МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Інформаційна система обліку замовлень пластикових вікон»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ.м-01 Большунов Денис Михайлович

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ «__» грудня 2021 р.

Науковий керівник _____ к.т.н., доц., Баранова І.В.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д. М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Суми-2021

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. кафедрою ІТ

_____В. В. Шендрик
«__» _____2021 р.

З А В Д А Н Н Я

на кваліфікаційну роботу магістра студентові

Большунов Денис Михайлович

1 Тема роботи *Інформаційна система обліку замовлень пластикових вікон*

керівник роботи *Баранова Ірина Володимирівна, к.т.н., доцент***,**

затверджені наказом по університету від «29» жовтня 2021 р. № 0787-VI

2 Термін здачі студентом закінченого проекту «___» грудня 2021 р.

3. Вхідні дані до проекту технічне завдання на розробку

4 Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) аналіз предметної області додатків для обліку замовлень, постановка задачі та методи дослідження, проектування інформаційної системи обліку замовлень пластикових вікон, реалізація інформаційної системи обліку замовлень пластикових вікон

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність роботи, мета та задачі, аналіз аналогів, функціональні вимоги, засоби реалізації, структурно-функціональне моделювання роботи додатку, проектування бази даних, схема додатку, практична реалізація, висновки.

6 Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

Дата видачі завдання _____.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення календарного плану робіт	01.09.2021 – 10.09.2021	
2	Визначення мети проекту	11.09.2021 – 15.09.2021	
3	Огляд аналогічних рішень	15.09.2021 – 20.09.2021	
4	Визначення функціональних вимог до інформаційної системи	21.09.2021 – 30.09.2021	
5	Визначення засобів реалізації	01.10. 2021 – 10.10.2021	
6	Моделювання та проектування інформаційної системи	11.10.2021 – 25.10.2021	
7	Реалізація інформаційної системи	26.10.2021 – 25.11.2021	
8	Тестування	26.11.2021 – 30.11.2021	
9	Складання документації	01.12.2021 – 02.12.2021	
10	Оформлення пояснювальної записки	03.12.2021 – 12.12.2021	
11	Здача інформаційної системи обліку замовлень пластикових вікон	До 15.12.2021	

Студент _____ Большунов Д.М.
(підпис)

Керівник роботи _____ к.т.н., доц. Баранова І.В.
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Інформаційна система обліку замовлень пластикових вікон».

Пояснювальна записка складається зі вступу, 4 розділів, висновків, списку використаних джерел із 38 найменувань, додатків. Загальний обсяг роботи – 100 сторінок, у тому числі 75 сторінок основного тексту, 3 сторінки списку використаних джерел, 19 сторінок додатків, 7 таблиць та 61 рисунок.

Кваліфікаційну роботу магістра присвячено розробці додатку, який полегшить роботу з опрацювання та аналізу замовлень пластикових вікон.

У першому розділі проведено аналіз предметної області. Було обґрунтовано актуальність розроблювального додатку, виконано аналіз останніх досліджень і публікацій, та описано огляд аналогічних рішень.

У другому розділі наведено постановку задачі та методи дослідження. Також описується вибір засобів реалізації для розробки додатку.

У третьому розділі описуються структурно-функціональне моделювання, проектування Use-Case діаграми та проектування бази даних.

Останній, четвертий розділ присвячено реалізації інформаційної системи обліку замовлень – описані складові етапи розробки додатку, наведено програмну реалізацію проєкту.

Результатом проведеної роботи є реалізована інформаційна система обліку замовлень пластикових вікон.

Практичним значенням інформаційної системи є скорочення обсягів ручної роботи, зниження ймовірності помилок, збільшення продуктивності праці співробітників інтернет-магазину.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, ІНТЕРНЕТ-МАГАЗИН, ОС ANDROID, JAVA, МОБІЛЬНИЙ ЗАСТОСУНОК, СТАТИСТИКА, ОБЛІК ЗАМОВЛЕНЬ, ДИЗАЙН, БАЗА ДАНИХ.

ЗМІСТ

ВСТУП

В даний час існує велика кількість програмних додатків, які реалізують інформаційні системи, що дозволяють працювати компаніям, підприємствам, інтернет-магазинам тощо з клієнтами і дозволяють супроводжувати замовлення на будь-якому етапі його виконання [1]. Однак більшість з них або незрозумілі для користувача, або використовують складні функції, або навпаки, багато їхніх функцій виявляються не потрібними при формуванні замовлень.

Найчастіше доводиться витрачати багато часу для того, щоб навчити персонал роботі з програмним додатком, що призводить до значних витрат часу. Все це говорить про необхідність створення інформаційної системи, яка б надавала можливість виконувати необхідні дії більш простим способом, особливо щодо обліку замовлень.

Для обліку замовлень зазвичай не потрібно великих та складних баз даних та програм, що працюють з ними. Буває досить невеликого інструментарію, що дозволяє виконувати основні операції із замовленням – такі як, наприклад: статус замовлення, відстеження виконання замовлення та інформація про нього, статистика про конкретне замовлення, формування невеликого звіту.

Розробка додатку ведеться для інтернет-магазину із продаж ПВХ вікон "СтеклоРама". Основний вид діяльності інтернет-магазину "СтеклоРама" – продаж ПВХ вікон [2].

В даний час облік замовлень ведеться вручну, тобто аналітика ведеться за допомогою співробітників інтернет-магазину, котрі виписують дані із БД, а також безпосередньо інтернет-магазину і таким чином складають звіт. Через те, що з плином часу інтернет-магазин побажав перейти на новий рівень з продажу товарів, було ухвалено рішення у створенні інформаційної системи для обліку замовлень.

За допомогою такої системи праця співробітників суттєво полегшиться, це значна та неоціненна допомога, адже одна людина або декілька людей не можуть тримати в пам’яті чи на паперових носіях дані, які в комп’ютері займають терабайти. Виходячи з вищевикладеної інформації, сформульовано необхідність у створенні програми процесу обліку замовлень для інтернет-магазину "СтеклоРама". Таким чином, тема магістерської роботи є актуальною.

При розробці програми враховано, що послідовність роботи з нею повинна відповідати наступному алгоритму: додаток отримує дані з існуючої бази даних, яку надає компанія, дані надходять у додаток на обробку за заданим запитом, який користувач сформував у процесі досягнення мети у використанні додатком, дані обробляються і виводиться результат користувачеві у вигляді графіка та невеликого звіту необхідного для розуміння побудованого графіка.

Метою магістерської роботи є створення інформаційної системи обліку замовлень пластикових вікон, що дозволить забезпечити легкість та швидкість роботи в обліку замовлень.

Об’єктом дослідження є інформаційні системи підтримки обліку замовлень.

Предметом дослідження є програмний додаток для реалізації інформаційної системи обліку замовлень пластикових вікон.

Завданнями магістерської роботи є:

- аналіз схожих рішень;
- обґрунтування значущості та актуальності обраної теми розробки у даній предметній області;
- аналіз можливих шляхів та способів проектування рішення поставлених завдань;
- створення інформаційної системи обліку замовлень пластикових вікон на базі ОС Android [3];
- впровадження в експлуатацію, а також подальший супровід.

Ґрунтуючись на цих потребах, було ухвалено рішення розробки мобільного додатка, оскільки саме в такому вигляді система буде максимально зручною для кінцевого користувача та зможе відповідати всім поставленим вимогам.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗРОБКИ ДОДАТКІВ ДЛЯ ОБЛІКУ ЗАМОВЛЕНЬ

1.1 Обґрунтування актуальності роботи

Одним з першочергових завдань, що вирішуються в рамках інтернет-магазину, є функція обліку замовлень. На сьогодні існує величезний вибір програмних продуктів, котрі надають такі функції для обліку замовлень. Проте більшість подібних рішень орієнтовані на широке коло споживачів, що робить ці системи непридатними для ведення обліку замовлень зі специфічними потребами. .Інформаційна система обліку замовлень пластикових вікон – це додаток на базі ОС Android, який націлений на ведення обліку у інтернет-магазині «СтеклоРама». Перевагою даного вибору є те, що додаток не потребуватиме його доопрацювання на стадії експлуатації тому, що додаток розробляється для певної організації з її специфічними потребами, для ведення обліку замовлень зокрема.

Під специфічними потребами розуміється наступне:

- Інтуїтивно зрозумілий інтерфейс;
- Швидке навчання користуванню застосунком;
- Візуалізація даних у в графічному вигляді для більш легкого сприйняття відображених результатів;
- Немає потреби переплачувати за зайвий функціонал;
- Додаток розрахований виключно на вузьке коло фахівців;
- Вхід в додаток потребує спеціального паролю;
- Підтримка додатку на стадії експлуатації якщо така знадобиться.

Актуальність теми полягає в тому, що ведення обліку замовлень дозволяє працівникам інтернет-магазину мінімізувати часові витрати на

проведення різних маніпуляцій з відображенням даних для більш точного прийняття рішень у тій чи іншій ситуації.

Це означає те, що співробітнику, котрий має доступ до використання додатком, буде легше обрати необхідний товар в застосунку, щоб швидко переглянути статус замовлення.

1.2 Аналіз останніх досліджень

Використання баз даних та інформаційних систем стає невід'ємною складовою ділової діяльності сучасної людини та функціонування успішних організацій. У зв'язку з цим велику актуальність набуває освоєння принципу побудови та ефективного застосування відповідних технологій та програмних продуктів. Сучасні інформаційні системи, що реалізують інтеграцію даних, характеризуються величезними обсягами даних, складною організацією, необхідністю задовольняти різноманітні вимоги численних користувачів [4].

Розробці систем управління та обліку на підприємствах присвячено багато публікацій. Зокрема у статті [5] показано, що автоматизації процесів обліку та супроводу замовлень компанії з виробництва друкованої продукції можна легко досягти шляхом розробки автоматизованої інформаційної системи.

З іншої сторони, ефективне управління підприємством в сучасних умовах неможливо без використання комп'ютерних технологій. Тому створення інформаційної системи, що дозволяє ефективно проводити статистичні дослідження та виявляти залежності результативних показників ефективності від продуктивних факторів, відіграє важливу роль в управлінні підприємством і при вирішенні різних виробничих завдань [6].

Актуальність теми дослідження пояснюється тим, що в 20 столітті відбулася зміна, яка надалі відіграла велику роль в історії Землі: індустріальний етап змінився етапом, який багато вчених називають

інформаційною революцією. І разом з цим професійне управління, вміння координувати ефективну роботу персоналу, правильність при проектуванні, реалізації та вдосконаленні бізнес-процесів, ефективне проведення організаційно-адміністративної та господарської діяльності стають основними критеріями при досягненні успіху в бізнесі.

В подібних умовах сучасні інформаційні технології, а також інтегровані інформаційні системи, що формувалися на їх основі, набувають значущості, перетворюючись на незамінні інструменти, які стають необхідними за сталого розвитку підприємств, а також організацій і в забезпеченні досягнення стратегічних цілей [7].

Мобільні додатки все більше проникають у корпоративне середовище, забезпечуючи нові конкурентні переваги компаніям. У 2020 році кількість активних користувачів інтернету досягла позначки 4,5 млрд. осіб, з яких понад 3 млрд. використовують для доступу мобільні пристрої. Наприклад, для міленіалів мобільність бізнесу – один із ключових моментів, на який вони звертають увагу при виборі роботи. І до кінця 2020 року, згідно з прогнозами, 70% доступу до корпоративних систем здійснюватиметься завдяки портативним пристроям. Пандемія лише прискорила цей процес [8].

Можливості впливу мобільних технологій на роботу підприємства невичерпні. Можна говорити про працівників, які використовують власні пристрої, та вплив цього процесу на бюджети комп’ютерних відділів та капітальні витрати. Можна уявити нові можливості для виробників, наприклад, постачальників споживчих товарів, які, враховуючи їх методи дистрибуції, традиційно були відрізані від кінцевих покупців, тепер вони можуть пропонувати знижки покупцям прямо в точках продажу. І це лише початок. Кадри, продажі, фінанси та обслуговування – це лише деякі області, в яких настане мобільна революція [9].

Технології не стоять на місці, вони постійно розвиваються і особливо це стосується мобільного світу. Всі ми помічали людей, які не можуть відірватися від мобільного телефону, і часто не має значення, сидять вони у кафе з

друзями, перебувають у родинному колі або їдуть у метро. Це відбувається тому, що мобільний телефон дає нам багато можливостей та користі. Винайдений порівняно недавно, він вже щільно інтегрувався в наше життя. Зараз без наших смартфонів ми й не впораємось. Точніше, нам довелося б без них дуже туго [10].

Бізнес існує з доісторичних часів. Якщо вірити підручникам з історії, це почалося ще з бартерної торгівлі, але з того часу вона перетворилася на щось набагато складніше, і без технологій це було б неможливо. Основні галузі світу впали б, якби раптово бізнес відібрав інформаційні технології. Це тому, що більшість бізнес-операцій не можуть проводитися у 21 столітті без використання сучасних технологій [11].

1.3 Огляд аналогічних рішень

Одна із основних вимог при створенні інформаційної системи у вигляді програмного додатку полягає в тому, щоб створюваний додаток був ефективнішим за попередній аналог. Аналіз обраних рішень проводитиметься за наступними критеріями: ціна додатку (рішення), вимоги до обладнання, налаштування, навчання та наявність мобільного застосунку.

Нижче наведено опис аналогів п’яти рішень та їх характеристик (додаток, що розробляється, і 4 аналогічних рішення).

Розглянемо відомі застосунки, аналогічні додатку, що розробляється.

1. «РемОнлайн» – це онлайн сервіс, який поєднує функції CRM системи, системи обліку замовлень, системи управління фінансами і складський облік. До того ж, РемОнлайн має багато інших функцій, наприклад, може друкувати документи та звіти, швидко аналізувати показники компанії та приймати ефективні управлінські рішення [12].

- Ціна додатку: хоббі – від 9€/міс, стартап – від 19€/міс, бізнес – від 29€/міс;

- Вимоги до обладнання: браузер;
- Налаштування: описані на сайті та безпосередньо в додатку;
- Навчання: інструкція з використання;
- Мобільний застосунок: присутній, але призначення додатку не відноситься до розроблюваного.

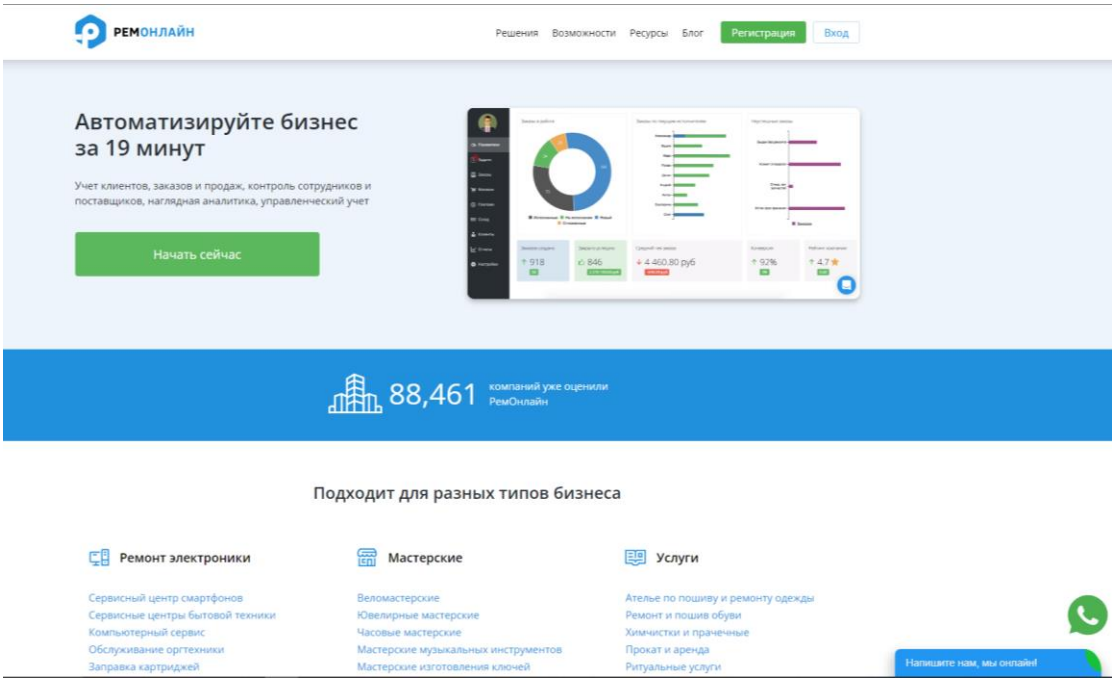

Рисунок 1.1 – Головна сторінка онлайн сервіса РемОнлайн

2. «МойСклад» – інтернет-сервіс для управління торгівлею, призначений для автоматизації малого та середнього бізнесу. Дозволяє керувати продажами та закупівлями, контролювати взаєморозрахунки, працювати з клієнтською базою, вести складський облік, а також друкувати всі необхідні для ведення бізнесу документи [13].

- Ціна додатку: безкоштовний – 0 рублів (урізаний варіант), старт – 450 рублів, базовий – 850 - 1000 рублів, профі – 2465 - 2900 рублів, корпоративний – 5965 - 6900 рублів;
- Вимоги до обладнання: браузер;
- Налаштування: описані на сайті;
- Навчання: інструкція з використання;

- Мобільний застосунок: присутній, але після завантаження в додаток без дзвінка служби підтримки не потрапити.

На рисунку 1.2 показано головну сторінку сайту, на рисунку 1.3 – мобільний застосунок.

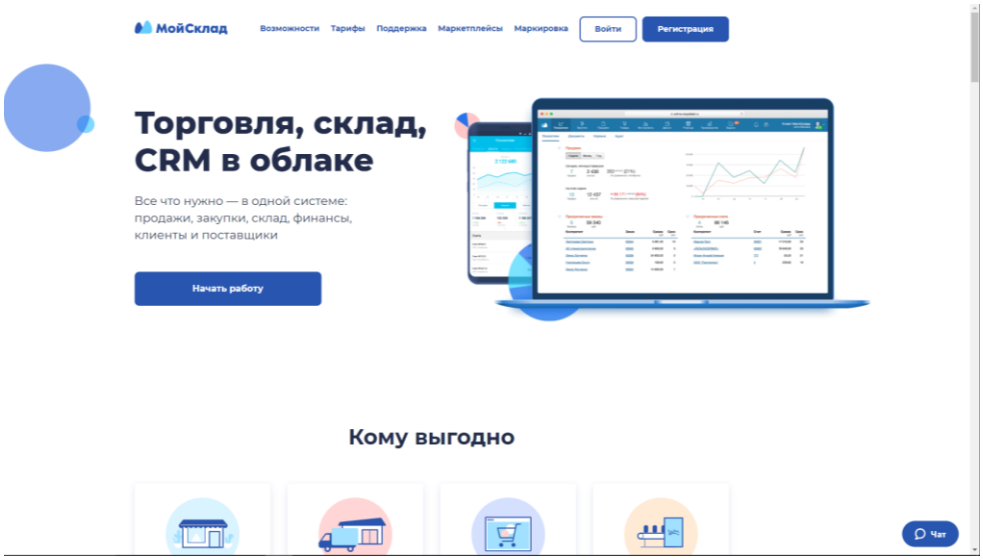

Рисунок 1.2 – Головна сторінка сайту МойСклад

Рисунок 1.3 – Мобільний застосунок МойСклад

3. «CRM Qsystem» – це система для роботи в малому або середньому інтернет-магазині будь-якого профілю. Дозволяє автоматизувати всі процеси оперативної роботи менеджера, працювати віддалено, стежити за продажами. Впровадження CRM зробить процес виконання замовлення максимально коротким за часом і за діями: отримав замовлення – підтвердив – зареєстрував – відправив, більшість дій здійснюється в один клік миші. Менеджер може не тільки стежити за роботою підлеглих, а й отримувати статистичну та фінансову інформацію про роботу інтернет-магазину за будь-який період часу. Співпраця з постачальниками також максимально спрощується. CRM автоматично проінформує клієнтів про виконані замовлення, або донесе іншу інформацію за допомогою SMS та email оповіщення [14].

- Ціна додатку: базовий – 300 грн/місяць, все включено – 500 грн/місяць;
- Вимоги до обладнання: мінімальні вимоги: Celeron 2,6ГГц 2Гб DDR3;
- Налаштування: описані на сайті та безпосередньо в настільному додатку;
- Навчання: інструкція з використання;
- Мобільний застосунок: відсутній.

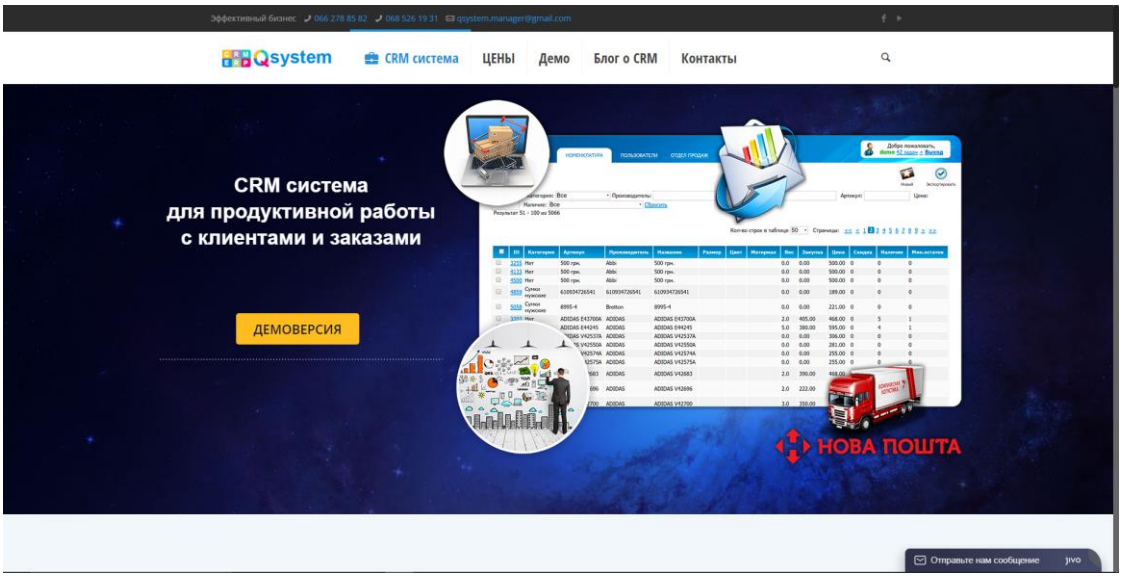

Рисунок 1.4 – Головна сторінка сайту CRM Qsystem

4. «Класс365» – онлайн-система для автоматизації бізнесу. Програма дозволяє в одному місці керувати торгівлею, складом, клієнтами та фінансами. Основне завдання Класс365 – дати можливість підприємцям керувати всіма бізнес-процесами малого бізнесу в одному місці та при цьому забезпечити прийнятну ціну. Система представлена як веб-сервіс та працює через інтернет. Можна керувати бізнесом у будь-якому місці, де є інтернет [15].

- Ціна додатку: безкоштовно – 0 грн/міс, початковий – 490 грн/міс, оптимальний – 790 грн/міс;
- Вимоги до обладнання: браузер та мінімальні вимоги для запуску сучасних мобільних додатків;
- Налаштування: описані на сайті та безпосередньо в додатку;
- Навчання: інструкція з використання;
- Мобільний застосунок: присутній.

