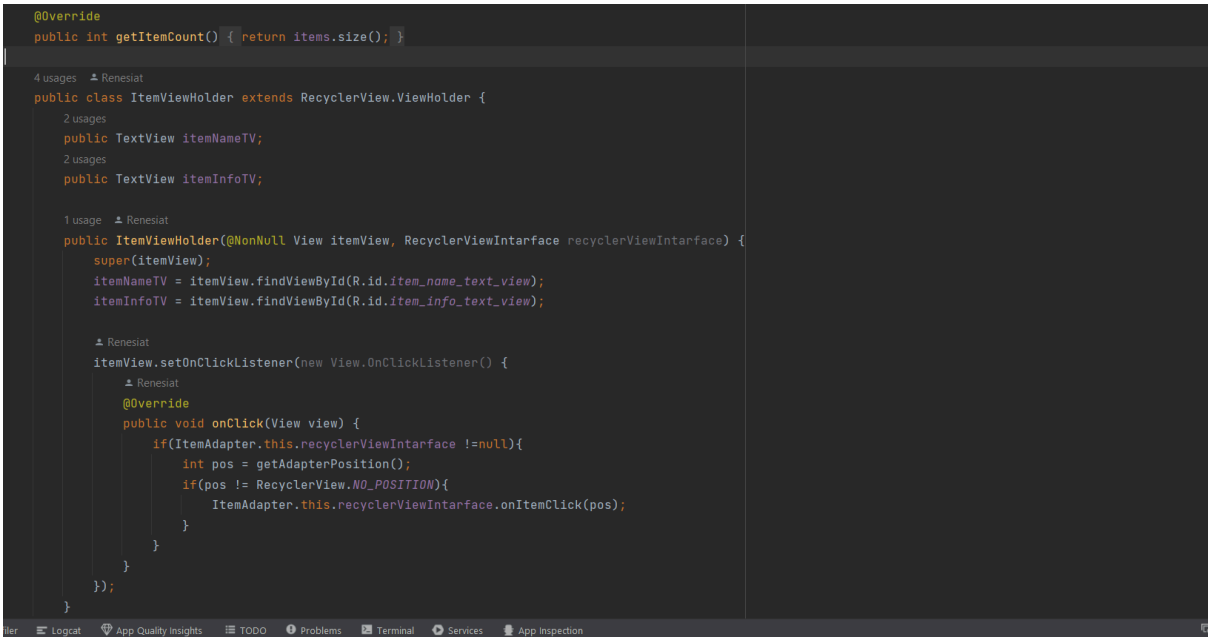
Після отримання відповіді від сервера, дані зберігаються в програмі і передаються в RecyclerView за допомогою адаптерів

```
public class ItemAdapter  extends RecyclerView.Adapter<ItemAdapter.ItemViewHolder> {
    4 usages
    private final RecyclerViewIntarface recyclerViewIntarface;
    2 usages
    Context context;
    4 usages
    List<Item> items;

    1 usage  Renesiat
    public ItemAdapter(RecyclerViewIntarface recyclerViewIntarface, Context context, List<Item> items) {
        this.recyclerViewIntarface = recyclerViewIntarface;
        this.context=context;
        this.items = items;
    }

    Renesiat
    @NonNull
    @Override
    public ItemViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        return new ItemViewHolder(LayoutInflater.from(context).inflate(R.layout.item_view,parent, attachToRoot: false), recyclerViewIntarface);
    }

    Renesiat
    @Override
    public void onBindViewHolder(@NonNull ItemViewHolder holder, int position) {
        holder.itemNameTV.setText(items.get(position).getName());
        holder.itemInfoTV.setText(items.get(position).getDescription());
    }
```



```
    @Override
    public int getItemCount() { return items.size(); }

    4 usages  Renesiat
    public class ItemViewHolder extends RecyclerView.ViewHolder {
        2 usages
        public TextView itemNameTV;
        2 usages
        public TextView itemInfoTV;

        1 usage  Renesiat
        public ItemViewHolder(@NonNull View itemView, RecyclerViewIntarface recyclerViewIntarface) {
            super(itemView);
            itemNameTV = itemView.findViewById(R.id.item_name_text_view);
            itemInfoTV = itemView.findViewById(R.id.item_info_text_view);

            Renesiat
            itemView.setOnClickListener(new View.OnClickListener() {
                Renesiat
                @Override
                public void onClick(View view) {
                    if(ItemAdapter.this.recyclerViewIntarface !=null){
                        int pos = getAdapterPosition();
                        if(pos != RecyclerView.NO_POSITION){
                            ItemAdapter.this.recyclerViewIntarface.onItemClick(pos);
                        }
                    }
                }
            });
        }
```

```
iler  Logcat  App Quality Insights  TODO  Problems  Terminal  Services  App Inspection
```

