Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С. Галущака»

**разработка** **мобильного приложения дЛЯ МАГАЗИНА МУЗЫКАЛЬНЫХ ИНСТРУМЕНТОВ**

Пояснительная записка к курсовому проекту

ПМ.01Разработка модулей программного обеспечения для компьютерных систем

МДК.01.03 Разработка мобильных приложений

НАТКиГ.201000.010.000ПЗ

Разработал:

студент группы ПР-21.106

Цветкова М.Э.

2023

**Содержание**

[Введение 3](#_Toc117112970)

[1 Исследовательский раздел 4](#_Toc117112971)

[1.1 Описание предметной области 4](#_Toc117112972)

[1.2 Образ клиента 5](#_Toc117112973)

[1.3 Сценарии 5](#_Toc117112974)

[1.4 Сбор и анализ прототипов 6](#_Toc117112975)

[2 Проектирование приложения 11](#_Toc117112976)

[2.1 UI/UX дизайн проекта 11](#_Toc117112977)

[2.2 Выбор технологии, языка и среды программирования 18](#_Toc117112978)

[3 Разработка мобильного приложения 19](#_Toc117112979)

[3.1 Разработка базы данных 19](#_Toc117112980)

[3.2 Разработка используемых плагинов 20](#_Toc117112981)

[3.3 Описание разработанных процедур и функций 21](#_Toc117112982)

[4 Тестирование 26](#_Toc117112983)

[4.1 Протокол тестирования дизайна приложения 26](#_Toc117112984)

[4.2 Протокол тестирования функционала приложения 26](#_Toc117112985)

[Заключение 29](#_Toc117112986)

[Библиография 30](#_Toc117112987)

[Приложение А 31](#_Toc117112988)

Введение

Музыка всегда являлась важной частью духовной жизни людей, например помогала людям, как в походе (сражении), поднимая дух воинов, так и простому народу для поднятия их настроения. Сейчас же музыка помогает не только в этом, но в выражении себя в том числе.

Музыка обостряет духовное состояние, раскрывает эмоции, чувства, переживания, настроения. Занятие музыкой расширяет сознание, культурно и духовно обогащает. Сила музыки в том, что она способна сделать человека счастливым.

За последние годы индустрия продаж музыкальных инструментов постепенно ушла в сеть. Причин для этого более чем достаточно. Прежде всего, магазин музыкальных инструментов в сети интернет существенно экономит средства при продаже поставляемой продукции.

Мобильное приложение по продаже музыкальных инструментов делает взаимодействие с магазином более удобным для клиента: в случае возникновения желания приобрести музыкальный инструмент пользователю не требуется выходить из дома, что облегчает приобретение. Клиент может заказать музыкальный инструмент с помощью приложения.

Целью курсового проекта является создание мобильного приложения для магазина, основной деятельностью которого является продажа музыкальных инструментов.

Задачами курсового проекта в связи с указанной целью являются:

* изучение предметной области мобильного приложения музыкальных инструментов;
* рассмотрение приложения с точки зрения пользователя для выявления функций приложения;
* разработка и реализация дизайна приложения;
* написание кода приложения;
* тестирование полученного продукта.

Объект исследования – мобильное приложение музыкальных инструментов.

Предмет исследования – изучение принципов функционирования и инструментов приложения.

# Исследовательский раздел

## Описание предметной области

Музыкантам музыка нужна, чтобы выразить свой внутренний мир с ее помощью, чтобы донести до слушателей свое видение мира. Слушателям музыка нужна, чтобы насладиться прекрасным, чтобы разделить чувства создателя музыки, постичь гармонию. Музыка – самый мощный по воздействию вид искусства. Поэтому музыка нужна врачам, чтобы избавлять пациентов от депрессии, тревожности, бессонницы. Также она нужна учителям, чтобы воспитывать в учениках чувство прекрасного. Музыка нужна деятелям других видов искусства, чтобы вдохновляться. Музыка нужна режиссерам, чтобы с ее помощью подчеркнуть особенности состояния героев. Всем людям музыка нужна для счастья и красоты.

В данном случае, предметной областью является магазин музыкальных инструментов. В магазин заходит потенциальный покупатель, который хочет купить товар. Покупатель может прийти как с целью купить определенный товар, так еще и не определится с выбором. В первом случае он, уже зная, что хочет приобрести, узнает у продавца консультанта о наличии необходимого ему товара, тот в свою очередь проверяет есть ли товар в зале и не найдя его ищет на складе. В случае если он нашел, то предоставляет товар клиенту. Во втором случае, когда клиент еще не определился, продавец консультант узнает, что клиент хочет приобрести и подбирает подходящий товар. Когда товар удовлетворяет запросу клиента, то продавец оформляет товар и покупатель получает гарантию с чеком.

Следует заострить внимание на потенциальных покупателях товара. Это, во-первых, учащиеся и педагоги музыкальных школ, училищ и консерваторий, где потребность в музыкальных инструментах имеется всегда. Во-вторых, это начинающие и профессиональные музыканты, чей энтузиазм и активность гораздо выше. Для тех и для других крайне важно, чтобы инструмент был качественным и в то же время не слишком дорогим. Именно эти две потребности и являются краеугольным камнем ценового планирования и, как следствие, разработки информационной системы.