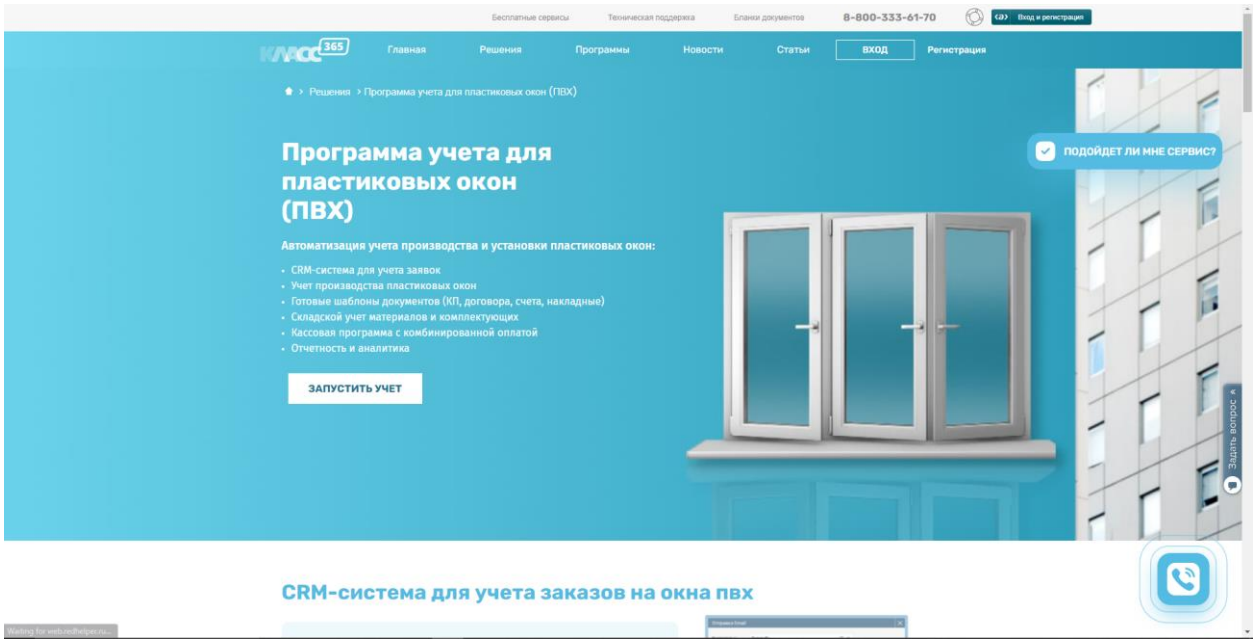

Рисунок 1.5 - Головна сторінка сайту Класс365

5. Розроблюваний застосунок «СтеклоРама» – це реалізація інформаційної системи у формі мобільного застосунку, призначення якої – ведення обліку замовлень пластикових вікон.

- Ціна додатку: відсутня;

- Вимоги до обладнання: мінімальні вимоги для запуску сучасних додатків;
- Налаштування: безпосередньо в додатку;
- Навчання: інструкція з використання;
- Мобільний застосунок: присутній.

Результати аналізу розглянутих застосунків представлено у таблиці 1.1.

Аналіз розглянутих рішень свідчить про те, що розроблюваний додаток не поступається аналогам за показниками, і навіть перевершує за деякими з них. Наприклад, ціна на продукцію у деяких рішень немаленька, враховуючи, що за використання того чи іншого додатка доведеться платити щомісяця, в свою чергу СтеклоРама не має абонентської плати.

Деякі організації не надають мобільних додатків, що в свою чергу, звичайно ж мінус, тому що браузер не завжди виступає універсальним варіантом для вирішення проблем. Незважаючи на те, що деякі сервіси надають мобільні рішення, не всі користувачі залишають позитивні відгуки після використання.

Наприклад, інтернет-сервіс «МойСклад» в магазині додатків Play Market від компанії Google, отримав велику кількість негативних відгуків від користувачів [16-17]. В масі своїй відгуки відносились лише до несхожості мобільного застосунку із онлайн сервісом, що представлений на сайті.

Це означає те, що при розробці додатку, необхідно аналізувати подібні рішення на наявність невідповідності із основними рішеннями (сайти, настільні додатки), котрі пропонують сервіси, компанії, підприємства і так далі. Вимоги до апаратної частини загалом однакові до всіх додатків. Також варто звернути увагу, що практично всі додатки мають інструкцію з використання та опис налаштування додатку.

Таким чином можна зробити висновок, що проєктований додаток буде актуальним, затребуваним та в ньому слід уникати виявлених недоліків.

Таблиця 1.1 – Аналіз аналогічних рішень.

Назва додатку / Критерії	**РемОнлайн**	**МойСклад**	**CRM Qsystem**	**Класс365**	**СтеклоРама**
Ціна додатку	Від 9€/міс, стартап – від 19€/міс, бізнес – від 29€/міс	Безкоштовний – 0 р. (урізаний варіант), старт – 450 р., базовий – 850 - 1000 р., профі – 2465 - 2900 р., корпоративний – 5965 - 6900 р.	базовий – 300 грн/місяць, все включено – 500 грн/місяць	Безкоштовно – 0 грн/міс, початковий – 490 грн/міс, оптимальний – 790 грн/міс	Відсутня
Вимоги до обладнання	Браузер	Браузер	мінімальні вимоги: Celeron 2,6ГГц 2Гб DDR3	Браузер та мінімальні вимоги для запуску сучасних мобільних	Мінімальні вимоги для запуску сучасних додатків на базі ОС
Налаштування	Описані на сайті та безпосе-редньо в додатку	Описані на сайті	описані на сайті та безпосередньо в настільному додатку	описані на сайті та безпосередньо в додатку	Безпосередньо в додатку
Навчання	Інструкція з використання	Інструкція з використання	Інструкція з використання	Інструкція з використання	Інструкція з використання
Мобіль-ний застосу-нок	Присутній, але призначення додатку не відноситься до розроблюваного	Присутній, але після завантаження в додаток без дзвінка служби підтримки не потрапити	Відсутній	Відсутній	Присутній

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі дослідження

Метою дипломної роботи є розробка інформаційної системи у формі мобільного додатка для ведення обліку замовлень пластикових вікон.

Ціллю виконання роботи є можливість за допомогою додатку автоматизувати процес ведення обліку замовлень та формувати необхідні статистичні звіти у графічному та текстовому вигляді.

Для досягнення мети роботи необхідно виконати такі задачі:

- Проаналізувати подібні рішення. Отримати інформацію про переваги недоліки, спробувати виявити слабкі та сильні сторони і в подальшому впровадити отримані дані в додаток;
- Замінити всі можливі дії, котрі стосуються обліку замовлень, що виконуються руками на автоматизований процес;
- Користування додатком не повинно вимагати спеціальних навичок;
- Не всі співробітники компанії повинні мати доступ до застосунку;
- Розробити алгоритм роботи застосунку.

Були сформульовані наступні вимоги, при виконанні яких додаток вважатиметься мінімально життєздатним продуктом:

1. Додаток повинен мати функціонал інтерактивної взаємодії із контентом;
2. Додаток має бути розроблений під платформу Android 5.0+;
3. Додаток повинен відображати статистику товару(ів), який вибирає користувач;
4. Після обирання товару, надати функції задання певного проміжку часу для відображення статистики по заданому проміжку;
5. Розробити функціонал «Список товарів», щоб користувач мав змогу отримувати швидкий доступ до обраних товарів;

6. Реалізувати функціонал видалення товару із Списку товарів;
7. Реалізувати алгоритм пошуку товару;
8. Реалізувати алгоритм «Фільтр»;
9. Реалізувати функціонал реєстрації та авторизації користувача;
10. Співробітник повинен мати спеціальний пароль (пароль видає менеджер), за допомогою якого можна буде увійти в додаток.

2.2 Вибір засобів реалізації

Проект, який розробляється, передбачає, що користувач буде взаємодіяти з додатком для досягнення поставленого завдання. Завдання розробника полягає в створенні додатку, який містив би у собі всі необхідні функції, та виконував завдання, поставлені змовником. Щоб додаток коректно працював, спочатку потрібно підібрати необхідні засоби реалізації для його розробки, проектування та написання логіки застосунку, описані далі в цьому підрозділі буде описано всі засоби реалізації.

Перед початком створення того чи іншого проекту часто виникає потреба в розумінні його ієрархії. Існує досить велика кількість рішень, котрі надають інструменти для проектування так званого потоку функціоналу додатку.

В рамках розроблюваного проекту, під потоком функціоналу додатку розуміється наступне – легко або складно деталізована ієрархічна схема, яка допомагає виявити (відобразити) основні функціональні складові додатку.

Зокрема для даного проекту використовувалось програмне забезпечення для проектування діаграм draw.io [18]. Даний застосунок дуже простий у використанні, має досить легкий інтерфейс, що в свою чергу скорочує час на навчання використанням. Також застосунок надає досить велику кількість інструментів, для проектування діаграм різної складності. Хотілося б ще згадати, що додаток абсолютно безкоштовний та може працювати як в

браузері, так і на персональному комп'ютері. На рисунку 2.1 відображено головну сторінку застосунку.

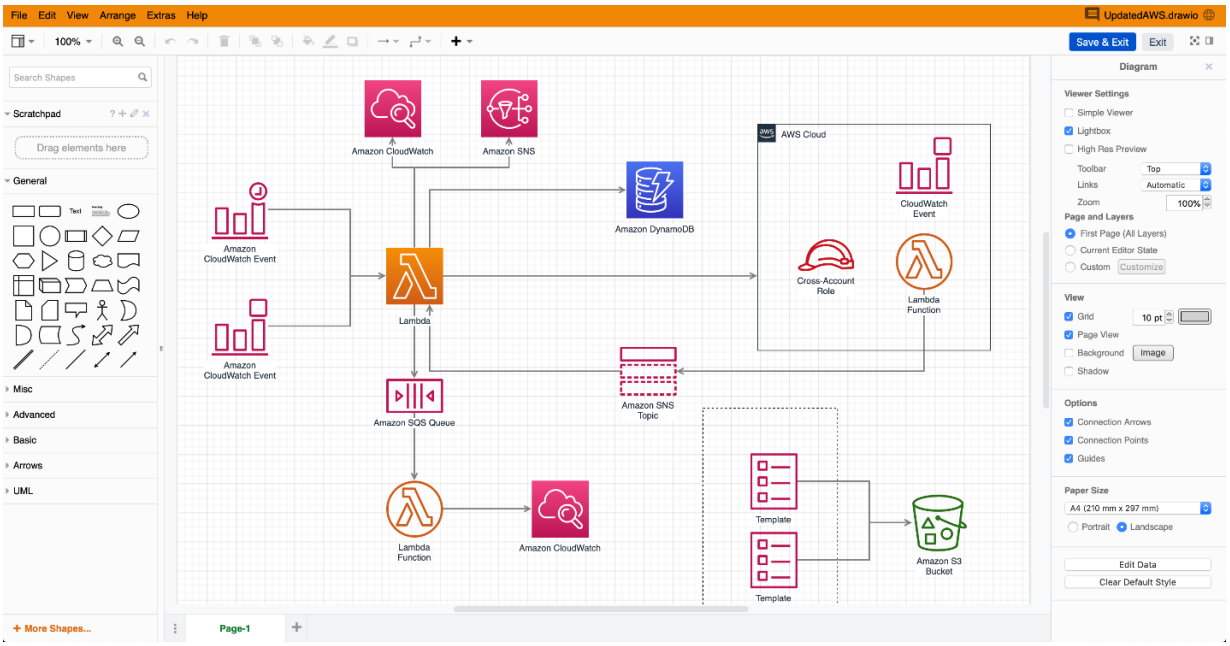

Рисунок 2.1 – Головна сторінка draw.io

Після завершення проектування необхідних для розробки схем, часто наступним кроком виступає створення дизайну. Даний проект не є виключенням. Під створенням (проектуванням) дизайну розуміється наступне – розробник із зібраних даних (часто в цьому плані виступає технічне завдання, в яке уже входять раніше розроблені схеми), займається створенням UI/UX дизайну [19]. Не можна сказати, що на сьогодні існує дуже велика кількість дійсно значимих рішень для UI/UX дизайну – їх не так вже й багато. Одним із найпопулярніших рішень є Figma [20].

Figma – це зручний графічний онлайн-сервіс із надання послуг для UI/UX проектування, за допомогою якого можна створювати складні інтерфейси мобільних застосунків та веб-сайтів. Також сервіс надає послуги із інтерактивного прототипування, що в свою чергу дозволяє за допомогою анімацій відобразити роботу чи то застосунку для мобільних пристроїв, чи веб-сайтів. Саме з використанням даного сервісу велось проектування

інформаційної системи обліку замовлень пластикових вікон. На рисунку 2.2 відображено головну сторінку сервісу.

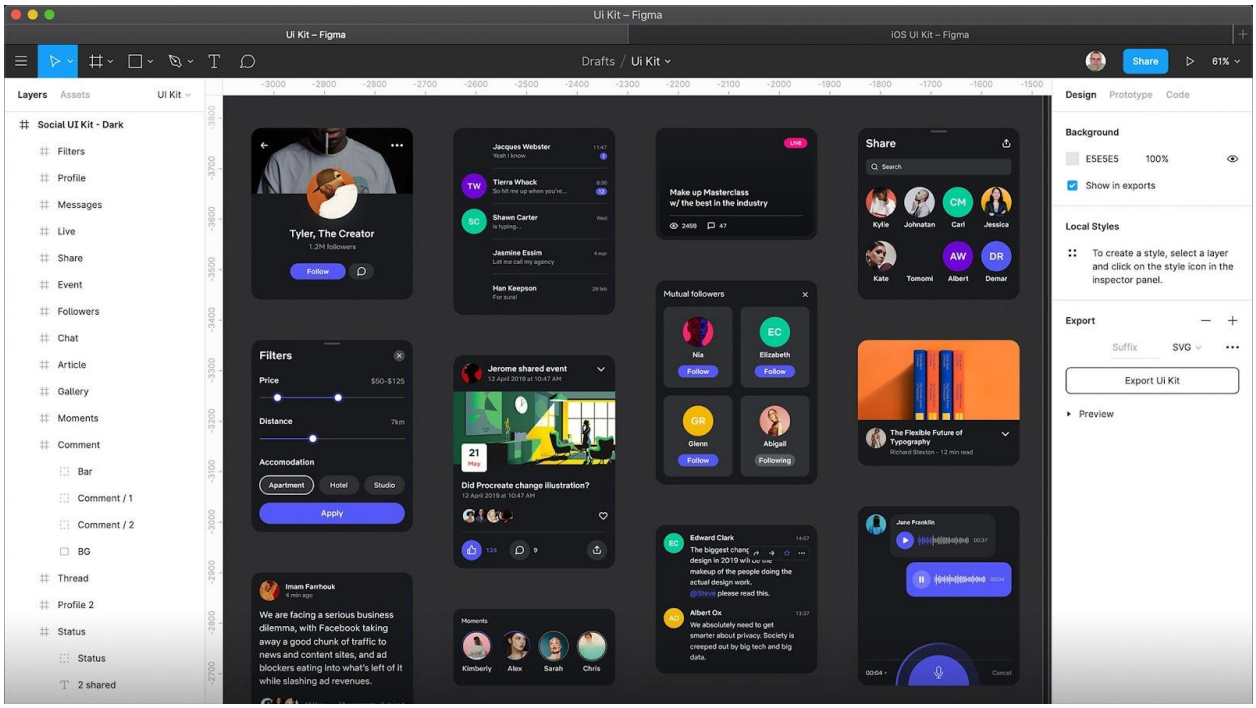

Рисунок 2.2 – Приклад використання Figma

Після створення дизайну настає черга написання логіки додатку. Перед початком написання коду необхідно обрати мову програмування. Існує безліч мов програмування, котрі пропонують свій інструментарій для створення додатків під операційну систему Android.

Незважаючи на велику кількість мов програмування, одною із основних, вже порівняно довгий час виступає – Java [21]. Мова програмування Java – це високорівнева, об’єктно-орієнтована мова програмування, котра пропонує дуже велику кількість вбудованих фреймворків, патернів, бібліотек для створення абсолютно різних за функціоналом застосунків, веб-сайтів, веб-додатків і так далі. Саме на цій мові програмування було написано всю логіку додатку. На рисунку 2.3 відображено головну сторінку сайту для розробників на мові програмування Java.

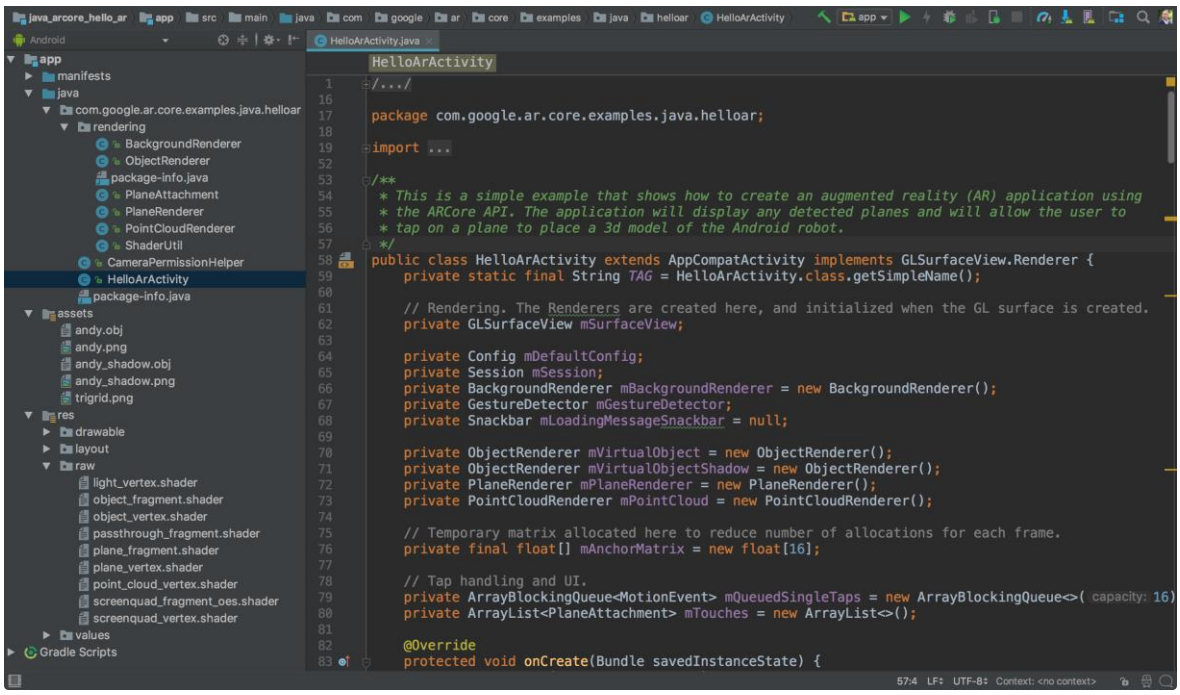

Рисунок 2.3 – Приклад використання Java

Далеко не кожен розробник має у своєму арсеналі глибокі знання програмування та значний багаж навичок в розробці програмного забезпечення. Щоб розробнику спростити процес написання коду, було придумано так званий комплект для розробки програмного забезпечення, по простому SDK (Sotfware Development Kit) [22]. Таких комплектів існує дуже велика кількість. Кожен із них призначений для різних потреб.

В свою чергу Android SDK – це універсальний засіб розробки мобільних додатків під операційну систему Android [23]. До складу Android SDK включені різні засоби розробки, у тому числі налагоджувач, великий набір бібліотек, емулятор телефону, набір документації та приклади додатків. Так як додаток разробляється під операційну систему Android доцільно використовувати саме Android SDK. Рисунок 2.4 відображає головну сторінку сайту із документацією Android SDK.

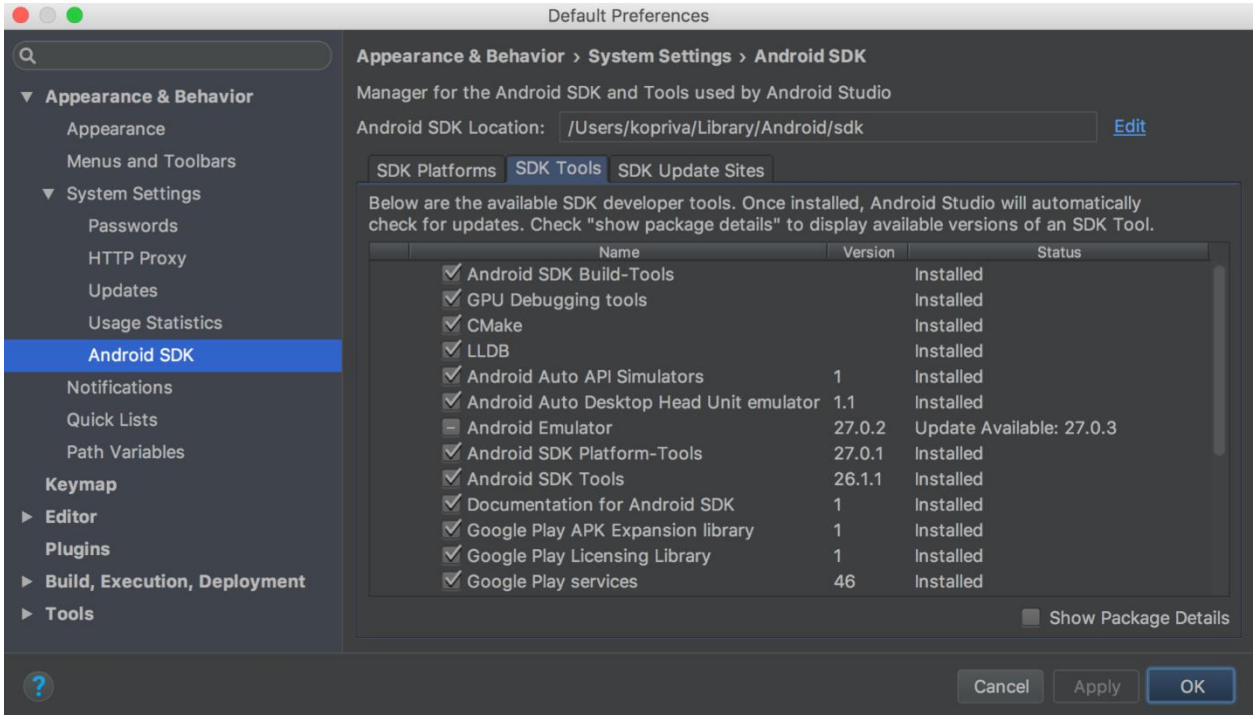

Рисунок 2.4 – Приклад використання Android SDK

Тепер перейдемо до опису вибору бази даних. Багатьом із нас відомо, що бази даних використовуються для зберігання, обслуговування та доступу до будь-яких типів даних, що зберігаються у створеній базі даних. Бази даних можуть зберігати інформацію про людей, місця, речі і так далі. Ця інформація збирається в одному місці, щоб за нею можна було зручно спостерігати та аналізувати, тобто бази даних можна розглядати як організований збір та зберігання інформації.

Майже всі додатки на базі операційної системи Android використовують систему управління базами даних SQLite [24]. SQLite представляє собою реляційну систему управління даними. Реляційна база даних – це набір даних із зумовленими зв’язками між ними [25]. Саме дана СУБД використовується для збереження даних в розроблюваному додатку. На рисунку 2.5 зображено головну сторінку СУБД SQLite.

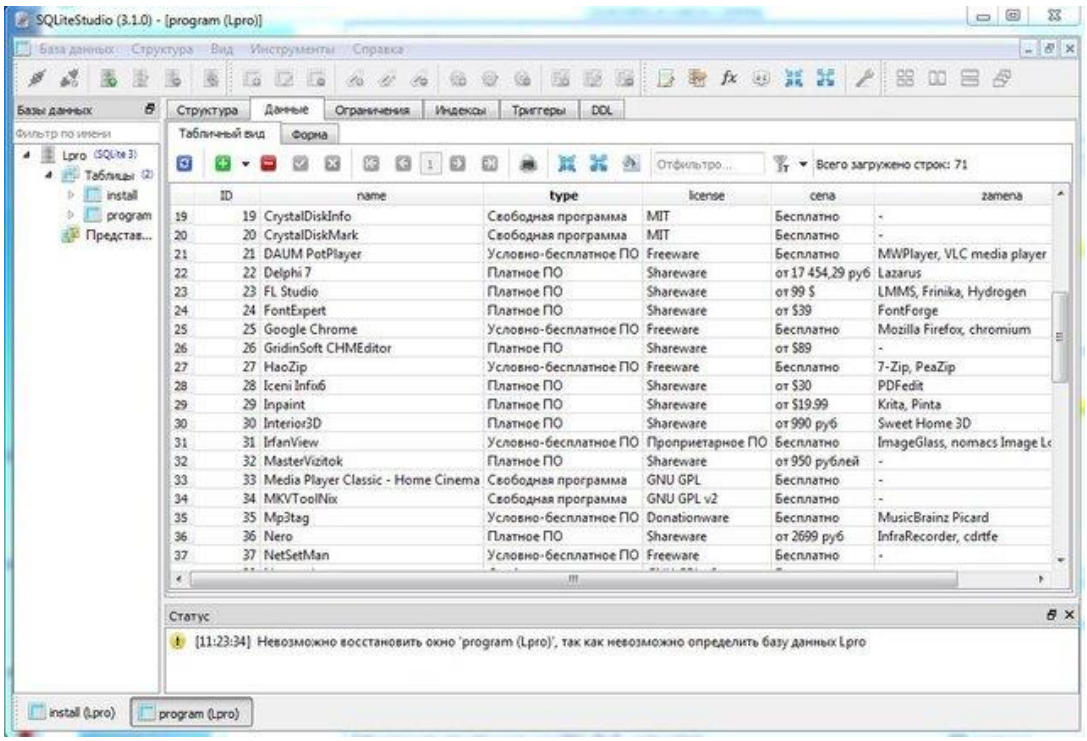

Рисунок 2.5 – Приклад використання СУБД SQLite

Щоб зберегти масу часу на пошук, встановлення, налаштування програм, найкраще за все використовувати вже готовий набір інструментів, для розробки застосунків під операційну систему Android. Тому останнім засобом реалізації обрано інтегроване середовище розробки (Integrated Development Environment – IDE) Android Studio [26].