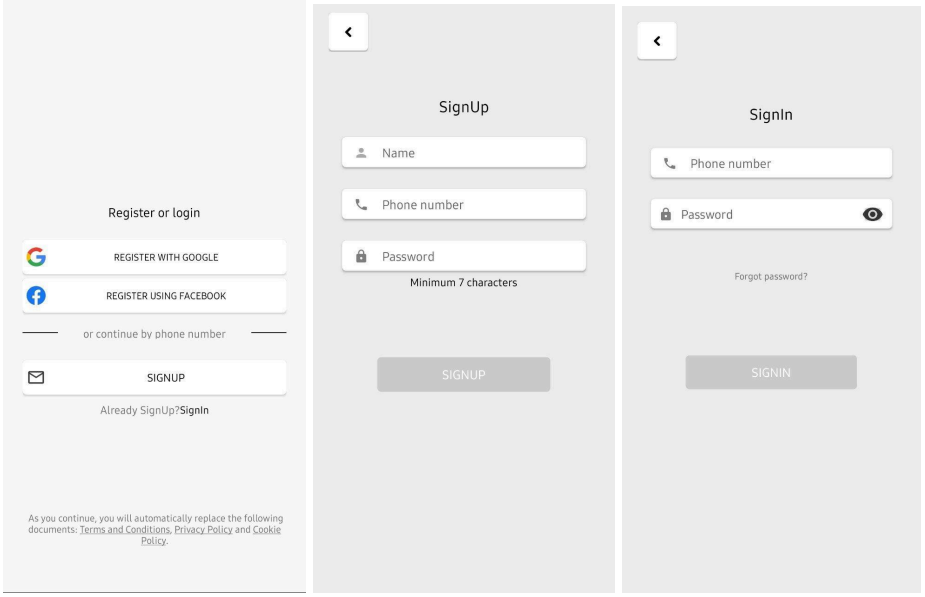
Як було зазначено з повним кодом проєкту можна познайомитись за посиланням: https://github.com/hermes-delivery-app/hermes_delivery_android_version

РОЗДІЛ 3
РЕЗУЛЬТАТИ ВИПРОБУВАНЬ

Програмний код клієнтської частини компілюється для створення інсталяційного файлу програми, який можна встановити на смартфоні або планшеті на базі Android. Основна серія тестів, як зазначено у звіті нижче, проводилася на смартфоні Samsung SM-A725F.

Початкова взаємодія користувача з програмою передбачає сторінку з варіаціями входу в систему, якщо новий користувач входить в систему, для початку, йому треба пройти реєстрацію в системі.

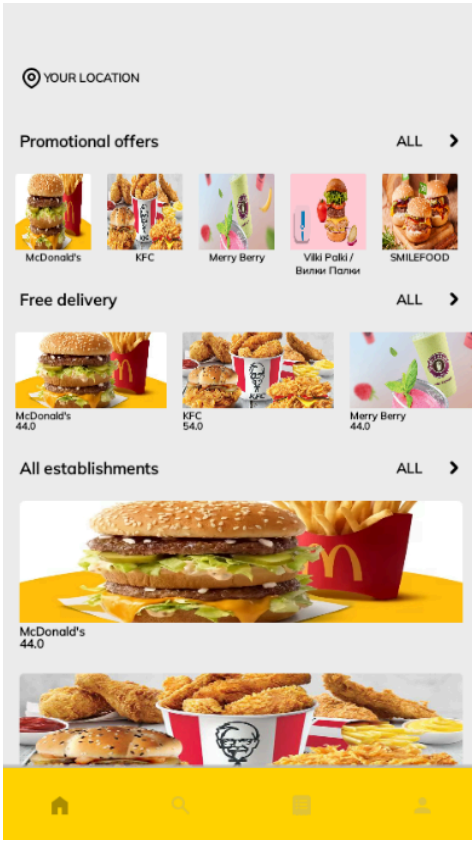
Для постійних користувачів доступний вхід за допомогою особистих даних, наданих під час реєстрації.