Android Studio – середовище розробки, створене компанією JetBrains на базі програмного забезпечення IntelliJ IDEA на замовлення корпорації Google [27-28]. Говорячи простою мовою – це багатий набір інструментів, що використовується для полегшення розробки Android додатків. Гнучкість у розробці, можливості тестування, підтримка декількох мов програмування та вбудований емулятор роблять дану IDE однією з найкращих у своєму сегменті.

До того ж, описані вище Java, Android SDK та SQLite уже встановлені в середовище розробки. Крім Android Studio звичайно на ринку існують і інші рішення, але конкретно для даного проекту було обрано саме дане середовище розробки, тому що практично не вимагає складних маніпуляцій із

настроюванням середовища для комфортної роботи. На рисунку 2.6 зображено головне вікно програми Android Studio.

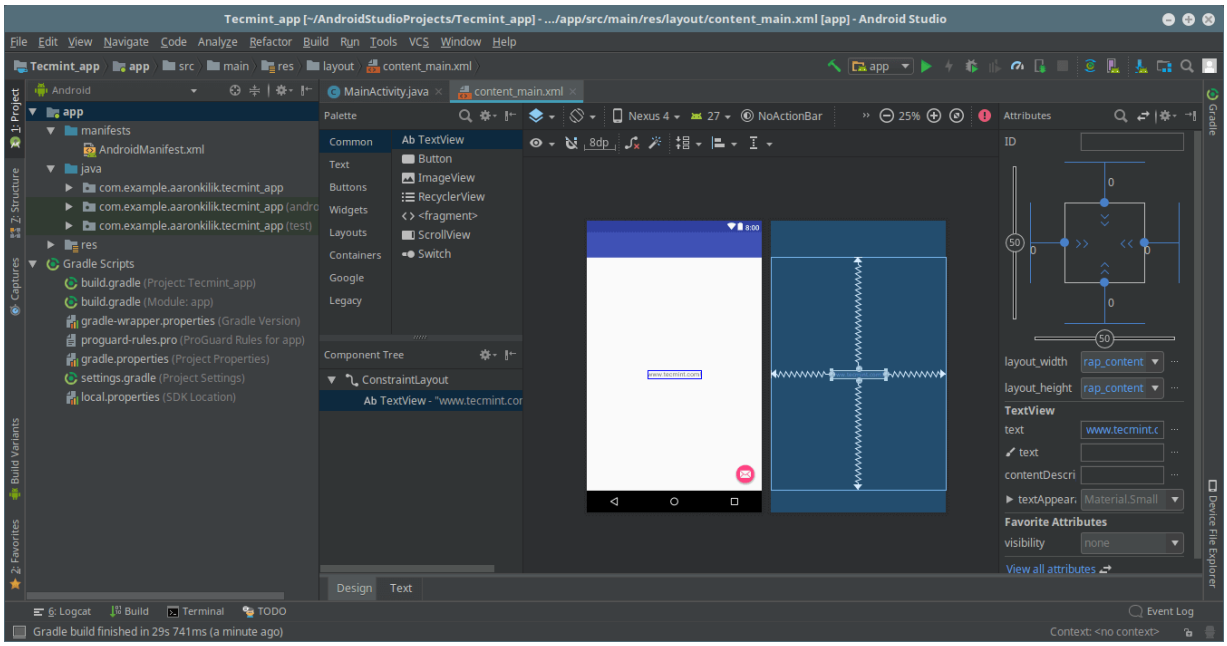

Рисунок 2.6 – Головне вікно програми

Для того, щоб розробити додаток, описаних вище програмних продуктів зазначено в необхідній кількості.

3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ЗАМОВЛЕНЬ ПЛАСТИКОВИХ ВІКОН

3.1 Структурно-функціональне моделювання проекту

Для того, щоб відобразити функціонування додатку було обрано методологію IDEF0 [29]. IDEF0 – це багатофункціональна методика прогнозування (функціональне моделювання), яка має графічне позначення, основне призначення якої є нормалізація та зображення бізнес-процесів. Характерним виключенням IDEF0 є його наголос на підпорядкованості об’єктів, другими словами – об’єкти даної методології представлені в вигляді функціональних блоків, що в свою чергу дозволяє проектувати необхідні процеси в розроблювальному продукті. Це надзвичайно зручно з точки зору розробки проекту в цілому, тому що наочність всіх процесів в графічному вигляді, дозволяє швидко визначатися із пріоритетністю виконання завдань.

IDEF0 являє собою багатофункціональну модель із послідовністю блоків, будь-який з яких – це так звана «чорна скринька» із вхідними та вихідними даними, елементами управління та механізмами, які детально розкладаються (декомпозуються) до необхідного рівня деталізації. Найбільш важливий процес (функція) розташований у верхньому лівому куті. Процеси зв’язані між собою за допомогою стрілок та описів функціональних блоків, будь-яка версія стрілки чи дії має своє значення та описується на діаграмі. Наведена модель дає можливість відобразити всі основні варіанти процесу.

Основним компонентом моделі IDEF0 є функція або процес. На діаграмі даний процес відображено в вигляді блоку. Блок в свою чергу – це прямокутник, всередині якого показано дію (процес). Процес може бути різним за масштабом – від діяльності компанії загалом і до конкретної маніпуляції зокрема.

Контекстна діаграма (діаграма верхнього рівня) є вершиною структури

дерева діаграм, демонструє призначення системи (основна функція) та її взаємодію із зовнішнім середовищем. У кожній модифікації може бути лише одна контекстна діаграма. Після завершення опису основної функції робиться багатофункціональне розкладання, тобто визначаються функції, що утворюють основну діаграму.

Контекстну діаграму інформаційної системи обліку замовлень пластикових вікон показано на рис. 3.1.

Рисунок 3.1 – Контекстна діаграма

Після того, як закінчили розробку контекстної діаграми, проведемо декомпозицію, описавши послідовність дій для інформаційної системи обліку замовлень пластикових вікон:

- Ідентифікація користувача;
- Реєстрація;
- Обирання товару;
- Обирання проміжку часу;
- Додати / видалити зі Списку товарів;
- Редагування облікового запису;
- Зберегти в БД.

Декомпозицію діаграми рівня IDEF0 наведено на рис. 3.2.

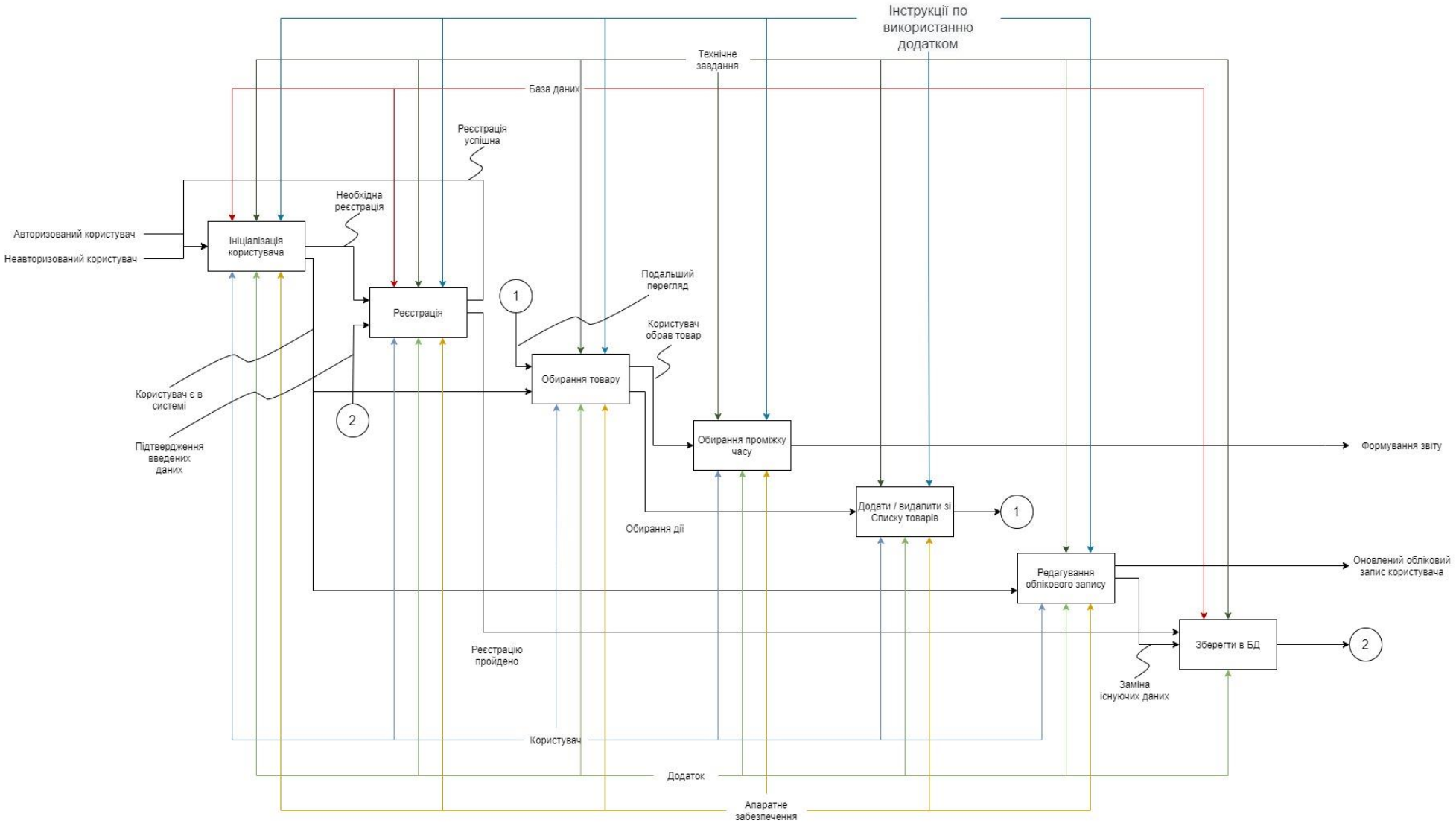

Рисунок 3.2 –Декомпозиції IDEF діаграми

3.2 Діаграма Use-Case

По завершенню моделювання IDEF діаграми, тобто діаграми функціональних процесів, перейдемо до розробки діаграми варіантів використання (ВВ). Діаграма Варіантів використання входить в один із етапів моделювання системи засобами мови UML [30].

Під UML (Unified Modeling Language) розуміється наступне – це уніфікована мовп об’єктно-орієнтованого моделювання, основне призначення якої є підтримка життєвого циклу продукту та активності учасників розроблювального продукту на різних етапах проекту.

Щодо діаграми Варіантів використання, основна роль діаграми полягає в відображенні сценаріїв використання (Use Cases), на діаграмі вони позначаються овалом, та користувачів (Actors), на діаграмі позначаються в формі чоловічка, які використовують функції системи [31]. Use Case diagram для інформаційної системи обліку замовлень пластикових вікон зображена на рисунку 3.3.

Користувачу надаються декілька функцій, а саме Реєстрація, Авторизація, Пошук товару, Використання фільтру, Перегляд списку товарів, Перегляд статистики по товару(ам), Аккаунт.

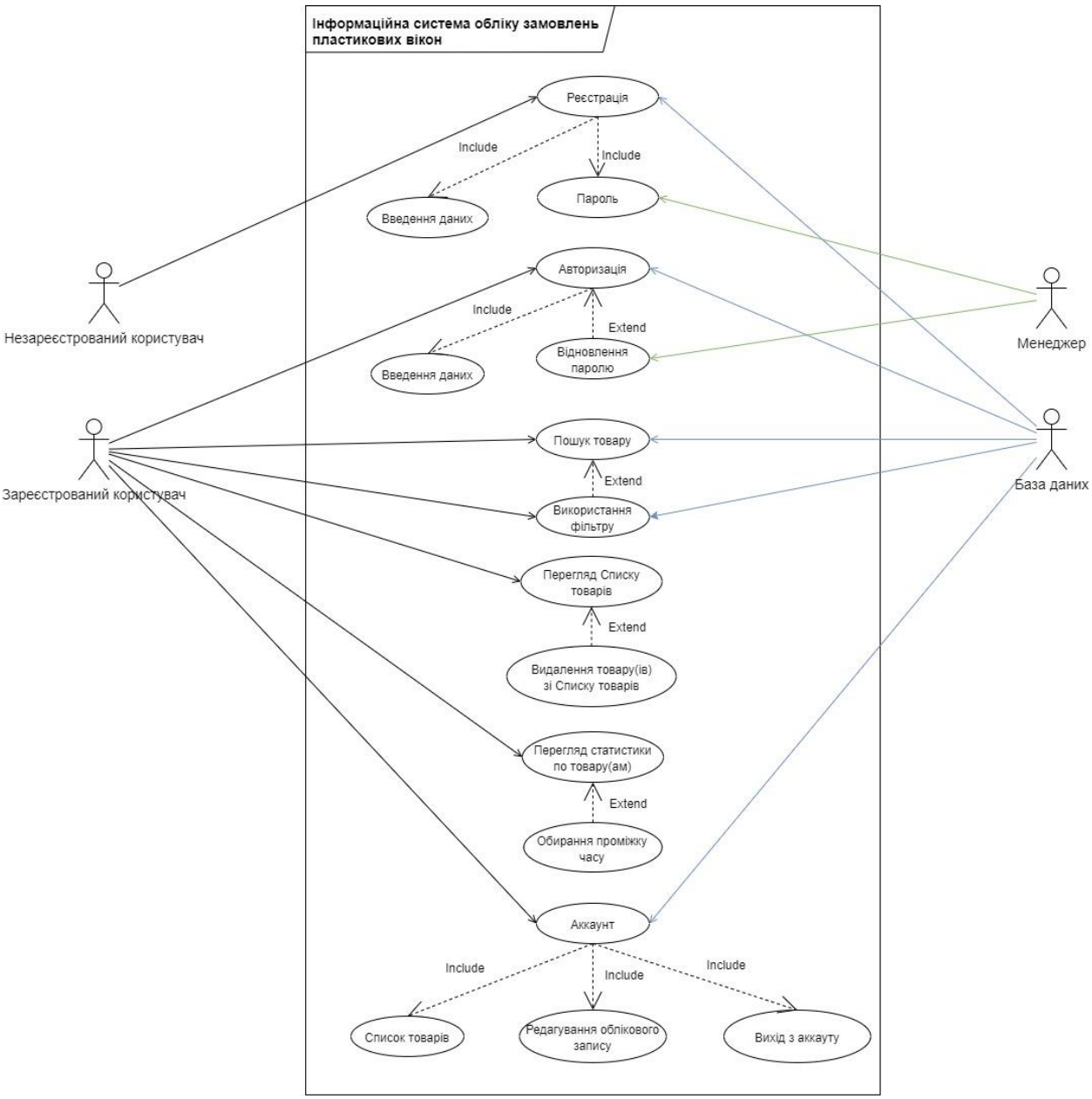

Рисунок 3.3 – Діаграма сценаріїв використання

Далі детальніше описано кожен із варіантів використання:

— Реєстрація – для використання додатку користувач повинен зареєструватися в ньому. В іншому випадку користувач не зможе ним користуватися. Реєстрація вимагає лише ініціали та спеціальний пароль;

— Авторизація – задля користування всіма функціями застосунка, користувач повинен бути авторизованим. Авторизація вимагає лише спеціального паролю;

— Пошук товару – якщо користувачу не вдається знайти необхідний

товар, завжди можна скористатися Пошуком;

— Використання фільтру – в ситуації, коли користувачу не вдається знайти товар за допомогою Пошуку, можна скористатися Фільтром для більш детального пошуку;

— Перегляд списку товарів – швидкий доступ до списку із доданими товарами в список товарів;

— Перегляд статистики по товару(ам) – користувач може переглянути статистику по обраному товару(ам). За замовчуванням статистика відображається за сьогодні, але при бажанні, користувач може обрати необхідний проміжок часу;

— Аккаунт – за необхідності користувач може відредагувати існуючі дані, змінити пароль, вийти з додатку та переглянути список товарів.

3.3 Проектування бази даних

Перед початком проектування будь-якої бази даних необхідно спочатку досконало вивчити документацію на розробку продукту. Основна проблема, яку потрібно буде вирішити розробнику – це правильно побудувати базу даних, а точніше схему (структуру) бази даних. База даних повинна бути розроблена таким чином, щоб вона була зрозумілою (потрібно для того, щоб інші розробники змогли швидко розібратися у всій ієрархії створеної БД), найбільш точно відображала проблему і не містила непотрібних даних. На даний час, найбільш популярним рішенням для подібних задача є ER-діаграма (Entity Relationship Diagram) [32]. Дана модель представляє з себе блок-схему, в якій відображено, як сутності (entities) (люди, концепції, об’єкти і так далі) взаємодіють (relationships) між собою всередині системи. Основне призначення полягає в наступному – спрощення процесу проектування бази даних.

На рисунку 3.4 приведено розроблену для інформаційної системи обліку замовлень пластикових вікон модель бази даних. Код створених таблиць можна переглянути в Додатку Б.

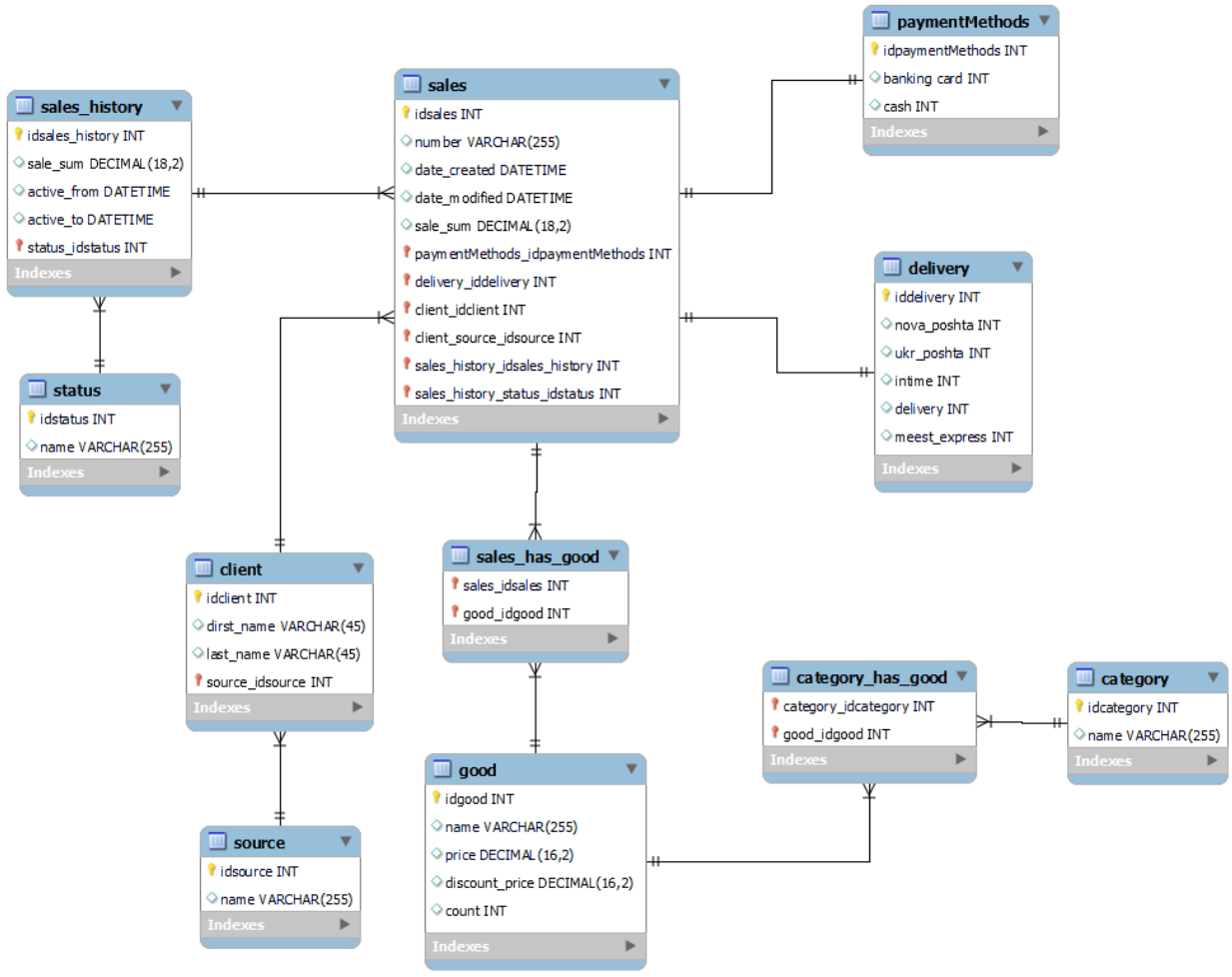

Рисунок 3.4 – ER діаграма додатку

В таблиці 3.1 описано всі таблиці бази даних, котрі відображено на рисунку 3.4.

Таблиця 3.1 – Опис таблиць та їх атрибутів.

Назва	Опис
sale – таблиця, в котрій зберігаються дані про продажі товарів.	**idsales** – ідентифікатор таблиці продаж. **number –** кількість продаж.

Продовження таблиці 3.1

	date_created – дата додавання товару до таблиці.
	date_modified – дата продажу товару. **sale_sum** – кількість проданих товарів. **idpaymentMethods (FK) –** зовнішній ключ таблиці dpaymentMethods. **iddelivery (FK)** – зовнішній ключ таблиці delivery. **idclient (FK)** – зовнішній ключ таблиці client. **client_idsource (FK)** – зовнішній ключ таблиці source. **idsales_history (FK)** – зовнішній ключ таблиці sales_history. **sales_status_idstatus (FK) -** зовнішній ключ таблиці status.
sales_has_good – проміжна таблиця, яка зберігає зовнішні ключі продаж та товарів.	**idsales (FK)** – зовнішній ключ таблиці sales. **idgood (FK) -** зовнішній ключ таблиці good.
good – таблиця, для збереження даних про товар.	**idgood –** ідентифікатор таблиці good. **name –** назва товару. **price –** ціна товару. **discount_price –** знижка на товар. **count –** поточна кількість товарів на даний момент.

Продовження таблиці 3.1

sales_history – дана таблиця зберігає історію продажів.	**idsales_history –** ідентифікатор таблиці sales_history. **sale_sum –** загальна сума продаж. **active_from –** дата додаваня товару до бази даних. **active_to –** як довго товар залишався в базі даних. **id_status (FK) –** зовнішній ключ табліці status.
paymentMethods – таблиця, для збереження даних про способи оплати.	**idpaymentMethods –** ідентифікатор таблиці paymentMethods. **banking_card –** відсоток оплачених товарів за допомогою кредитної карти. **cash -** відсоток оплачених товарів готівкою.
delivery - таблиця, для збереження даних про способи доставки.	**iddelivery –** ідентифікатор таблиці delivery. **nova_poshta –** відсоток доставлених товарів за допомогою Нової Пошти. **ukr_poshta –** відсоток доставлених товарів за допомогою Укр Пошти. **intime –** відсоток доставлених товарів за допомогою ІнТайм. **delivery –** відсоток доставлених товарів за допомогою Делівері. **meest_express -** відсоток доставлених товарів за допомогою Міст Експрес.

Продовження таблиці 3.1

client - таблиця, для збереження даних про користувача додатку.	**idclient –** ідентифікатор таблиці client. **first_name –** ім’я користувача. **last_name –** прізвище користувача. **idsource (FK) -** зовнішній ключ табліці source.
status - таблиця, для збереження статусу товару, таких як: оплачено, очікування оплати, оплата при отриманні і так далі.	**idstatus -** ідентифікатор таблиці status. **name –** назва статусу товару.
source – таблиця, для збору інформації про те, яким способом користувач потрапив до застосунку.	**idsource –** ідентифікатор таблиці source. **name –** назва способу.
category – таблиця, для збеження інформації про те, до якої категорії відноситься те чи інше вікно.	**idcategory -** ідентифікатор таблиці category. **name –** назва категорії.
category_has_good - проміжна таблиця, яка зберігає зовнішні ключі категорії та товару.	**idcategory (FK) –** ідентифікатор таблиці category. **idgood (FK) -** ідентифікатор таблиці good.

4. РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ЗАМОВЛЕНЬ ПЛАСТИКОВИХ ВІКОН

4.1 Складові етапи розробки додатку

Розробка додатку проходила у гнучкому режимі, тобто процес вівся в ітераційному режимі. Під ітераційним режимом розуміється наступне – кожна ітерація поєднує у собі перераховані стадії розробки. Звичайно при такому підході кожну ітерацію (складовий етап) можна описати. Для опису було виділено чотири основні етапи розробки додатку. Перший етап – збір необхідної інформації та виявлення мети. Другий етап – розробка дизайну на основі отриманої інформації. Третій етап – розробка логіки додатку, написання коду. І завершальний четвертий етап – тестування та введення в експлуатацію.

Перед початком створення будь-якого проекту стоїть завдання виявлення – що є метою розробки проекту та які задачі буде виконувати кінцевий продукт. Для цього необхідно зібрати якомога більше інформації. Найчастіше інформацію отримують від аналізу подібних рішень, які вже давно є на ринку і стали успішними. У народі такий підхід називають – аналіз конкурентів [33]. Звичайно існують і інші підходи для збору інформації, ось деякі з них: якісний аналіз, кількісний аналіз, збір даних на основі опитувань, брейнстормінг і так далі [34].

Так як додаток, що розробляється, не є унікальним в своєму роді, то для даних цілей було обрано саме аналіз конкурентів. Даний метод допомагає швидко визначити, як кінцевий продукт приблизно повинен виглядати. Процес збору інформації на основі конкурентного аналізу полягає в наступному: необхідно визначити сильні та слабкі сторони, аналіз зручності використання рішенням, аналіз контенту (наповнення). Саме за таким

алгоритмом було здійснено збір інформації в рамках додатку, що розробляється.

Після того, як проведений аналіз конкурентів, настає черга проектування, або інакше кажучи, процес створення дизайну майбутнього продукту – мобільного додатка. Процес створення дизайну дуже важливий для розробки мобільного додатку. Добре спроектований дизайн допомагає швидко освоїтися з наданим функціоналом додатку і швидко зрозуміти, як саме працює той чи інший блок у додатку. Одним з найважливіших етапів при розробці є дотримання поставлених цілей і завдань, а також їх здійснення в продукті, що розробляється. На підставі отриманих даних після аналізу подібних рішень створено дизайн майбутньої додатку інформаційної системи обліку замовлень пластикових вікон.

Далі показано декілька чорнових варіантів майбутнього додатку із дотриманням всіх поставлених цілей та задач.

На рисунку 4.1 відображеного головну сторінку додатку.

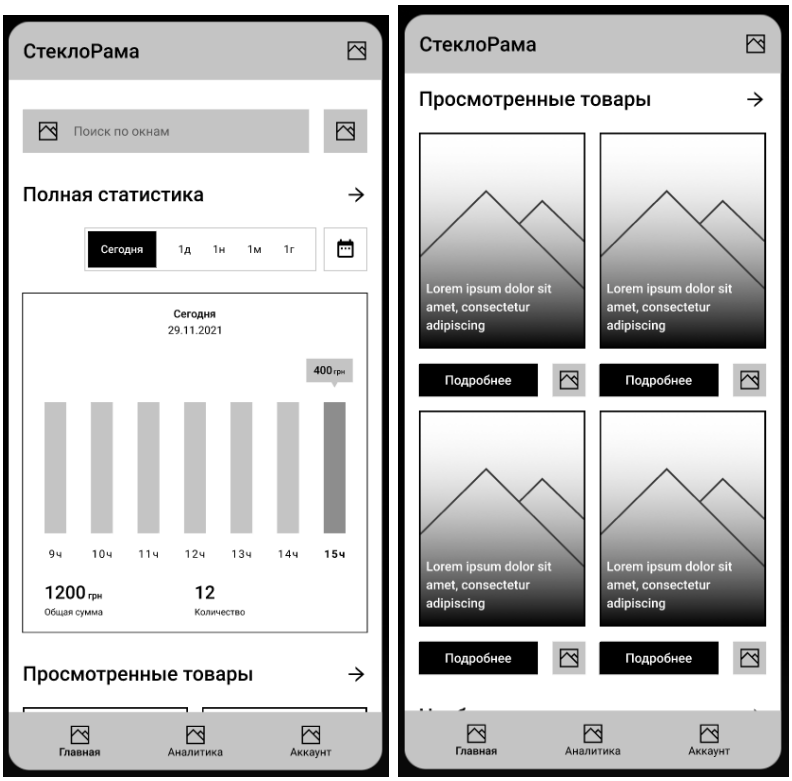

Рисунок 4.1 – Пункт меню додатку Главная

На рисунку 4.2 відображено сторінку із аналітикою.

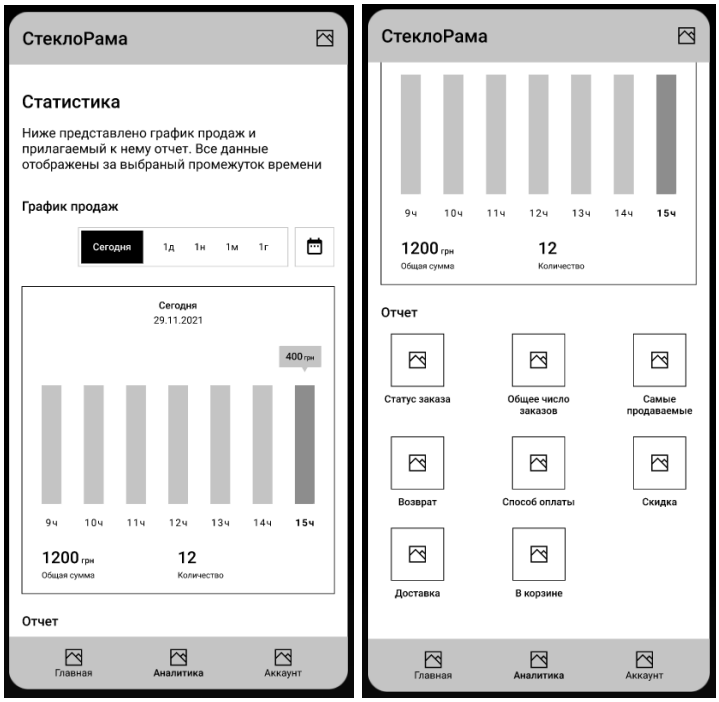

Рисунок 4.2 – Пункт меню додатку Аналітика

На рисунку 4.3 відображено сторінку із обліковим записом користувача.

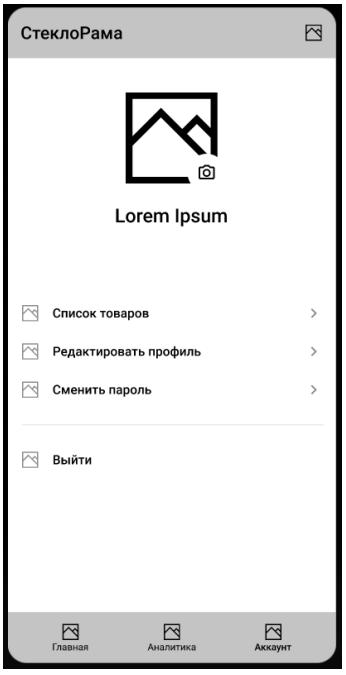

Рисунок 4.3 – Пункт меню додатку Аккаунт

Після завершення створення дизайну, настає черга програмування. Перед початком написання якого б то не було коду, необхідно визначитися з мовою програмування. У розділі 2 вже було згадано, що основною мовою програмування для створення мобільних застосунків під платформу Android є Java. Додаток, що розробляється, не є виключенням, тож мовою програмування обрано саме Java. На цьому етапі розробник працює над написанням коду, основним завданням якого є зв'язування дизайну та програмної частини воєдино. На даному етапі виконуються такі дії: розробник проектує базу даних, пише необхідну логіку для збереження та видачі потрібної інформації у додаток, пише код для логічної послідовності дій під час використання додатком.

І нарешті останнім етапом розробки додатку йде тестування та введення в експлуатацію. На цьому етапі розробнику необхідно протестувати повністю весь додаток. Якщо після ретельного тестування виявлено помилки – обов'язково їх усунути. У випадку, якщо замовника щось не задовольняє, розробник повинен внести коригування та усунути неполадки. Тільки після того, як тестування завершилося з позитивним результатом, додаток можна вводити в експлуатацію.

4.2 Програмна реалізація

Наступним кроком описано логіку додатку, а саме які класи інтерфейси, методи та графічну складову було створено протягом розробки мобільного застосунку.

Оскільки основною мовою програмування було обрано Java, а середовищем розробки Android Studio, для початку опишемо у вигляді таблиці класи, інтерфейси та методи, створені під час роботи над проектом. На рисунку 4.4 відображено пакет com.example.steklorama, в якому зберігаються класи та інтерфейси даного додатку.

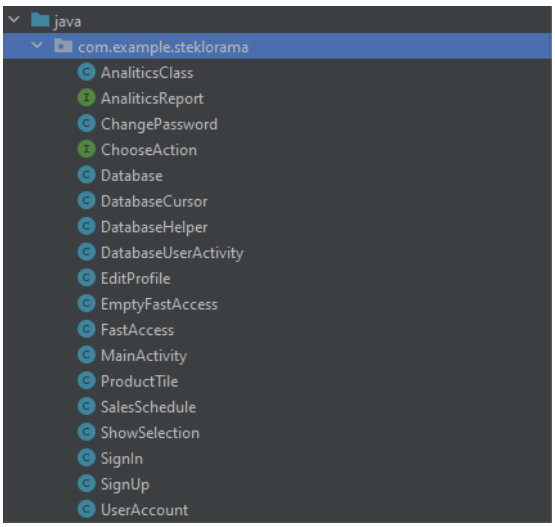

Рисунок 4.4 – Класи та інтерфейси додатку

Таблиця 4.1 – Опис класів, інтерфейсів та методів

Клас	Метод(и)
AnaliticsClass – клас, який керує всією логікою пункту меню «Аналитика»	**statisticsByDefault() –** метод, який відображає інформацію в пункті меню «Аналитика» за замовчуванням. Також даний метод здійснює перехід від секції «Полная статистика» на головній сторінці програми.
	statisticsByProduct() – відображення статистики по обраному товару.
AnaliticsReport – Інтерфейс, який містить методи, необхідні для відображення звіту.	**currentProductCost() –** метод повертає поточну ціну обраного товару.
	availabilityInStore() – метод перевіряє чи є товар в наявності.
	totalNumberOfOrders() – метод повертає загальну кількість замовлень.

Продовження таблиці 4.1

	orderStatus() – даний метод повертає кількість товарів, що очікують оплати та кількість товарів, що будуть оплачені при отриманні.
	bestSelling() – даний метод використовується для того, щоб відобразити товари із найбільшою кількістю продажів.
	paymentMethods() - метод, відображення у відсотковому еквіваленті, яким методом оплати користуються користувачі.
	orderReturn() – кількість всіх товарів, що були повернуті.
	productDiscount() – кількість всіх товарів зі знижкою.
	delivery() – дана функція робить підрахунок у відсотковому вигляді, яка компанія більше за все використовується для доставки.
	inCart() – кількість всіх товарів, що знаходяться в корзині.
ChangePassword – клас, призначення якого надати користувачу змогу змінювати пароль від облікового запису.	**usersEmail() –** метод, котрий перевіряє правильність введеної електронної пошти. Також даний метод потрібен для того, щоб відправити користувачу листа на пошту з підтвердженням зміни паролю.

Продовження таблиці 4.1

	createNewPassword() – створення підтвердження нового паролю.
ChooseAction – інтерфейс, який надає методи, для переходу від одного пункту меню в інший.	**goToMain() –** перейти на сторінку «Главная».
	goToAnalitics() - перейти на сторінку «Аналитика».
	goToAccount() - перейти на сторінку «Аккаунт».
Database – даний клас використовується для підключення да Бази даних.	**openOrCreateDatabase() –** метод, для створення або відкриття нової бази даних.
	exceSQL() – метод, для виконання запитів до бази даних.
	close() – завершення роботи бази даних.
DatabaseCursor – клас, призначення якого, виконати запит до бази даних та знайти необхідну інформацію по заданому запиту.	**rawQuery() –** даний метод використовується для отримання будь-яких даних із бази даних.
	getCount() – кількість віх обраних записів в базі даних.
	moveToFirst() – метод, який дає змогу переміститися до першого запису в базі даних.
	moveToNext() - метод, який дає змогу переміститися до наступного запису в базі даних.

Продовження таблиці 4.1

	moveToLast() - метод, який дає змогу переміститися до останнього запису в базі даних.
	isAfterLast() - повертає значення, чи вказує курсор на позицію після останнього рядка в таблиці.
	get*(Int, Float, Double, Char, String) – даний методи використовуються для того, щоб отримати дані в певному форматі.
	simpleCursorAdapter() - даний метод дозволяє адаптувати отриманий за допомогою курсору набір даних, для відображення у спискових елементах на кшталт компонента ListView.
DatabaseHelper - створює допоміжний об’єкт для створення, відкриття та/або управління базою даних.	**onCreate() -** викликається при спробі доступу до бази даних, але коли ще база даних не існує.
	onUpgrade() - викликається для оновлення таблиці бази даних.
DatabasUserActivity – клас, для управління так званою CRUD-логікою (створення, оновлення та видалення) даних.	**update() –** оновлення даних.
	insert() – додавання нових даних.
	delete() – видалення даних.
	ContentValues() – даний конструктор служить, щоб ініціалізувати об’єкт, котрий буде використано для додавання даних до БД.

Продовження таблиці 4.1

	put() – саме цей метод використовується для додавання нового об’єкта у вигляді ключ-значення.
EditProfile – клас, для управління редагуванням облікового запису користувача.	**setNew*(Name, Surname, Position, Email) –** методи, для заміни нового імені, прізвища, посади та email.
	getOld*(Name, Surname, Position, Email) – методи повертають значення ім’я, прізвища, посади та email.
	saveChanges() – метод, для збереження виконаних операцій по редагуванню облікового запису.
EmptyFastAccess – клас, призначення якого відображення пустої сторінки «Список товаров».	**goToMainPage() –** перехід на головну сторінку.
FastAccess – клас, який отримує об’єкт товару, щоб користувач мав змогу переглянути або видалити зі списку товарі бажаний товар.	**clearAllList() –** видалити всі додані товари.
MainActivity – головний клас додатку. Саме цей клас завантажується при першому запуску додатку.	**main() –** головний метод додатку. Без даного методу додаток ні при яких обставинах не запуститься.
	onCreate() - метод, який визначає початкову установку параметрів при ініціалізації активності.

Продовження таблиці 4.1

	goToFastAccess() - перехід до списку товарів.
	setOnClickListener() – метод повертає значення чи було натиснуто на кнопку.
	findViewById() – метод, призначення якого – знайти необхідний ідентифікатор представлення, що шукається.
	intent() – метод, для взаємодії між різними об’єктами діяльності (activity).
ProductTile – клас, для взаємодії користувача та плитки з товаром.	**showDetails() –** кнопка «Подробнее» натиснута і користувач потрапляє в пункт меню «Аналитика».
	addToFastAccess() – додати товар в «Список товаров».
	removeFromFastAccess() – видалити товар зі списку товарів.
SalesSchedule – клас, для обирання довільної дати чи проміжку часу, для відображення результату інформації за обраний проміжок.	**currentSales() –** метод, котрий за замовчуванням відображає статус продажів на графіку.
	choosenDaySales() – вибрати довільну дату.
	choosenWeekSales() - вибрати довільну неділю.
	choosenMonthSales() - вибрати довільний місяць.

Продовження таблиці 4.1

	choosenYearSales() - вибрати довільний рік.
	setArbitraryRange() – задати довільний проміжок часу.
	setAnotherRange() – змінити обрану дату.
	getAnotherRange() – отримати нову дату.
ShowSelection – клас, для переходу на нову сторінку із обраною підбіркою.	**selectedSection()** – обирання однієї із наданих підбірок.
SignIn – клас авторизації користувача.	**signIn()** – користувач почав проходити авторизацію, даний метод контролює весь процес авторизації.
	goToSignUp() – якщо користувач не має облікового запису, даний метод буде слугувати для переходу на сторінку реєстрації.
	forgotPassword() – в тому випадку, якщо користувач забув пароль від облікового запису, даний метод буде визвано.
SignUp – клас реєстрації користувача.	**signUp()** – даний метод буде визвано, коли користувач почне процес проходження реєстрації.
	goToSignIn() – якщо з якихось причин користувач згадав дані

Продовження таблиці 4.1

	необхідні для авторизації, цей метод буде викликано.
UserAccount – клас, для редагування облікового запису.	**changeAvatar() –** при натисканні на кнопку зміни аватару, буде викликано даний метод.
	selectNewPhoto() – обирання існуючого фото.
	makeNewPhoto() – зробити фото.
	goToFastAccess() – перейти до списку товарів.
	signOut() – вийти із додатку.

Далі показано файли розмітки на мові XML. XML файли використовуються для збереження і передачі даних в зручному для розуміння форматі. При розробці Android додатків, xml використовується для розробки макетів. В таблиці 4.2 описано розроблені xml файли. На рисунку 4.5 відображено xml файли, котрі було створено при розробці додатку.

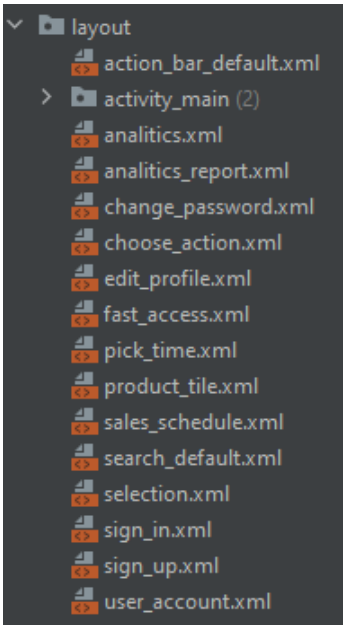

Рисунок 4.5 – XML файли в пакеті layout

Таблиця 4.2 – Розроблені XML файли та їх опис

Назва	Опис
action_bar_default.xml	У цьому файлі міститься розмітка компонента action bar, в якому розміщено логотип додатку та активну функцію для переходу в «Список товаров».
analytics.xml	Даний файл містить розмітку для одного із пунктів меню, а саме «Аналитика».
analytics_report.xml	Файл із розміткою всіх пунктів звіту.
change_password.xml	Для того щоб користувач мав змогу змінити пароль, даний файл розміщує в собі всі необхідні поля вводу та пояснення до них.
choose_action.xml	Розмітка всіх пунктів меню, котрі необхідні для навігації по головним складовим додатку, а саме «Главная», «Аналитика» і «Аккаунит».
edit_profile.xml	Розмітка сторінки, для редагування облікового запису.
fast_access.xml	В даному файлі написаний код, для доступу до списку товарів.
pick_time.xml	Цей файл містить в собі розмітку, для обирання проміжної або проміжку часу.
product_tile.xml	Розмітка товару.

Продовження таблиці 4.2

sales_schedule.xml	Розміщення графіка в додатку.
search_default.xml	Зовнішній вигляд компонента «Поиск», та його розміщення в додатку.
selection.xml	Даний файл відповідає за те, як буде відображено обрану підбірку із товарів.
sign_in.xml	Розмітка компонентів для авторизації.
sign_up.xml	Розмітка компонентів для реєстрації.
user_account.xml	Даний файл містить розмітку для одного із пунктів меню, а саме «Аккаунт».

У додатку Б наведено приклади коду розроблених xml файлів.

4.3 Використання інформаційної системи обліку замовлень пластикових вікон

Далі буде описано взаємодію додатку та користувача.

При першому входженні в додаток, користувач потрапляє на сторінку Авторизації. На цій сторінці йому пропонується ввести email та пароль від облікового запису, якщо такий існує. На рисунку 4.6 показано сторінку Авторизації. За замовчуванням кнопка «Войти» неактивна. Тільки після того, як користувач почне заповнювати поля, кнопка стане активною і користувач зможе по ній клікнути.

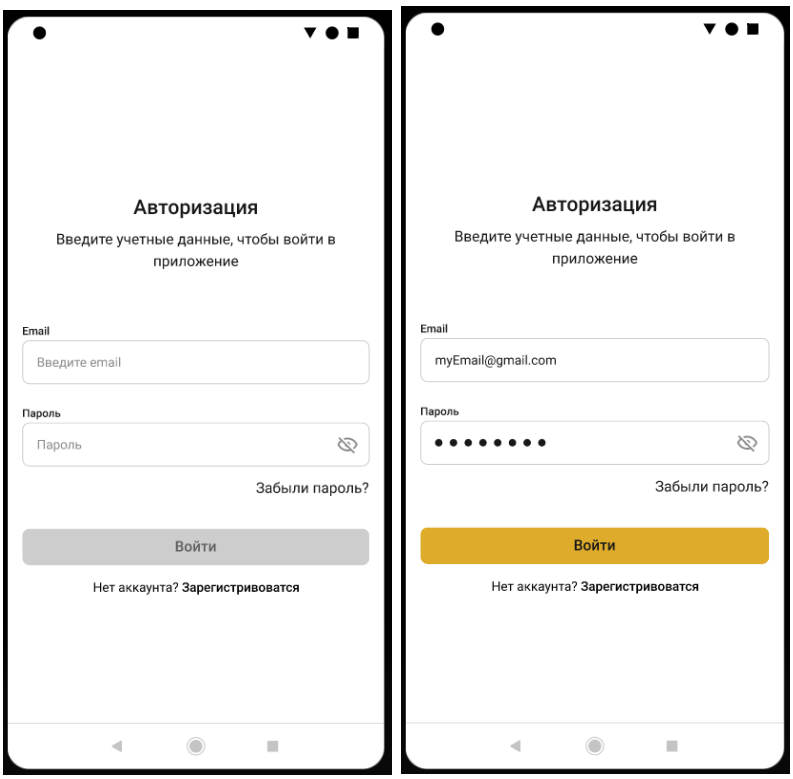

Рисунок 4.6 – Сторінка Авторизації

Якщо користувач ввів неправильні дані, додаток покаже помилку та запропонує перевірити введені дані.

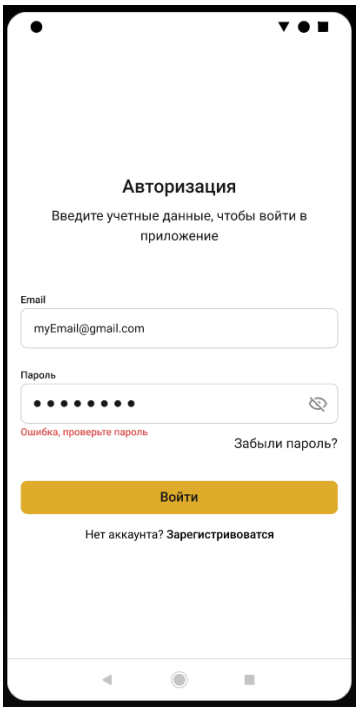

Рисунок 4.7 – Помилка при введенні даних

В тому випадку, якщо користувач не має облікового запису, він завжди може пройти Реєстрацію. На цій сторінці йому пропонується ввести Ім’я, Прізвище, Посаду, Email та Пароль. Так само, як і з Авторизацією, кнопка «Зерегистрироватся» не активна до тих пір, поки користувач не почне заповняти поля.

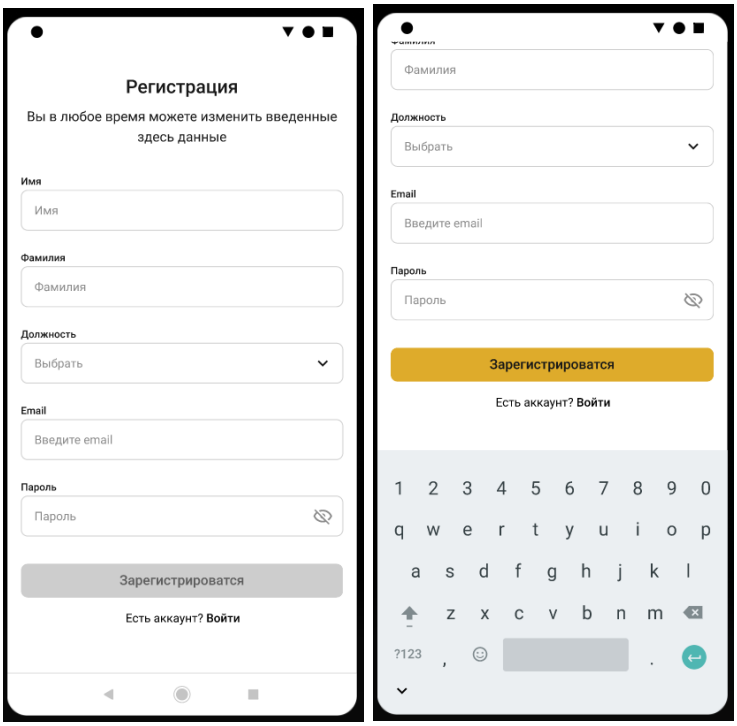

Рисунок 4.8 – Реєстрація користувача

Під час реєстрації також можуть статися помилки. На малюнку 4.7 показано помилку у полі Email – помилка в тому, що користувач із подібною електронною поштою вже існує. У такому випадку користувач може натиснути на відповідну кнопку, яка знаходиться під кнопкою «Зарегистрироваться» і увійти до існуючого облікового запису.