На рисунках зображена сторінка при запуску, регестрація і вхід в систему

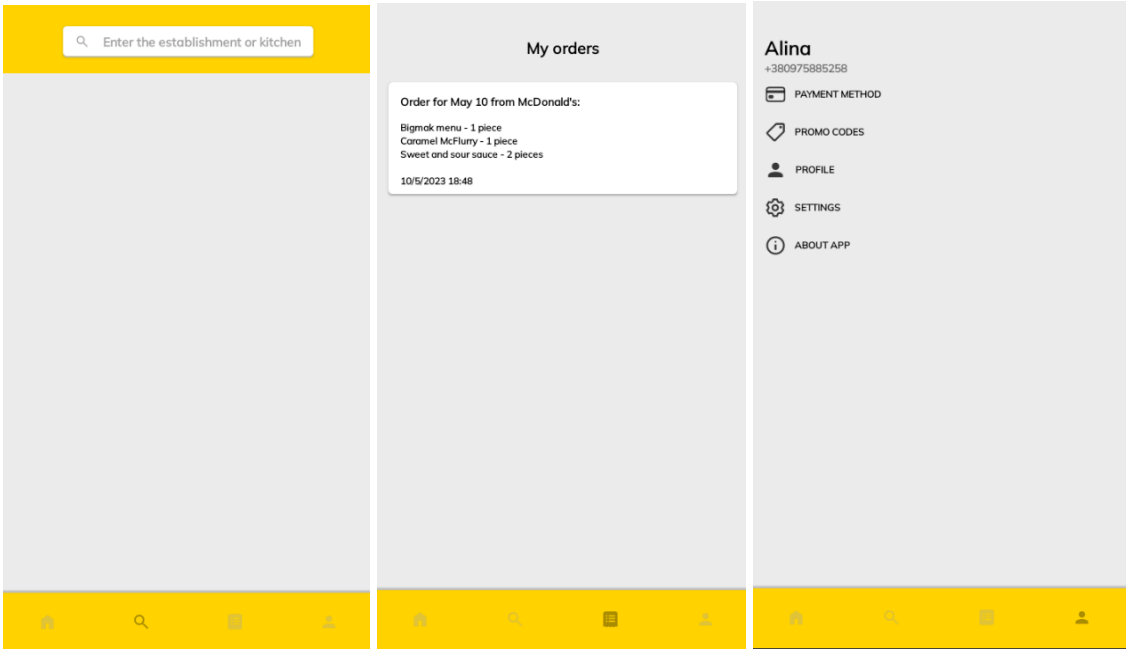
Після успішного входу в систему користувач отримує доступ до основних функцій системи. До основної сторінки програми на яких він може бачити і взаємодіяти з представленими послугами, такими як:

- Перелік магазинів які є в додатку
- Магазини, що надають скидки або безкоштовну доставку
- Перейти на інші сторінки в меню розташованому знизу

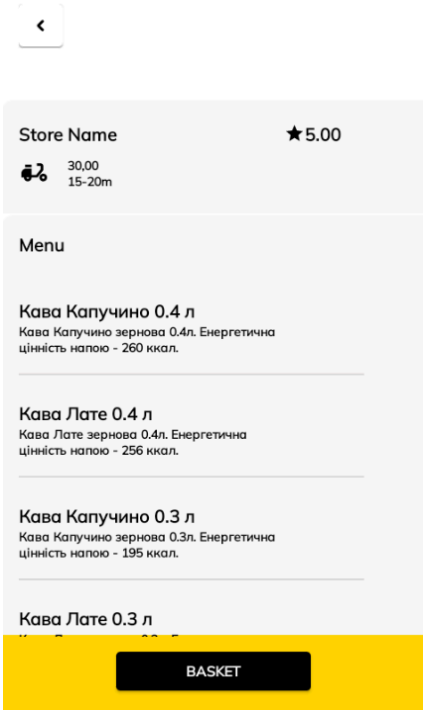


В меню наведеному нижче ми можемо перейти на:

- Пошук магазинів в додатку
- Оплату замовлення
- Особистий кабінет користувача



При виборі магазину ми маємо змогу перейти до вибору продуктів і додавання їх в кошик:



Поточну ітерацію програми, що включає клієнтський компонент, можна отримати з репозиторію, розташованого за посиланням:
https://github.com/hermes-delivery-app/hermes_delivery_android_version

ВИСНОВКИ

Після проведення обширного аналізу інформаційних систем сконцентрованих на комунікації продавець-покупець стало очевидно, що на ринку переважають системи, спеціально розроблені для підтримки торгових послуг. Ці системи відіграють вирішальну роль у зміцненні торгових відносин між користувачами. Як яскравий приклад для реалізації було обрано застосунок служби доставки.

Впроваджена система управління селекційним контентом містить механізм координації, який дозволяє працювати з нею. Відповідно до розподіленої клієнт-серверної архітектури система складається з трьох окремих компонентів, таких як “Mockup”, “Frontend” і “Backend”

Система пройшла тестування, включаючи встановлення на фізичних пристроях і численні сеанси зв’язку. Результати цих випробувань підтвердили загальну ефективність системи та продемонстрували її придатність для сприяння безперебійній торговій взаємодії між користувачами.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “Android Programming: The Big Nerd Ranch Guide”; Philips B.,Stewart C., 2012p.
2. “Android Programing: Pushing the Limits”; Hellman E., 2013p.
3. “Efficient Android Threading: Asynchronous Processing Technique for Android Applications”; Goransson A., 2014p.
4. Документація Android Studio (https://developer.android.com/docs)
5. Документація бібліотеки Okhttp(https://square.github.io/okhttp/)
6. Документація бібліотеки Retrofit(https://square.github.io/retrofit/)