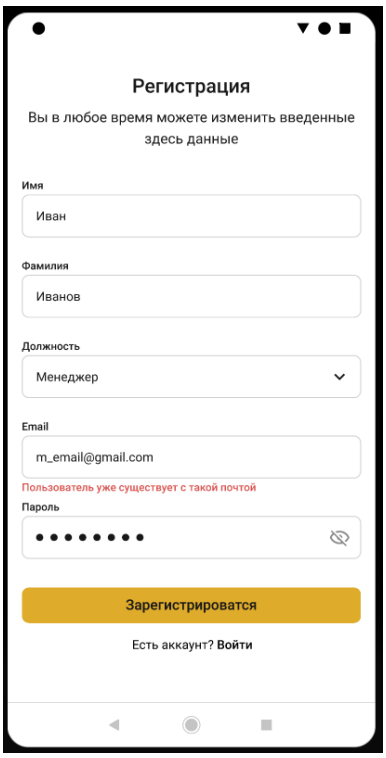

Рисунок 4.9 – Помилка на сторінці «Зарегистрироваться»

Якщо користувач ввів все правильно і система не показала жодних повідомлень з введеними даними, то реєстрація вважається успішною.

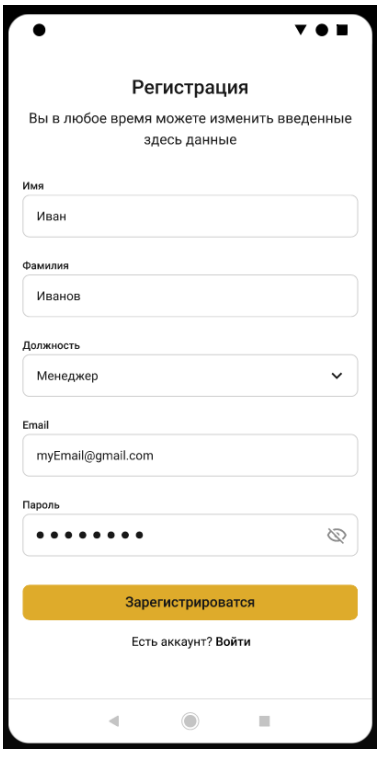

Рисунок 4.10 – Успішна реєстрація користувача

Якщо користувач пройшов успішну Авторизацію, або якщо знадобилось Реєстрацію, він потрапляє на сторінку «Главная». Ця сторінка містить у собі іконку, що веде в «Список товаров», «Поиск по окнам», «Фильтр» та різні підбірки із товарами.

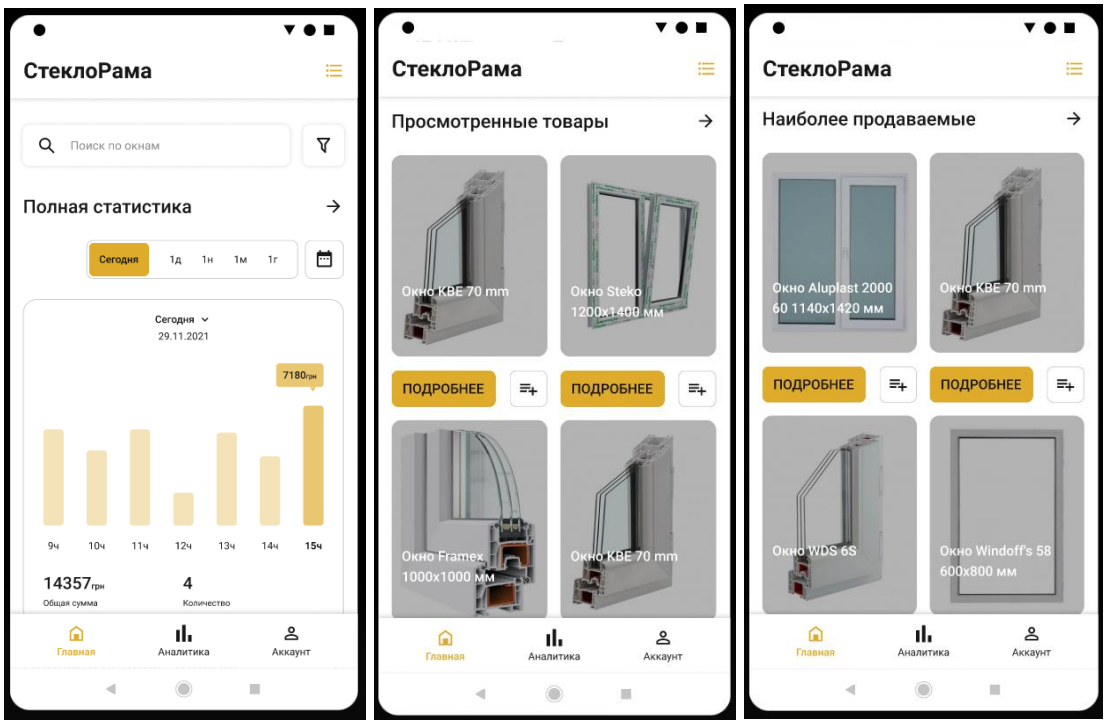

Рисунок 4.11 – Сторінка додатку «Главная»

При натисканні на іконку в правому верхньому кутку додатку, користувач потрапить на сторінку зі списком товарів. На рисунку 4.12 зображено «Список товаров» за замовчуванням. Як можна помітити, він порожній, то ж користувачу пропонується перейти на сторінку «Главная» і додати товар, що цікавить, до списку товарів.

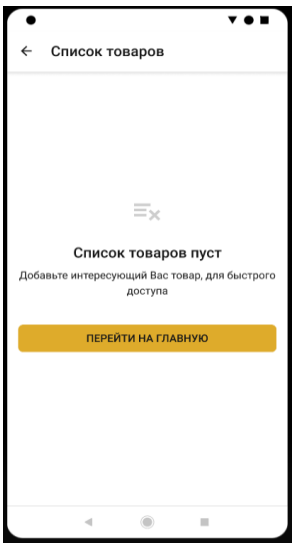

Рисунок 4.12 – Список товарів за замовчуванням

Якщо користувач перейшов на головну сторінку та додав до списку декілька товарів, натиснувши відповідну кнопку біля самого товару, то всі відмічені товари потраплять в «Список товаров». Користувач отримає миттєве повідомлення, в якому буде сказано, що товар успішно додано в «Список товаров».

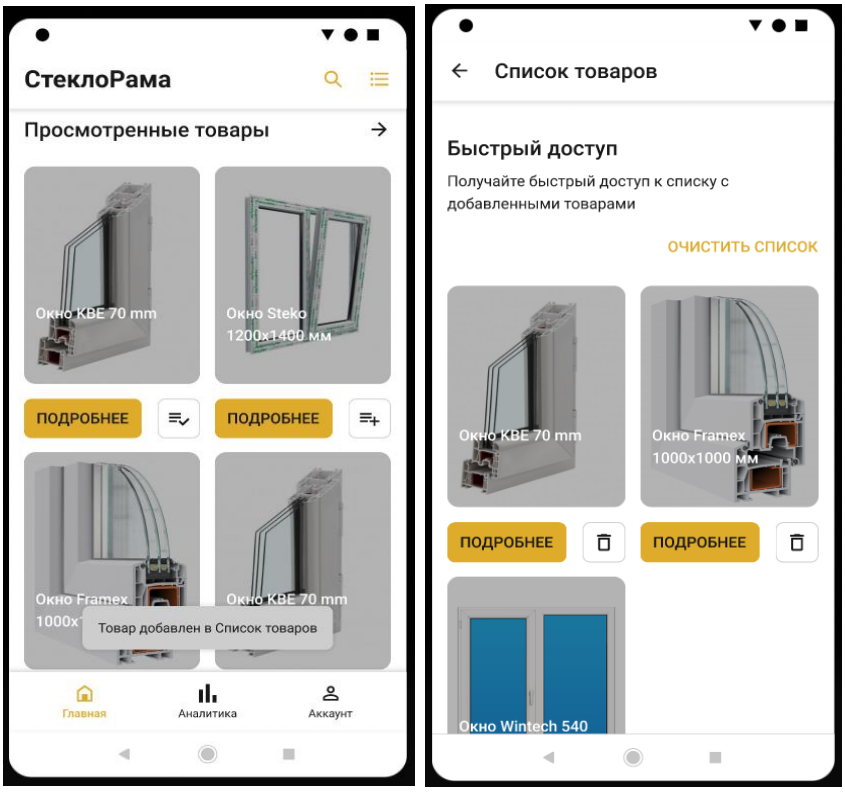

Рисунок 4.13 – Товари, котрі було додано в «Список товаров»

При натисканні на кнопку «Очистить список», всі товари буде видалено зі списку товарів, а сам «Список товаров» перейде в стан за замовчуванням, про який вище вже було описано.

В тому випадку, якщо користувач захоче видалити товар, він може натиснуту відповідну кнопку з іконкою, на якій зображено сміттєве відро, після чого товар буде видалено, а користувач в свою чергу отримає миттєве повідомлення про успішне видалення товару зі списку.

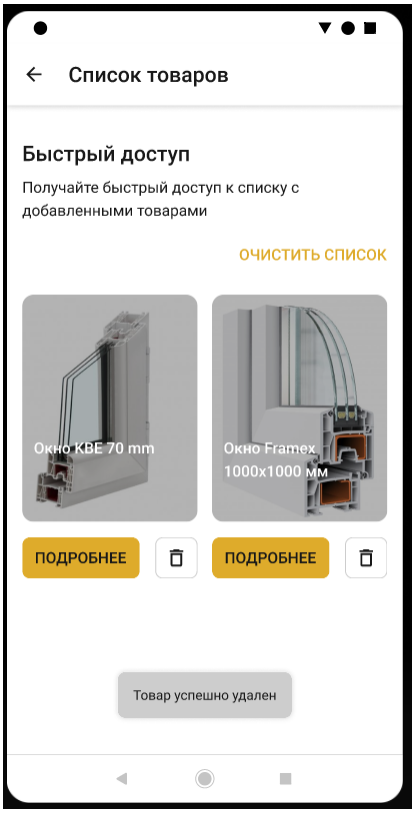

Рисунок 4.14 – Успішне видалення товару зі списку товарів

Наступна функція, котру пропонує додаток це – «Поиск» (пошук). За допомогою пошуку, користувач може знайти товар, що цікавить його. За замовчуванням пошук пропонує користувачу товари, котрі раніше він шукав.

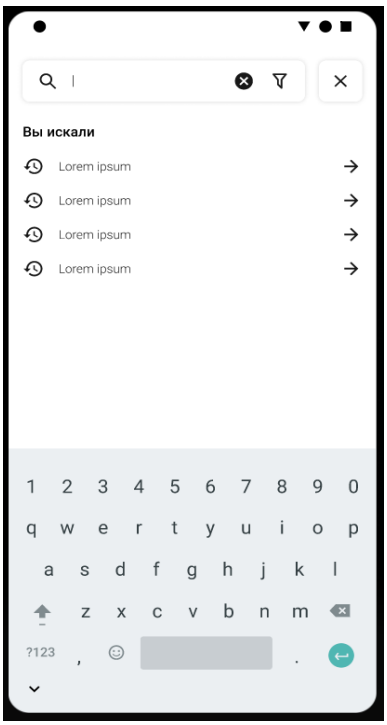

Рисунок 4.15 – Пошук за замовчуванням

Якщо користувач почав використовувати пошук, йому будуть пропонуватись варіанти товарів, які можна вибрати для подальшого переходу на сторінку «Аналитика».

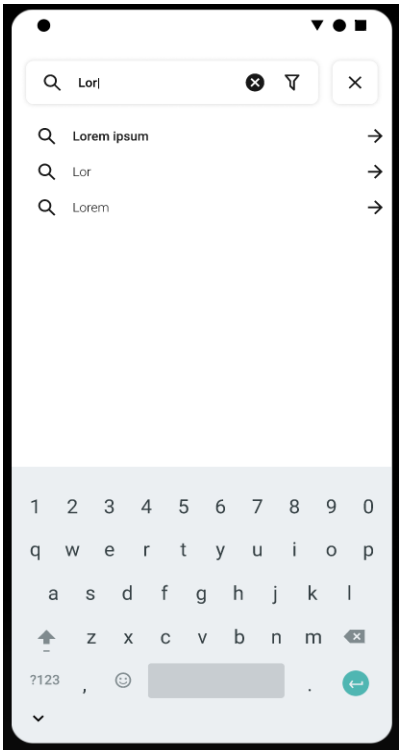

Рисунок 4.16 – Можливі варіанти пошуку

Біля поля пошуку знаходиться ще одна кнопка «Фильтр». При натисканні на дану кнопку користувач потрапить на наступну сторінку, показану на рисунку 4.17.

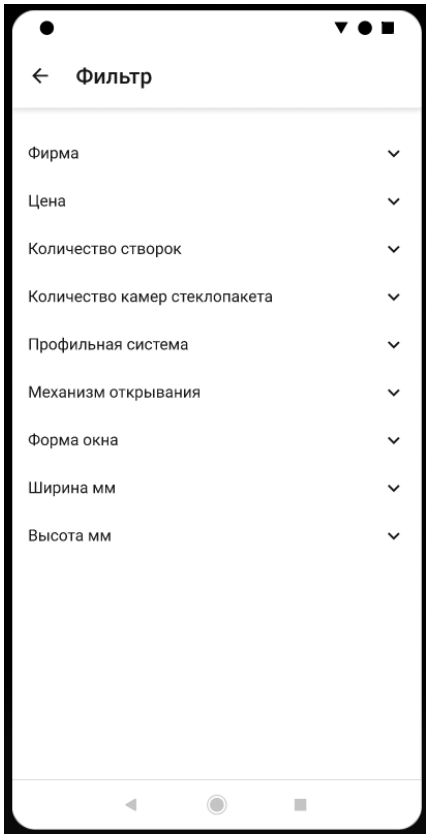

Рисунок 4.17 – Сторінка «Фильтр» за замовчуванням

Користувачу пропонується відфільтрувати пошук бажаного товару за характеристиками, котрі надає фільтр, для більш детального пошуку.

Далі буде показано перелік вікон, на котрих відображено всі характеристики вкладок, що зображені на рисунку 4.18.

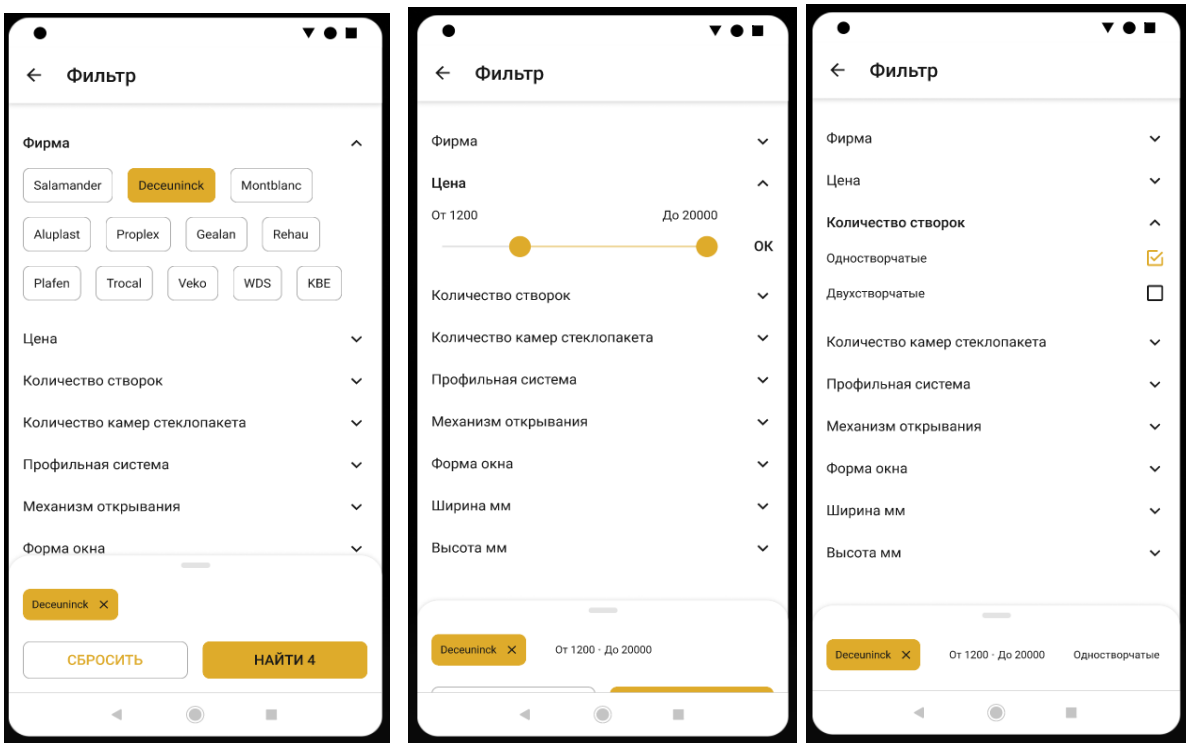

Рисунок 4.18 – Фільтр по фірмі, ціні та кількості стулок вікна

Наступний перелік використання фільтру із трьох характеристик представлено на рисунку 4.19.

Рисунок 4.19 – Кількість камер склопакету, профільна система та механізм відкривання

Наступні три вікна демонструють застосування інших параметрів фільтру (рис. 4.19).

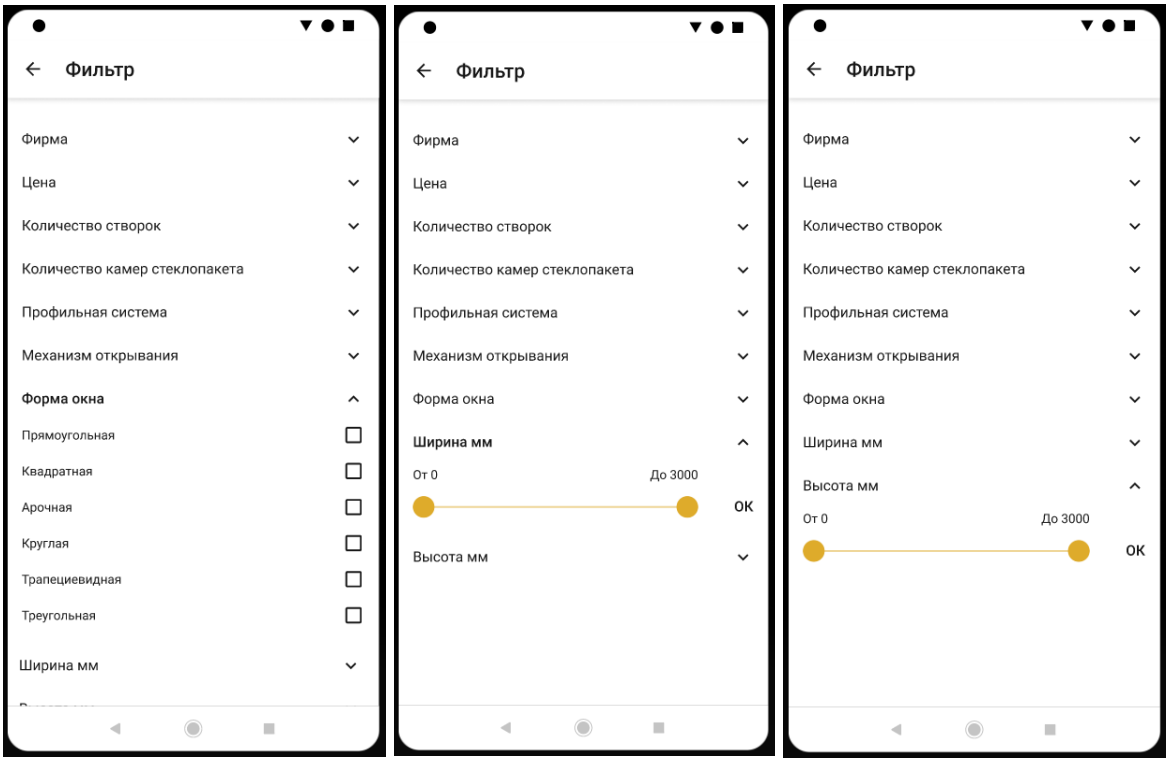

Рисунок 4.20 – Форма вікна, ширина та висота в мм

Після обрання необхідних характеристик для фільтру, користувач має змогу переглянути обрані характеристики, просто потягнувши в гору повзунок активного елементу додатку, що знаходиться в нижній частині додатку. Після чого буде показано сторінку зі «Списком фильтров». Якщо якийсь із фільтрів не потрібен користувачу, він має змогу в будь-який час видалити його зі списку фільтрів.

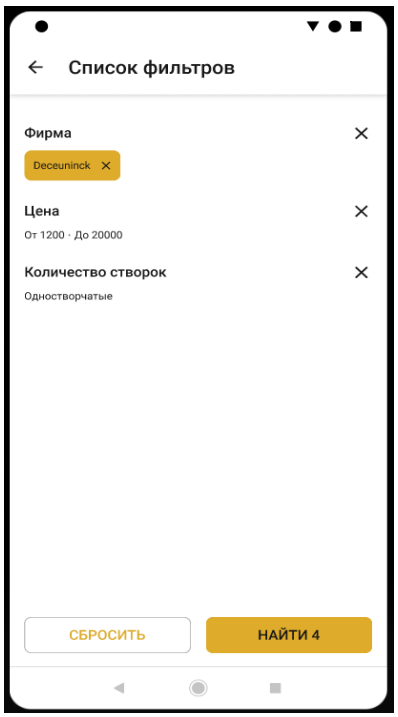

Рисунок 4.21 – «Список фильтров» з обраними характеристиками для фільтрування

Зрештою, користувач потрапить на сторінку з результатами фільтра, в якому буде показано товари, що підходять під обрані характеристики.

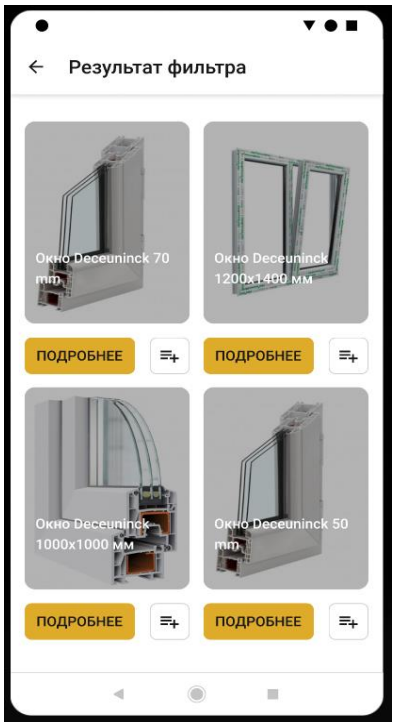

Рисунок 4.22 – Результати фільтрування

Коли користувач натиснув на головній сторінці по будь-якій із підбірок, додаток відобразить ті товари, котрі знаходяться в даній підбірці. Наприклад, користувач обрав підбірку «Просмотренные товары», додаток перенаправить користувача на сторінку з даною підбіркою, де йому буде запропоновано перелік товарів.

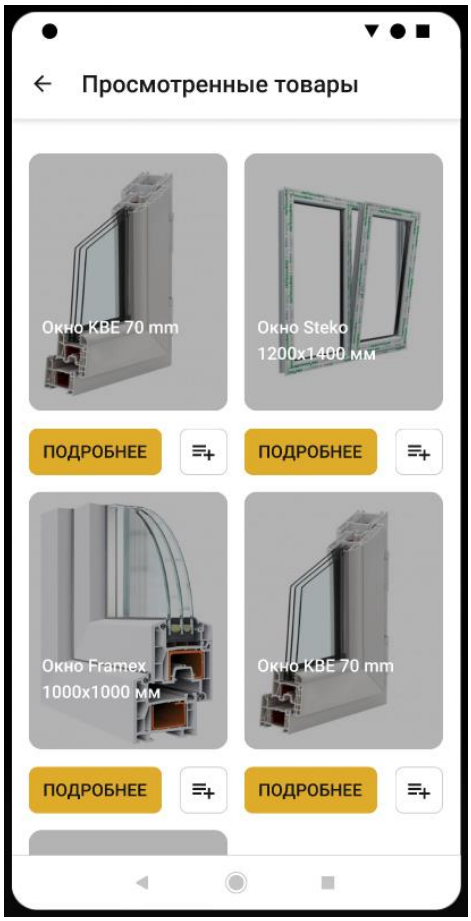

Рисунок 4.23 – Одна із підбірок в додатку

Після натискання «Подробнее» на будь-якому із відображених товарів, користувача буде переправлено на сторінку «Аналитика», де він матиме змогу переглянути статистику за обраним товаром та почитати звіт.

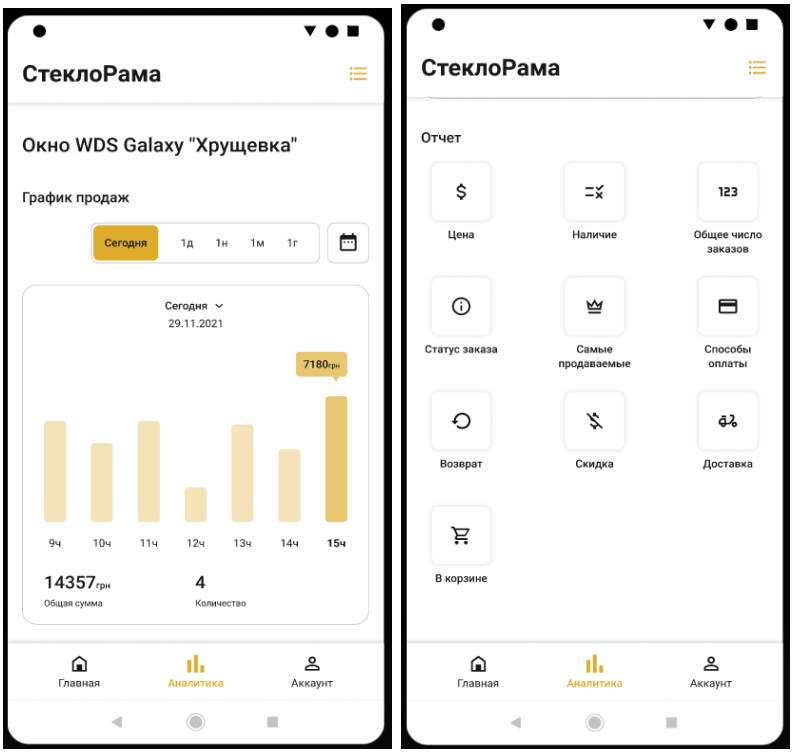

Рисунок 4.24 – Аналітика обраного товару

За замовчуванням всі сувої звіту приховані. Користувачеві потрібно лише натиснути на відповідний пункт звіту і він може переглянути інформацію всередині цього пункту.

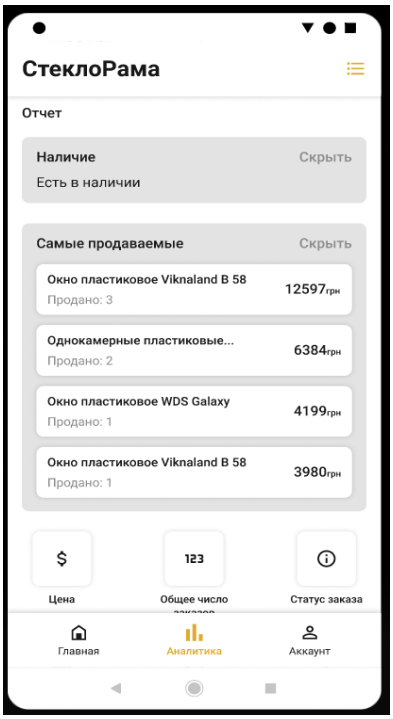

Рисунок 4.25 – Відображення інформації всередині пункту звіту

При бажанні, користувач може переглянути весь звіт. На рисунку 4.26 показано всі пункти звіту в розгорнутому вигляді.

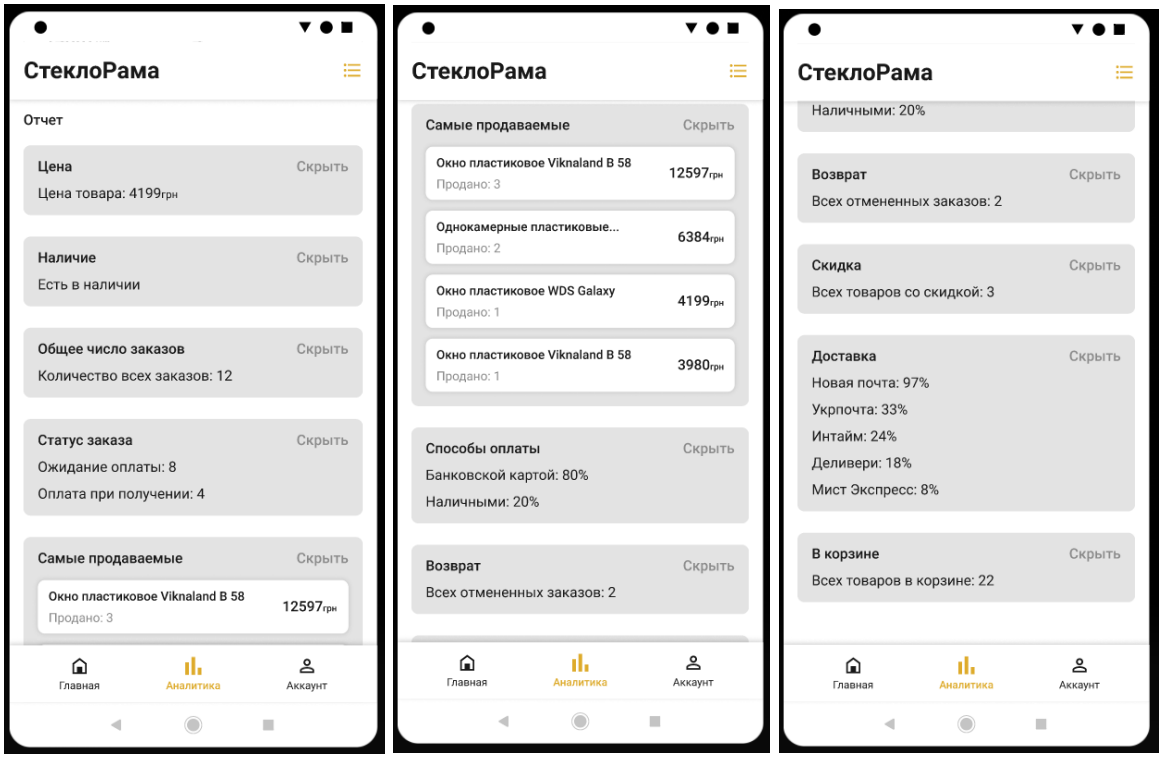

Рисунок 4.26 – Весь перелік пунктів звіту

За замовчуванням, пункт меню «Аналитика» має наступний вигляд: короткий опис того, що представлено на даній сторінці, графік продаж та звіт. Також важливо розуміти, що за замовчуванням у звіті відсутні декілька пунктів. Це все тому, що в даному випадку звіт складається по всім товарам, а не по конкретному (обраному).

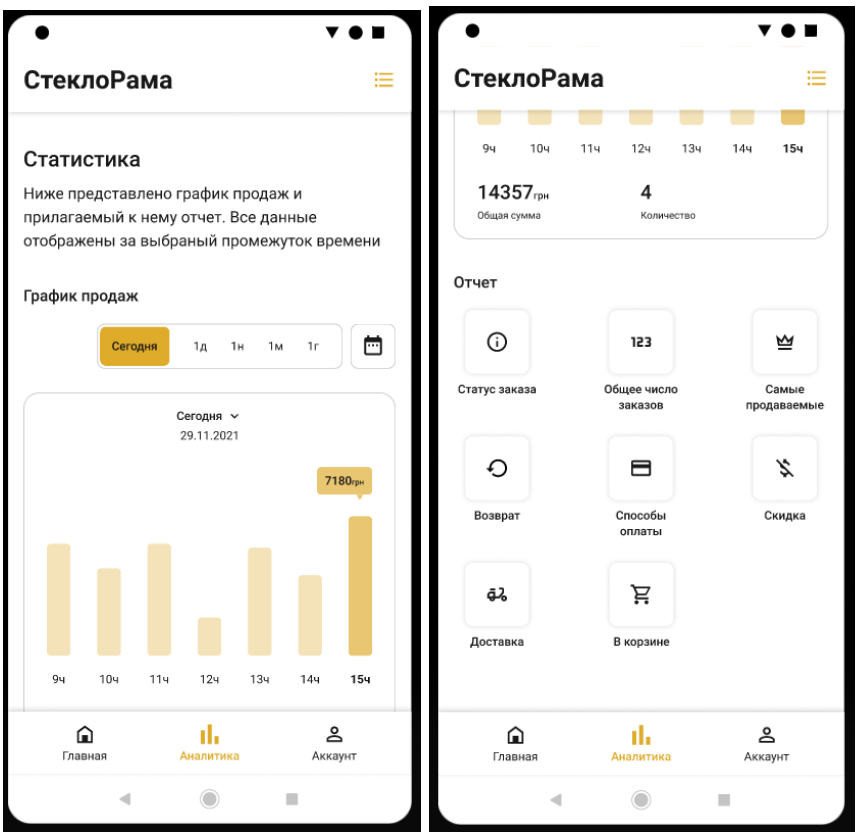

Рисунок 4.27 – Пункт меню «Аналитика» за замовчуванням

Далі наведено рисунки, на яких зображено процес встановлення часу для відображення даних за обраний проміжок часу. За замовчуванням обрано «Сегодня». Це означає те, що дані на графіку та в звіті оновлюються за кожну пройдену годину в реальному часі.

Алгоритм обирання довільної дати чи проміжку часу наступний: користувач натискає на один із представлених елементів «1д», «1н», «1м», «1г» або «задати довільний проміжок часу» (на рисунку 4.27 зображено в вигляді іконки календаря), після чого з’являється календар і користувач обирає бажану дату чи проміжок часу.

На рисунку 4.28 відображено обирання елементу 1д та наступний вибір довільної дати. Після чого користувачу буде представлено дані за обрану дату. Щоб змінити дату, користувачу необхідно лиши натиснути відповідний елемент на графіку, після чого знову з’явиться календар, який надасть змогу обрати іншу дату.

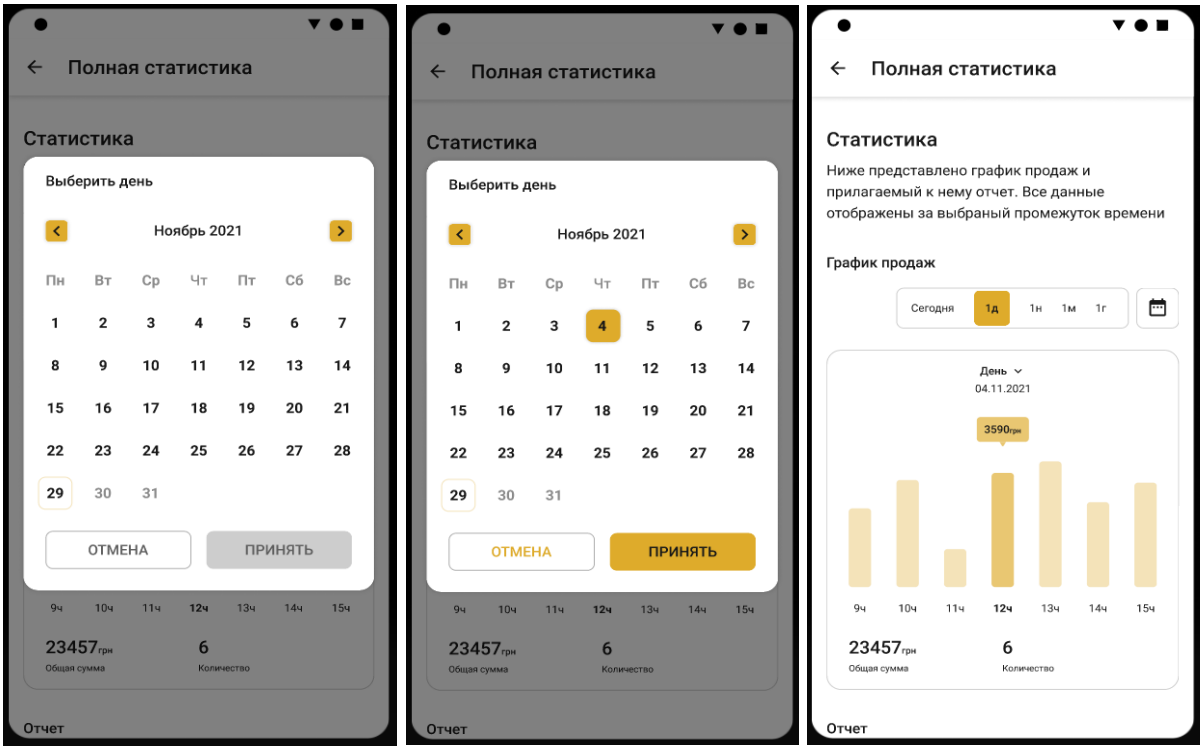

Рисунок 4.28 – Обрано елемент відображення даних «1д»

На рисунку4.29 наведено відображення даних за один тиждень.

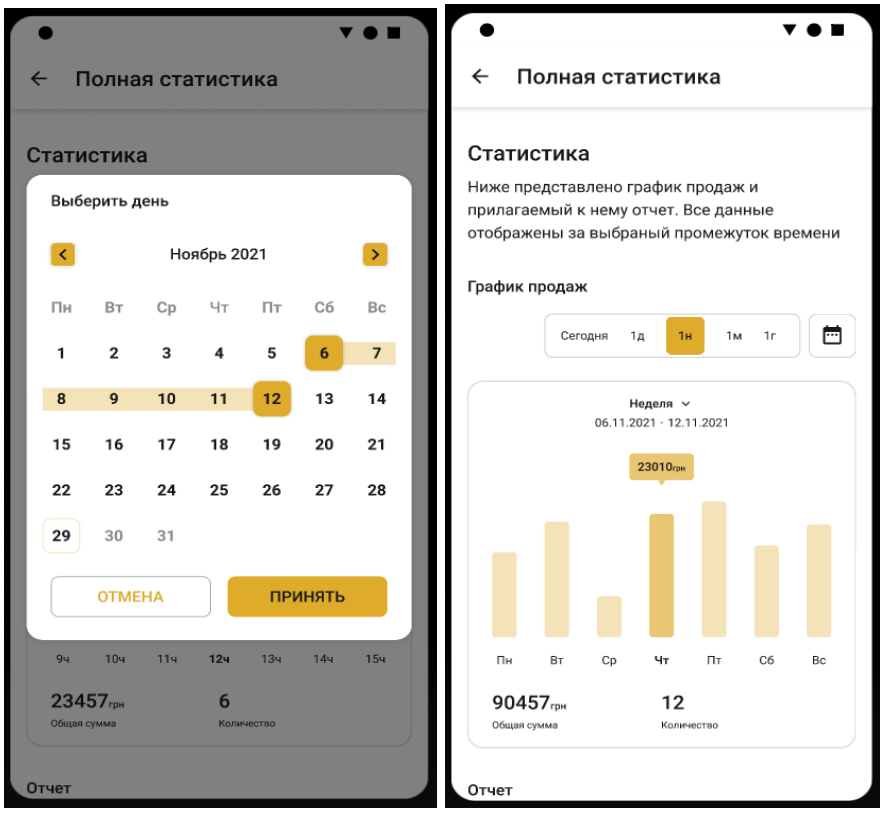

Рисунок 4.29 – Обрано елемент відображення даних «1н»

На наступному рисунку зображено відображення даних за один місяць. Алгоритм такий же, користувач обирає при бажанні рік, після чого обирає місяць.

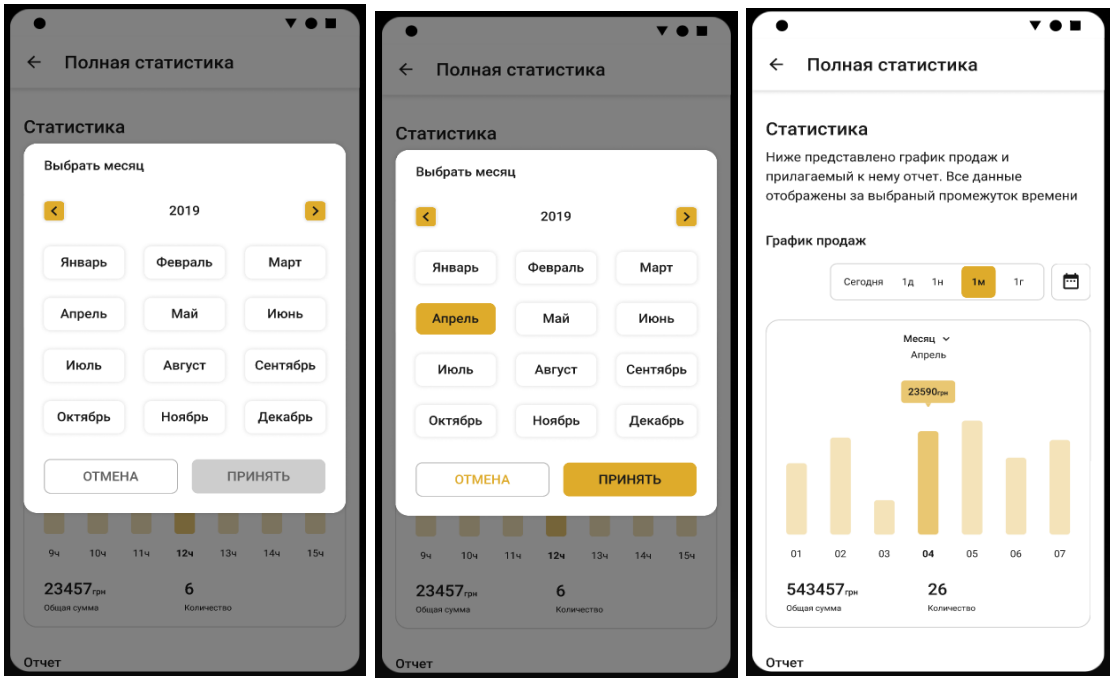

Рисунок 4.30 – Обрано елемент відображення даних «1м»

На наступному рисунку зображено відображення даних за один рік.

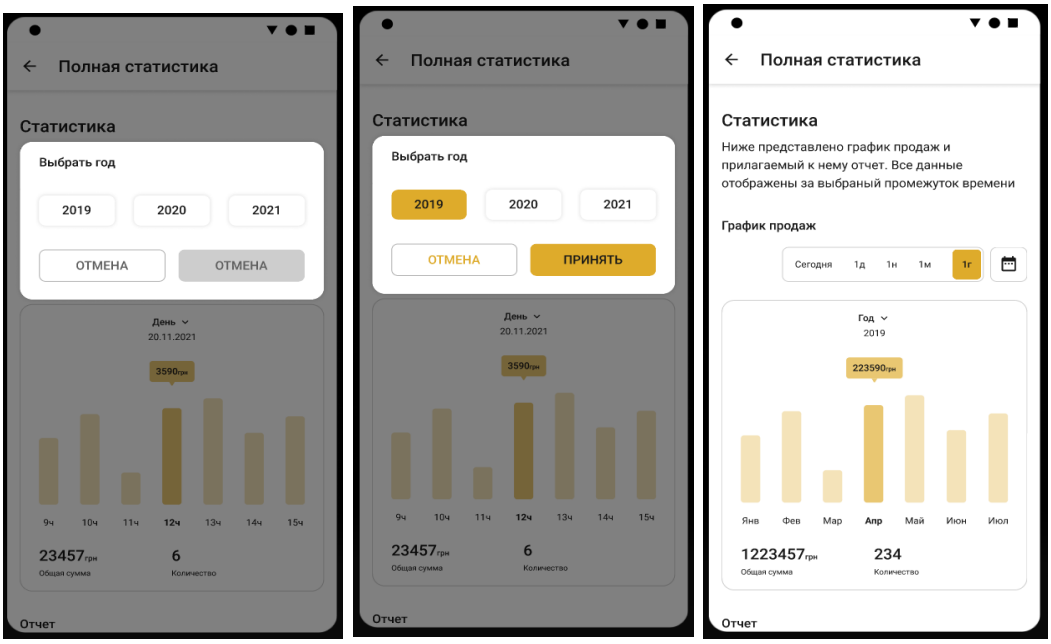

Рисунок 4.31 – Обрано елемент відображення даних «1г»

І нарешті останнім елементом обирання часу є довільне задання проміжку часу.

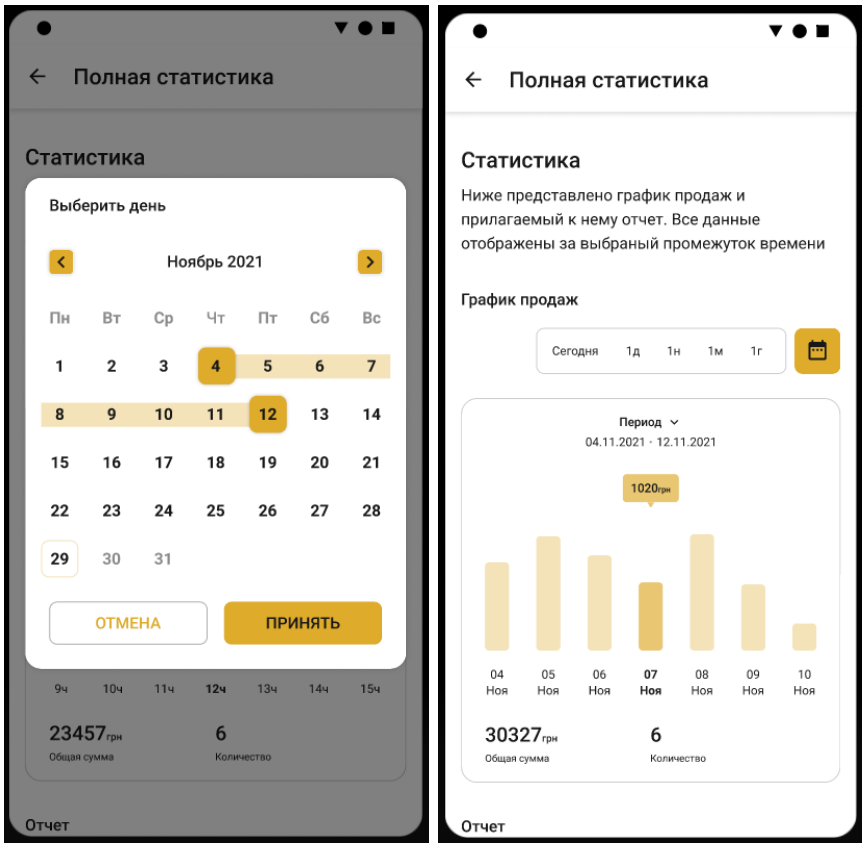

Рисунок 4.32 - Обрано елемент відображення даних «задати довільний проміжок часу»

Далі опишемо роботу з обліковим записом. Додаток підтримує редагування введених при реєстрації. Все, що вимагається від користувача – це зайти в пункт меню «Аккаунт», вибрати функцію «Редактировать профиль» і внести бажані корегування в наявні дані. Також в даному пункті меню можна побачити функцію «Список товаров», про який вже було описано вище, «Смена пароля» и функцію «Выйти». На подальших рисунках продемонстровано редагування облікового запису, зміну паролю та виходження з додатку. На рисунку 4.33 відображено зміну фото.

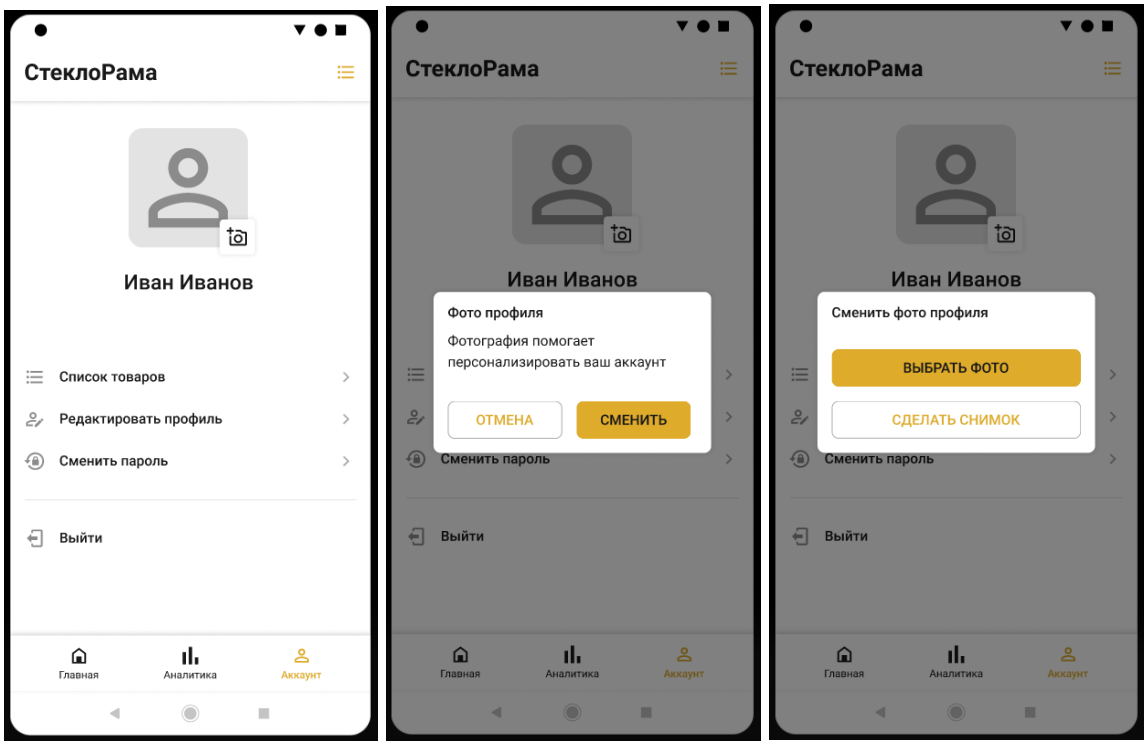

Рисунок 4.33 – Вибір нового фото профілю

Після того як користувач обрав бажане фото, воно відображається в додатку.

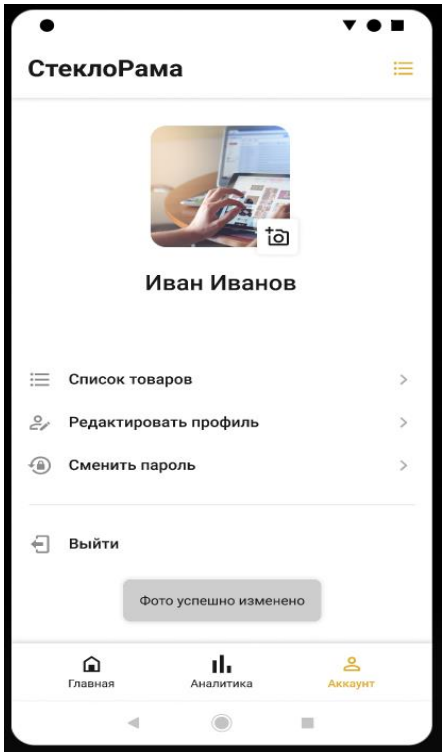

Рисунок 4.34 – Успішна зміна фото

Тепер перейдемо до опису редагування облікового запису. На рисунку 4.35 показано сторінку редагування профілю за замовчуванням.

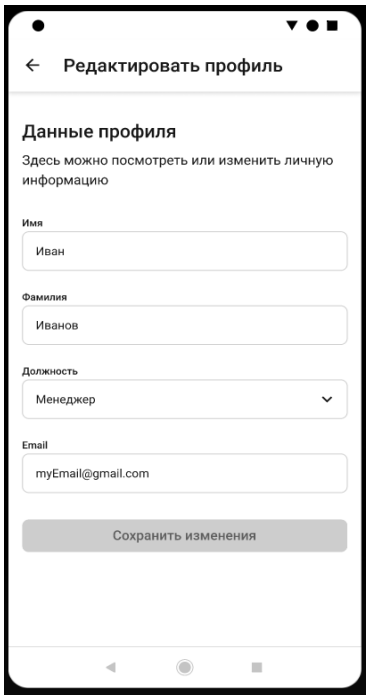

Рисунок 4.35 – Сторінка редагування профілю за замовчуванням

Якщо користувач ввів некоректні дані, додаток повідоме про це. Зміни не будуть внесені до тих пір, поки не зникне помилка.

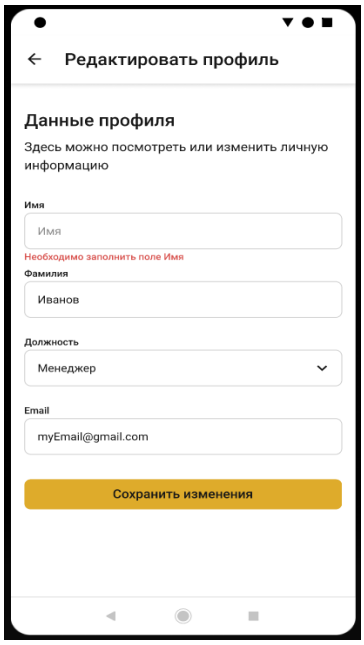

Рисунок 4.36 – Помилки при заповненні даних

Після внесення всіх бажаних змін, користувач може зберегти виконані маніпуляції та перейти в «Аккаунт».

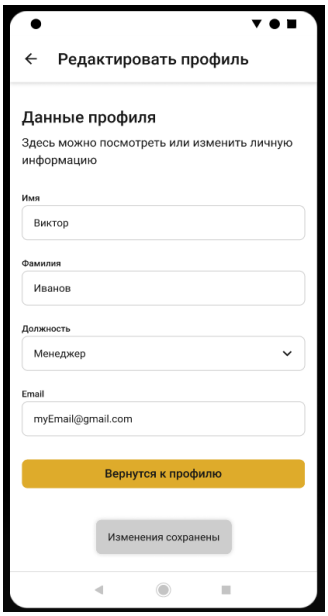

Рисунок 4.37 – Успішна зміна даних облікового запису

Якщо за будь-яких обставин користувач захотів змінити пароль, йому достатньо перейти до Акаунту та вибрати функцію Змінити пароль. Після цього користувачеві буде доступний перелік з операцій, які необхідно пройти для успішної зміни пароля. На рисунку 4.37 відображено сторінку зміни паролю за замовчуванням.

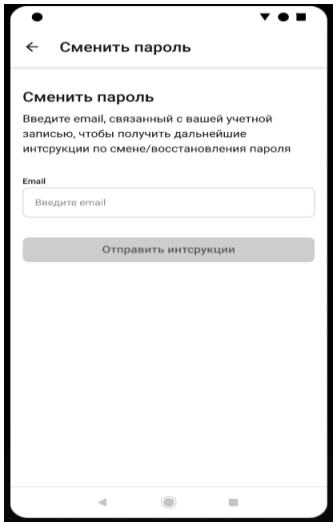

Рисунок 4.38 – Сторінка зміни паролю за замовчуванням

Користувачу необхідно ввести електронну пошту, яка прив’язана до облікового запису. Якщо пошту введено без помилок, система відобразить повідомлення, в якому сказано, що на дану пошту надіслано листа із подальшими інструкціями для зміни паролю. В інакшому випадку, якщо було введено невірну електронну пошту, користувач отримає повідомлення про помилку.

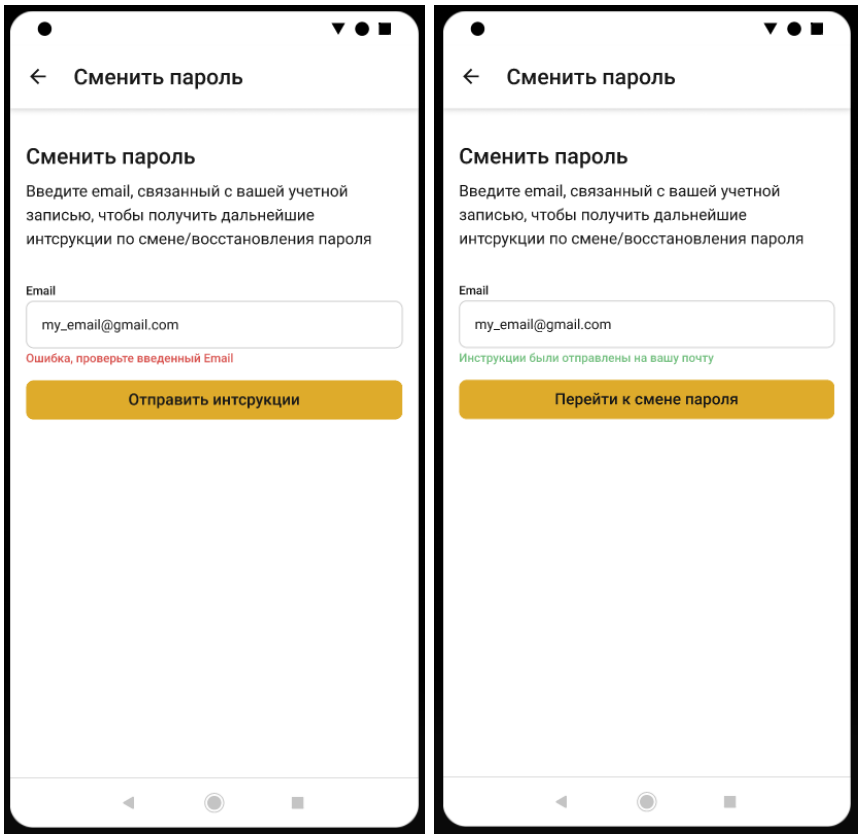

Рисунок 4.39 – Помилка та успіх при заповненні поля email

При натисканні кнопки «Перейти к смене пароля» користувач переходить на сторінку, де йому пропонується вести новий пароль та підтвердити його.

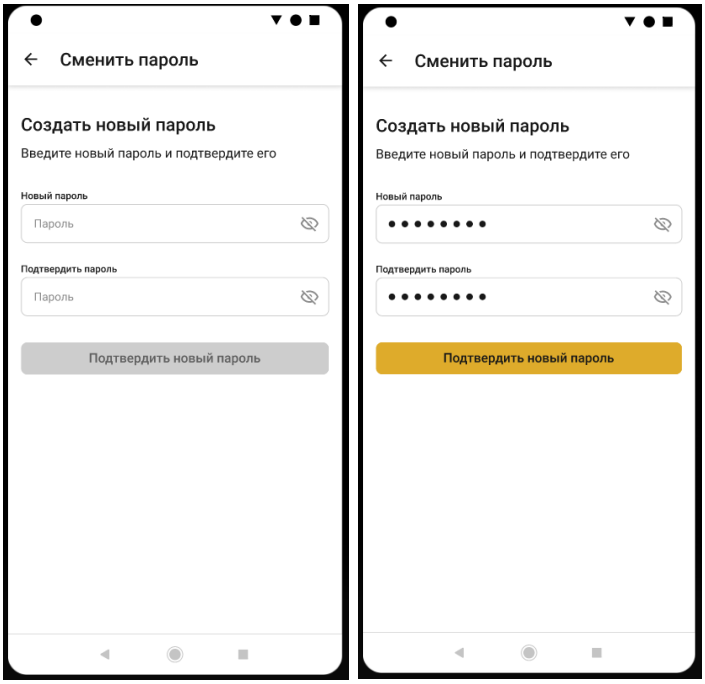

Рисунок 4.40 – Введення та підтвердження нового паролю

Якщо користувач при підтверджені паролю ввів неправильно дані, додаток повідомить йому про це та запропонує перевірити правильність введеного паролю.

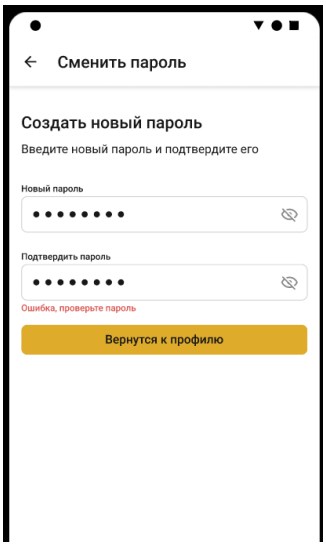

Рисунок 4.41 – Помилка при підтвердженні паролю

Після всіх маніпуляцій зі зміною паролю, додаток повідомить користувача про успішну зміну паролю і користувач зможе повернутися до облікового запису.

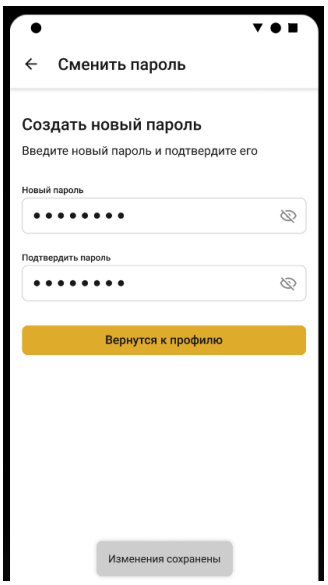

Рисунок 4.42 – Успішна зміна паролю

Користувач завжди може вийти з додатку. Для цього необхідно перейти в «Аккаунт» та натиснути на кнопку «Выйти».

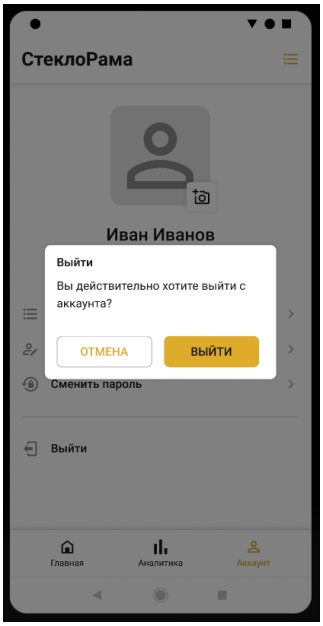

Рисунок 4.43 – Вихід з додатку

Таким чином, поставлена мета роботи виконана – розроблена інформаційна система у вигляді мобільного додатку для обліку замовлень інтернет магазину, який виконує задані функціональні вимоги.

ВИСНОВКИ

Магістерська робота присвячена розробці інформаційної системи обліку замовлень пластикових вікон.

На початку роботи над кваліфікаційною роботою було сформульовано мету проекту за допомогою методології SMART, що дає розуміння і можливість прогнозувати досягнення мети. Завдання, яке виконує система SMART в розроблювальному додатку – збільшити ймовірність досягнення бажаного результату.

В ході написання роботи наведено обґрунтування актуальності створюваного проекту та представлено аналіз останніх досліджень у сфері інформаційних технологій та бізнесу. Проведений огляд аналогічних рішень описує існуючі проекти, які пропонують розв’язання схожих задач. В результаті аналізу виявлено переваги, які потрібно реалізувати у додатку.

Аналіз предметної області дозволив сформулювати постановку задачі дослідження, виокремити мету роботи, задачі для її досягнення та опис вимог, котрим повинен відповідати створюваний додаток.

WBS є інструментом, що дозволяє керівнику проекту отримати чітку картину кінцевого та всіх проміжних результатів проекту. Поєднання робочої та організаційної структур дає можливість інтегрувати, планувати, контролювати роботу та порівнювати її виконання протягом роботи над проектом. OBS у свою чергу використовується для визначення обов’язків з управління проектами.

Для наочного зображення етапів проекту та розбиття його на завдання меншого розміру для зручності керування використовується діаграма Ганта. Створена діаграма відображає зв’язок між датами початку та завершення завдань, контрольними точками та залежними завданнями.

Після описана ймовірність настання ризиків, здатних надати негативний чи позитивний вплив на цілі розроблювального проекту.

Для наочного відображення функціональних процесів в додатку використано методологію IDEF. Спочатку створена контекстна діаграма, яка показує процес в цілому, далі цей процес декомпозовано, щоб виявити функціональні залежності всередині загального процеса (контекстна діаграма). Щоб відобразити взаємодію користувача та системи створена діаграма варіантів використання, на якій графічно зображено взаємодію акторів з додатком, та кому яка роль відведена для злагодженого функціонування додатку.

Також було описано базу даних зі всіма таблицями, котрі було створено в ході розробки мобільного застосунку.

Представлено опис реалізації інформаційної системи обліку замовлень пластикових вікон описано складові етапи розробки, програмну реалізацію, та опис користування додатком.

На даний момент можна зробити висновок, що розроблена система обліку замовлень є оптимальним рішенням проблеми, оскільки по-перше, не вимагає додаткових витрат на купівлю додаткового ПЗ, по-друге, містить мінімальний необхідний набір функцій, що дозволяє вирішувати конкретну проблему.

Практичним значенням розробки інформаційної системи є скорочення обсягів ручної роботи, зниження ймовірності помилок, збільшення продуктивності праці співробітників інтернет-магазину з появою додаткових одиниць робочого часу.

Використання інформаційної системи обліку замовлень дозволяє прискорити доступ до даних, а також, що досить важливо, скоротити кількість інформації на паперових носіях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Информационные системы: стаття. URL: https://www.accaglobal.com/cis/ru/qualifications/russian-language-advanced-diploma/Learningresources/PM/Technical_articles/Information_systems.html.

2. Интернет-магазин: как это работает?: стаття. URL: https://buhgalter911.com/news/news-1051228.html.

3. Операционная система Android: история создания и развития: стаття. URL: https://cyberleninka.ru/article/n/operatsionnaya-sistema-android-istoriya-sozdaniya-i-razvitiya-razrabotka-prilozheniy-dlya-platformy-android/viewer.

4. Автоматизация системы заказов на малом предприятии: стаття. URL: https://cyberleninka.ru/article/n/informatsionnaya-sistema-k-voprosu-opredeleniya-ponyatiya/viewer .

5. Розроботка информационной системы для учета и сопровождения заказов компании по производству печатной продукции: стаття. URL: https://cyberleninka.ru/article/n/razrabotka-informatsionnoy-sistemy-dlya-ucheta-i-soprovozhdeniya-zakazov-kompanii-po-proizvodstvu-pechatnoy-produktsii/viewer.

6. Автоматизированная информационная система для производителей пластиковых окон: стаття. URL: https://cyberleninka.ru/article/n/avtomatizirovannaya-informatsionnaya-sistema-dlya-proizvoditeley-plastikovyh-okon/viewer.

7. Целесообразность использования информационных систем на предприятии: стаття. URL: https://e-koncept.ru/2017/970401.htm.

8. Мобильные приложения для бизнеса: исллedования. URL: https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%9C%D0%BE%D0%B1%D0%B8%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B4%D0%BB%D1%8F_%D0%

B1%D0%B8%D0%B7%D0%BD%D0%B5%D1%81%D0%B0._%D0%9E%D0%B1%D0%B7%D0%BE%D1%80_TAdviser.

9. Как мобильные технологии изменяют предприятия: стаття. URL: https://www.sybase.ru/system/files/pdf/harvard_business_review_how_mobility_is_changing_the_enterprise_russian.pdf.

10. Как разработка мобильного приложения для бизнеса способствует успеху компании: стаття. URL: https://smartum.pro/ru/blog-ru/kak-razrabotka-mobilnogo-prilozheniya-sposobstvuyet-uspekhy-kompanii/.

11. Информационные технологии и их использование в управлении бизнесом: стаття. URL: https://vc.ru/trade/72668-informacionnye-tehnologii-i-ih-ispolzovanie-v-upravlenii-biznesom.

12. Что такое РемОнлайн и для чего он нужен?: документація. URL: https://help.remonline.ru/ru/articles/3582321-%D1%87%D1%82%D0%BE-%D1%82%D0%B0%D0%BA%D0%BE%D0%B5-%D1%80%D0%B5%D0%BC%D0%BE%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD-%D0%B8-%D0%B4%D0%BB%D1%8F-%D1%87%D0%B5%D0%B3%D0%BE-%D0%BE%D0%BD-%D0%BD%D1%83%D0%B6%D0%B5%D0%BD.

13. CRM для управления торговлей МойСклад: стаття. URL: https://www.clouderp.ru/tool/794/.

14. Модули CRM системы: документація. URL: https://qsystem.com.ua/crm-sistema/opisanie-modulej-crm-sistemy/#:~:text=CRM%20Qsystem%20%E2%80%94%20%D1%8D%D1%82%D0%BE%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D0%B4%D0%BB%D1%8F,%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%D1%82%D1%8C%20%D1%83%D0%B4%D0%B0%D0%BB%D0%B5%D0%BD%D0%BD%D0%BE%2C%20%D1%81%D0%BB%D0%B5%D0%B4%D0%B8%D1%82%D1%8C%20%D0%B7%D0%B0%20%D0%BF%D1%80%D0%BE%D0%B4%D0%B0%D0%B6%D0%B0%D0%BC%D0%B8.

15. Класс365 – торговля, склад, CRM: стаття. URL: https://www.audit-it.ru/software/trade/684330.html.

16. PLAY MARKET – что это?: стаття. URL: https://hs-store.ru/articles/interesnoe/play-market-chto-eto/.

17. История Google: стаття. URL: https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8F_Google.

18. Draw.io – бесплатное стредство для создания блок-схем, информатики, прототипов: навчальний блог. URL: https://el-blog.ru/draw-io/.

19. Modular Design Frameworks: книга. URL: https://link.springer.com/book/10.1007%2F978-1-4842-1688-0.

20. Figma – код и дизайн стали ближе: стаття. URL: https://vc.ru/design/159990-figma-kod-i-dizayn-stali-blizhe.

21. Васильев А.Н. Java. Объектно-ориентированное программирование. Учебное пособие. Стандарт третьего поколения. – СПб., Питер, 2014. – 400 с.

22. Как работают SDK и API: стаття. URL: https://vc.ru/dev/224259-kak-rabotayut-sdk-i-api.

23. James Steele, Nelson To. The Android Developer's Cookbook: Building Applications with the Android SDK. – Pearson Education, Inc. – 2011. - 316 p.

24. Sunny Kumar Aditya, Vikash Kumar Karn. Android SQLite Essentials: книга. – Packt Publishing, 2014. – 101 p.

25. Кириллов В.В. Введение в реляционные базы данных //В.В. Кириллов, Г.Ю.Громов. - СПб., БХВ-Петербург, 2009. – 464 с.:

26. Barbara Hohensee. Android for beginners. Developing apps using Android Studio: Babelcube Inc. – Babelcube Inc., 2014. – 406 p.

27. О компании JetBrains: стаття. URL: https://www.jetbrains.com/ru-ru/company/.

28. IntelliJ IDEA, the most intelligent Java IDE: стаття. URL: https://alfasoft.com/products/wp-content/uploads/Comparisons_IntelliJIDEA-1.pdf.

29. Методология IDEF0: стаття. URL: https://itteach.ru/bpwin/metodologiya-idef0.

30. UML для бизнес-моделирования: зачем нужны диаграммы процессов: стаття. URL: https://evergreens.com.ua/ru/articles/uml-diagrams.html.

31. Побудова діаграм варіантів використання (UseCase Diagrams): URL: http://www.tsatu.edu.ua/kn/wp-content/uploads/sites/16/laboratorna-robota-5-diahramy-variantiv-vykorystannja.pdf.

32. ER: диаграммы сущность – связь: стаття. URL: https://dl.sumdu.edu.ua/textbooks/90002/400789/ER_Modeling.pdf.

33. Конкурентный анализ и маркетинговое исследование: соотношение понятий и процедур проведения: дослідження. URL: https://openbooks.itmo.ru/read_economics/6899/6899.pdf.

34. Методы сбора информации: стаття. URL: https://www.nlb.by/content/bibliotekaryam/nauchnye-issledovaniya-metodika-provedeniya/podgotovitelnyy-etap/metod-sbora-informatsii/.

35. Как ставить цели по SMART: стаття. URL: https://uprav.ru/blog/kak-stavit-tseli-po-smart/.

36. Структура декомпозиции работ WBS: стаття. URL: https://www.cfin.ru/itm/project/wbs.shtml.

37. Organization Breakdown Structure (OBS): стаття. URL: https://uplandsoftware.com/psa/resources/glossary/organization-breakdown-structure-obs/.

38. Диаграмма Ганта для вас в новинку? Начните отсюда: стаття. URL: https://asana.com/ru/resources/gantt-chart-basics.

ДОДАТОК А
ПЛАНУВАННЯ РОБІТ

А.1 Ідентифікація мети проекта

Метою дипломної роботи є розробка інформаційної системи у вигляді мобільного додатка для ведення обліку замовлень пластикових вікон. Ціллю виконання роботи є можливість за допомогою додатку автоматизувати процес ведення обліку замовлень та формувати необхідні статистичні звіти у графічному та текстовому вигляді. Одною із поставлених задач, для досягнення мети є необхідність в заміні всіх можливих дій, котрі стосуються обліку замовлень, що виконуються руками на автоматизований процес. Деталізація мети проекту методом SMART [35] наведена в таблиці 1.1.

Таблиця А.1 – Деталізація мети методом SMART

Specific (конкретна)	Розробка інформаційної системи обліку замовлень пластикових вікон
Measurable (вимірювання)	Проект вважається виконаним лише за умови, що замовник затвердив його. Додаток повинен містити тільки ту інформацію, яка відносяться безпосередньо до заданого завдання.
Achievable (досяжна, узгоджена)	Розробник повинен володіти знаннями з ООП, тому що мову програмування було обрано Java . Також необхідно володіти мовою

Продовження таблиці А.1

	запитів SQLite, для проектування та написання запитів до БД. Задля побудови графічної частини додатку, розробнику необхідно володіти знаннями для розробки інтерфейсів та прототипування. Для останнього вибрано онлайн-сервіс Figma.
Relevant (реалістична)	Поставлену мету звичайно можна реалізувати тому, що на даний момент існує велика кількість ПЗ та ресурсів, які за допомогою своїх функцій, скорочують час побудови, проектування, розробки проекту.
Time-framed (обмежена в часі)	Обмеженість в часі зумовлена рішенням замовника, щоб якомога швидше отримати програмний продукт (додаток).

А.2 Планування змісту робіт ІТ проекту

Щоб оптимально спланувати виконання робіт у найкоротші терміни та без затримок, необхідно визначити порядок виконання робіт та терміни виконання кожної роботи.

Структурна декомпозиція роботи **(**ієрархічна структура розбиття роботи, WBS**)** – це ієрархічна побудова роботи, побудована з метою логічного розподілу всіх робіт для виконання проекту та подана у графічній формі [36]. Структура декомпозиції робіт являє собою поєднання декількох рівнів, кожен з яких в кінцевому рахунку декомпозується між розподілом роботи попереднього рівня на його компоненти. Компонент найнижчого рівня – являє

собою склад робіт, або так званий робочий пакет. Ключова перевага цього процесу – це надання структурованого бачення того, що потрібно досягти в кінцевому рахунку. Повна WBS структура наведена на рисунку 1.1.

Як правило наступним кроком є створення OBS структури. Отже далі буде представлено саме її опис. Організаційна структура проекту (OBS) представляє собою графічне відображення учасників проекту (фізичних та юридичних осіб) та їх відповідальних осіб, які беруть участь у створенні проекту [37]. На найвищому рівні OBS структури розташовані менеджер та команда з управління проектами; на наступному рівні йдеться опис розробників. Кінцевий рівень структури OBS представлений виконавцями (рис.1.2).

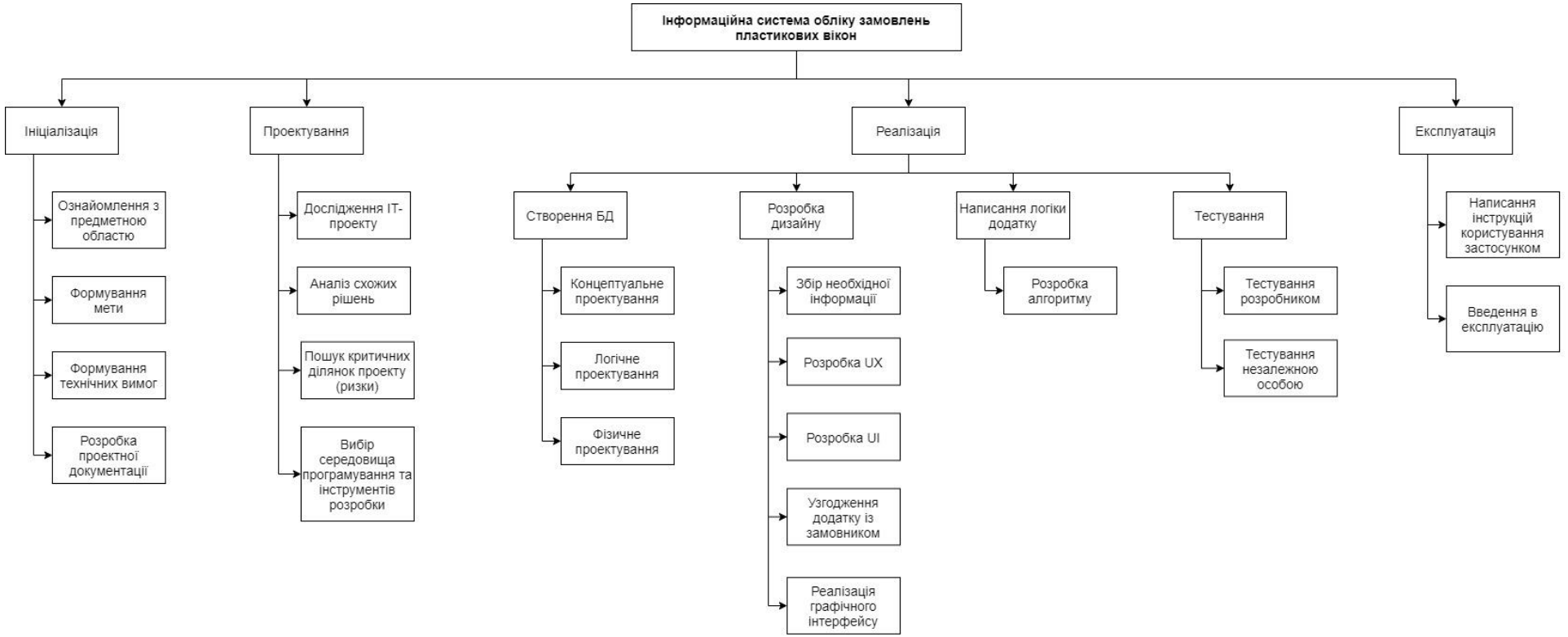

Рисунок А.1 – Структура проекту

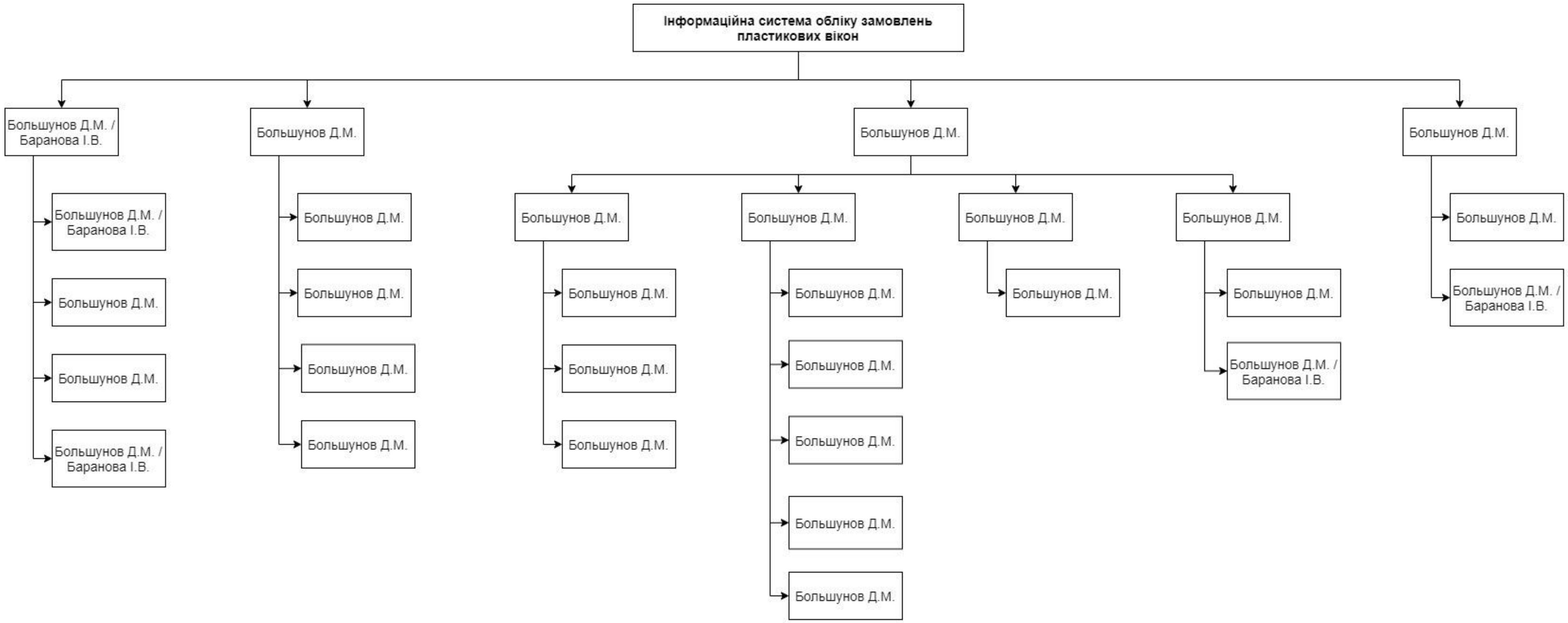

Рисунок А.2 – Організаційна структура проекту

А.3 Побудова календарного графіку ІТ проекту

Календарне планування полягає у складанні часової діаграми робіт та розподіл між роботами трудових ресурсів (виконавців). Результатом календарного планування є діаграма Ганта, що графічно відображає періоди виконання робіт.

Діаграма Ганта – це відома версія діаграми (придумана Генрі Ганттом), використання якої необхідне для планування та контролю за виконанням проекту [38]. Така інтерактивна мережева діаграма існує та використовується майже у всіх системах управління проектами. На схемі (структурі) показано завдання та стадії плану з урахуванням часу на їх виконання. Завдання на діаграмі можуть існувати залежно одне від одного, наприклад, одне завдання може з’явитися лише в кінці другого. Крім того, на діаграмі може відображатися відсоток виконання завдання та відповідальність за його виконання.

	Інформаційна система обліку замовлень пластикових вікон	67 days	Wed 01/09/21	Thu 02/12/21	
	Ініціалізація	6 days	Wed 01/09/2	Wed 08/09/2	
	Проектування	7 days	Thu 09/09/2	Fri 17/09/21	2
	Реалізація	52 days	Mon 20/09/2	Tue 30/11/2	7
	Створення БД	9 days	Mon 20/09/2	Thu 30/09/2	7
	Розробка дизайну	27 days	Fri 01/10/21	Mon 08/11/21	13
	Написання логіки додатку	14 days	Tue 09/11/21	Fri 26/11/21	17
	Тестування	2 days	Mon 29/11/2	Tue 30/11/2	23
	Експлуатація	2 days	Wed 01/12/2	Thu 02/12/2	12

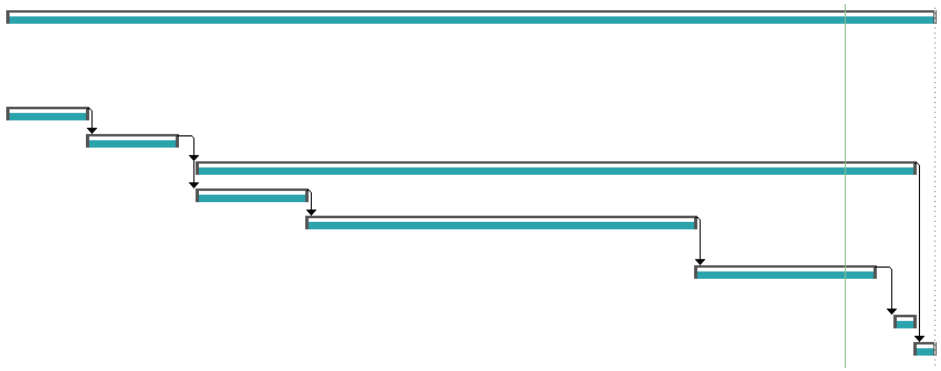

Рисунок А.3 – Діаграма Ганта

А.4 Планування ризиків ІТ проекту

Як відомо, кожен проект, в незалежності від масштабу, має свої певні ризики, основною причиною виникнення яких є невизначеність. Ризики можна класифікувати за двома основними видами, а саме: відомі та невідомі. Відомі - це ті ризики, які вже оцінені або ж визначені. Не відомі в свою чергу – це ті, котрі неможливо спрогнозувати.

Процес управління ризиками включає наступні етапи:

- ідентифікація;
- процес оцінювання ризиків, який включає в себе якісний та кількісний аналіз;
- заходи реагування на ризики;
- моніторинг заходів і ризиків.

Ризики проекту, що розроблюється представлені у таблиці А.2.

Таблиця А.2 – Класифікація ризиків

№	Назва ризику	Ймовірність	Величина втрат
1	Замовник довго не відповідає	1	4
2	Неоптимальний розподіл часу	4	4
3	Неполадки з Інтернет зв’язком	4	5
4	Недостатній рівень знань розробника	1	3
5	Замовника не влаштовує кінцевий результат	2	2
6	Захворювання розробника	2	3
7	Проблеми із Базою Даних	2	3
8	Платне програмне забезпечення	4	3
9	Вихід з ладу апаратного забезпечення	5	2
10	Затримка надання матеріалу	2	4

За ймовірністю виникнення:

- слабо ймовірнісні - 1;

- мало ймовірнісні - 2;
- імовірні - 3;
- досить імовірні - 4;
- майже імовірні - 5.

За величиною втрат:

- Мінімальна - 1;
- Низька - 2;
- Середня - 3;
- Висока - 4;
- Максимальна - 5.

Таблиця А.3 – Матриця ризиків

5		9			
4			8	2	3
3					
2		**5**	**6, 7**	**10**	
1			**4**	**1**	
Величина / Ймовірність	1	2	3	4	5

ДОДАТОК Б
КОДИ ФАЙЛІВ ПРОЄКТУ

Б.1 База даних

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DAT
E,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema steklorama
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `steklorama` DEFAULT CHARACTER SET utf8 ;
USE `steklorama` ;

-- -----------------------------------------------------
-- Table `steklorama`.`paymentMethods`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`paymentMethods` (
  `idpaymentMethods` INT NOT NULL,
  `banking card` INT NULL,
  `cash` INT NULL,
  PRIMARY KEY (`idpaymentMethods`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`delivery`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`delivery` (
  `iddelivery` INT NOT NULL,
  `nova_poshta` INT NULL,
  `ukr_poshta` INT NULL,
  `intime` INT NULL,
  `delivery` INT NULL,
  `meest_express` INT NULL,
  PRIMARY KEY (`iddelivery`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `steklorama`.`source`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`source` (
  `idsource` INT NOT NULL,
  `name` VARCHAR(255) NULL,
  PRIMARY KEY (`idsource`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`client`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`client` (
  `idclient` INT NOT NULL,
  `dirst_name` VARCHAR(45) NULL,
  `last_name` VARCHAR(45) NULL,
  `source_idsource` INT NOT NULL,
  PRIMARY KEY (`idclient`, `source_idsource`),
  INDEX `fk_client_source1_idx` (`source_idsource` ASC) VISIBLE,
  CONSTRAINT `fk_client_source1`
    FOREIGN KEY (`source_idsource`)
    REFERENCES `steklorama`.`source` (`idsource`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`status`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`status` (
  `idstatus` INT NOT NULL,
  `name` VARCHAR(255) NULL,
  PRIMARY KEY (`idstatus`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`sales_history`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`sales_history` (
  `idsales_history` INT NOT NULL,
  `sale_sum` DECIMAL(18,2) NULL,
```

```
  `active_from` DATETIME NULL,
  `active_to` DATETIME NULL,
  `status_idstatus` INT NOT NULL,
  PRIMARY KEY (`idsales_history`, `status_idstatus`),
  INDEX `fk_sales_history_status1_idx` (`status_idstatus` ASC) VISIBLE,
  CONSTRAINT `fk_sales_history_status1`
    FOREIGN KEY (`status_idstatus`)
    REFERENCES `steklorama`.`status` (`idstatus`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB


-- -----------------------------------------------------
-- Table `steklorama`.`sales`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`sales` (
  `idsales` INT NOT NULL,
  `number` VARCHAR(255) NULL,
  `date_created` DATETIME NULL,
  `date_modified` DATETIME NULL,
  `sale_sum` DECIMAL(18,2) NULL,
  `paymentMethods_idpaymentMethods` INT NOT NULL,
  `delivery_iddelivery` INT NOT NULL,
  `client_idclient` INT NOT NULL,
  `client_source_idsource` INT NOT NULL,
  `sales_history_idsales_history` INT NOT NULL,
  `sales_history_status_idstatus` INT NOT NULL,
  PRIMARY KEY (`idsales`, `paymentMethods_idpaymentMethods`, `delivery_iddelivery`, `client_idclient`, `client_source_idsource`, `sales_history_idsales_history`, `sales_history_status_idstatus`),
  INDEX `fk_sales_paymentMethods1_idx` (`paymentMethods_idpaymentMethods` ASC) VISIBLE,
  INDEX `fk_sales_delivery1_idx` (`delivery_iddelivery` ASC) VISIBLE,
  INDEX `fk_sales_client1_idx` (`client_idclient` ASC, `client_source_idsource` ASC) VISIBLE,
  INDEX `fk_sales_sales_history1_idx` (`sales_history_idsales_history` ASC, `sales_history_status_idstatus` ASC) VISIBLE,
  CONSTRAINT `fk_sales_paymentMethods1`
    FOREIGN KEY (`paymentMethods_idpaymentMethods`)
    REFERENCES `steklorama`.`paymentMethods` (`idpaymentMethods`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_sales_delivery1`
    FOREIGN KEY (`delivery_iddelivery`)
    REFERENCES `steklorama`.`delivery` (`iddelivery`)
```

```
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_sales_client1`
    FOREIGN KEY (`client_idclient` , `client_source_idsource`)
    REFERENCES `steklorama`.`client` (`idclient` , `source_idsource`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_sales_sales_history1`
    FOREIGN KEY (`sales_history_idsales_history` , `sales_history_status_idstatus`)
    REFERENCES `steklorama`.`sales_history` (`idsales_history` , `status_idstatus`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`category`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`category` (
  `idcategory` INT NOT NULL,
  `name` VARCHAR(255) NULL,
  PRIMARY KEY (`idcategory`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`good`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`good` (
  `idgood` INT NOT NULL,
  `name` VARCHAR(255) NULL,
  `price` DECIMAL(16,2) NULL,
  `discount_price` DECIMAL(16,2) NULL,
  `count` INT NULL,
  PRIMARY KEY (`idgood`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`sales_has_good`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`sales_has_good` (
  `sales_idsales` INT NOT NULL,
  `good_idgood` INT NOT NULL,
  PRIMARY KEY (`sales_idsales`, `good_idgood`),
```

```
  INDEX `fk_sales_has_good_good1_idx` (`good_idgood` ASC) VISIBLE,
  INDEX `fk_sales_has_good_sales_idx` (`sales_idsales` ASC) VISIBLE,
  CONSTRAINT `fk_sales_has_good_sales`
    FOREIGN KEY (`sales_idsales`)
    REFERENCES `steklorama`.`sales` (`idsales`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_sales_has_good_good1`
    FOREIGN KEY (`good_idgood`)
    REFERENCES `steklorama`.`good` (`idgood`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `steklorama`.`category_has_good`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `steklorama`.`category_has_good` (
  `category_idcategory` INT NOT NULL,
  `good_idgood` INT NOT NULL,
  PRIMARY KEY (`category_idcategory`, `good_idgood`),
  INDEX `fk_category_has_good_good1_idx` (`good_idgood` ASC) VISIBLE,
  INDEX `fk_category_has_good_category1_idx` (`category_idcategory` ASC) VISIBLE,
  CONSTRAINT `fk_category_has_good_category1`
    FOREIGN KEY (`category_idcategory`)
    REFERENCES `steklorama`.`category` (`idcategory`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_category_has_good_good1`
    FOREIGN KEY (`good_idgood`)
    REFERENCES `steklorama`.`good` (`idgood`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Б.1 action_bar_default

```
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:cardElevation="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingStart="16dp"
        android:paddingTop="17dp"
        android:paddingEnd="16dp"
        android:paddingBottom="17dp">

        <TextView
            android:id="@+id/logo"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:fontFamily="@font/roboto_bold"
            android:text="СтеклоРама"
            android:textColor="@color/mainTextColor"
            android:textSize="26sp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <ImageView
            android:id="@+id/listOfProducts"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/ic_mdi_format_list_bulleted"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:contentDescription="Список товаров"/>

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

```
</androidx.cardview.widget.CardView>
```

Б.1 pick_time

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_alignParentEnd="true">

        <LinearLayout
            android:id="@+id/pickTimeLayout"
            android:layout_width="wrap_content"
            android:layout_height="46dp"
            android:orientation="horizontal"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toTopOf="parent">

            <LinearLayout
                android:id="@+id/describedPeriods"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:background="@drawable/rounded_border_8_corner"
                android:gravity="center|start"
                android:orientation="horizontal"
                android:paddingStart="3dp"
                android:paddingTop="3dp"
                android:paddingEnd="24dp"
                android:paddingBottom="3dp">

                <androidx.appcompat.widget.AppCompatButton
                    android:id="@+id/currentDate"
                    android:layout_width="71dp"
```

```
    android:layout_height="wrap_content"
    android:background="@drawable/round_corner_8"
    android:fontFamily="@font/roboto_medium"
    android:text="Сегодня"
    android:textAllCaps="false"
    android:textColor="@color/mainTextColor"
    android:textSize="12sp" />

<TextView
    android:id="@+id/pickDayDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="24dp"
    android:fontFamily="@font/roboto_regular"
    android:text="1д"
    android:textAllCaps="false"
    android:textColor="@color/mainTextColor"
    android:textSize="12sp" />

<TextView
    android:id="@+id/pickWeekDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="24dp"
    android:fontFamily="@font/roboto_regular"
    android:text="1н"
    android:textAllCaps="false"
    android:textColor="@color/mainTextColor"
    android:textSize="12sp" />

<TextView
    android:id="@+id/pickMonthDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="24dp"
    android:fontFamily="@font/roboto_regular"
    android:text="1м"
    android:textAllCaps="false"
    android:textColor="@color/mainTextColor"
    android:textSize="12sp" />

<TextView
```

```
            android:id="@+id/pickYearDate"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="24dp"
            android:fontFamily="@font/roboto_regular"
            android:text="1r"
            android:textAllCaps="false"
            android:textColor="@color/mainTextColor"
            android:textSize="12sp" />

        </LinearLayout>

        <androidx.appcompat.widget.AppCompatImageButton
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:layout_gravity="center"
          android:layout_marginStart="8dp"
          android:layout_toEndOf="@id/describedPeriods"
          android:background="@drawable/rounded_border_8_corner"
          android:contentDescription="date range"
          android:padding="12dp"
          android:src="@drawable/ic_mdi_calendar_range" />

      </LinearLayout>

      <com.github.mikephil.charting.charts.BarChart
        android:id="@+id/barChart"
        android:layout_width="match_parent"
        android:layout_height="384dp"
        app:layout_constraintTop_toBottomOf="@id/pickTimeLayout"
        android:layout_marginTop="24dp"
        android:background="@drawable/rounded_border_16_corner"/>

    </androidx.constraintlayout.widget.ConstraintLayout>

</RelativeLayout>
```

Б.1 search_default

```
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
xmlns:card_view="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="32dp">

<androidx.cardview.widget.CardView
    android:id="@+id/cardView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginEnd="16dp"
    android:descendantFocusability="beforeDescendants"
    android:focusableInTouchMode="true"
    app:cardBackgroundColor="@color/primaryColor"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/cardView2"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    card_view:cardCornerRadius="8dp"
    card_view:cardPreventCornerOverlap="false"
    card_view:cardUseCompatPadding="true">

    <EditText
        android:id="@+id/searchBar"
        android:layout_width="match_parent"
        android:layout_height="match_parent"

        android:autofillHints="name"
        android:background="@color/primaryColor"
        android:drawableStart="@drawable/ic_search"
        android:drawablePadding="16dp"
        android:elevation="10dp"
        android:fontFamily="@font/roboto_regular"
        android:hint="Поиск по окнам"
        android:inputType="textPersonName"
        android:paddingStart="16dp"
        android:paddingEnd="0dp"
        android:textColor="@color/colorTones9"
        android:textSize="14sp" />

</androidx.cardview.widget.CardView>
```

```
    <androidx.cardview.widget.CardView
        android:id="@+id/cardView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        card_view:cardBackgroundColor="@color/shadowColor"
        card_view:cardCornerRadius="8dp"
        card_view:cardPreventCornerOverlap="false"
        card_view:cardUseCompatPadding="true">

        <ImageButton
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="@color/primaryColor"
            android:src="@drawable/ic_mdi_filter_outline"
            android:layout_gravity="center"
            android:padding="16dp"
            android:contentDescription="filter" />

    </androidx.cardview.widget.CardView>

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