## Образ клиента

Клиентами являются все те, кто интересуется музыкальными инструментами и дополнительными деталями к ним.

Спрос на музыкальные инструменты среди россиян заметно вырос за последние несколько месяцев. Как подсчитали аналитики Wikdbtrries для газеты «Газеты.Ru», продаж музыкальных инструментов увеличились более чем в 3,5 раза в штуках в августе-сентябре по отношению к аналогичному периоду прошлого года.

Так во время самоизоляции люди начали больше интересоваться музыкальными инструментами и спрос на приложение «магазин музыкальных инструментов» вырос.

## Сценарии

Молодой человек решил приобрести свой первый музыкальный инструмент и решил это сделать через мобильное устройство, для быстрого изучения лучшего инструмента, а также быстрой покупкой.

Труппа музыкантов ограничена в свободном времени, что мешает периодически обновлять износившиеся инструмент. Группе удобнее заказывать инструменты через приложение.

Консерватория решила обновить имеющиеся музыкальные инструменты, а для быстрой покупки они обратились к приложению, ведь оптовая закупка нуждается в доставке.

## Сбор и анализ прототипов

У данного мобильного приложения конкурентов немного, так как чаще магазины музыкальных инструментов имеют личные сайты, нежели приложения. В магазине приложений Google Play владельцев телефонов с операционной системой Android было найдено 2 конкурента. Сначала рассмотрим дизайн, ошибки и достижения приложения, имеющегося в Google Play – Музторг.

Музторг имеет не особо яркое цветовое решение приложения, преобладают оранжевый, чёрный и белый цвета.

Главная страница отсутствует.

У приложения есть шапка, информационная часть и навигационное меню в нижней части экрана. Шапку приложения составляет название магазина и объявление о скидке сезона. В навигационном меню имеются кнопки для открытия дополнительных окон экрана, такие как «Каталог», «Войти», «Корзина», «Чат». Внутри информационной части приложение предлагает выполнить вход, записаться в сервис и просмотреть напоминания.

На рисунке 1 представлена раздел «Каталог».

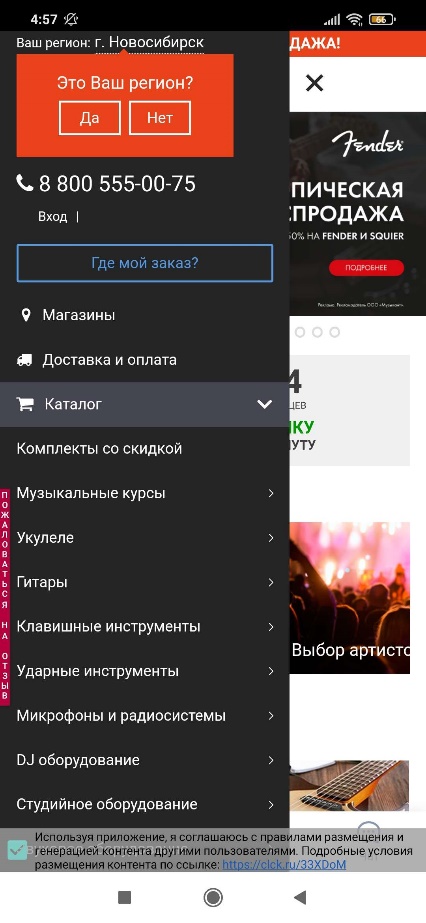


Рисунок 1 – Раздел «Каталог»

На данной странице можно выбрать нужный раздел среди представленных для поиска нужных инструментов и их составляющих. На рисунке 2 представлена раздел «Войти».

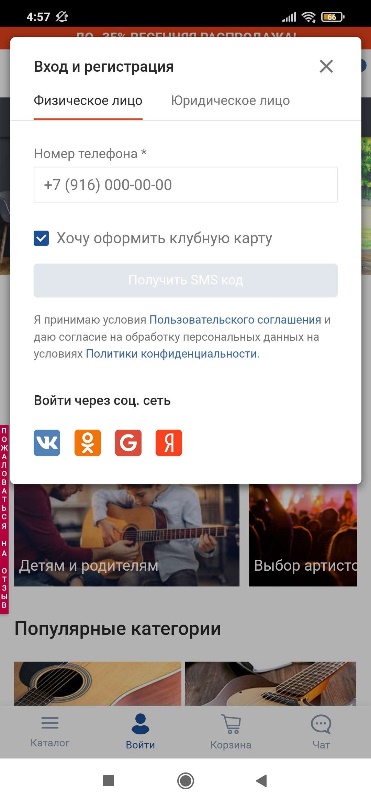


Рисунок 2 – Страница «Войти»

Внутри информационной части данной страницы имеется возможность войти или зарегистрироваться в приложении, что нужно для покупки инструментов и других предметов, связанных с ними.

На рисунок 3 представлена страница «Корзина»

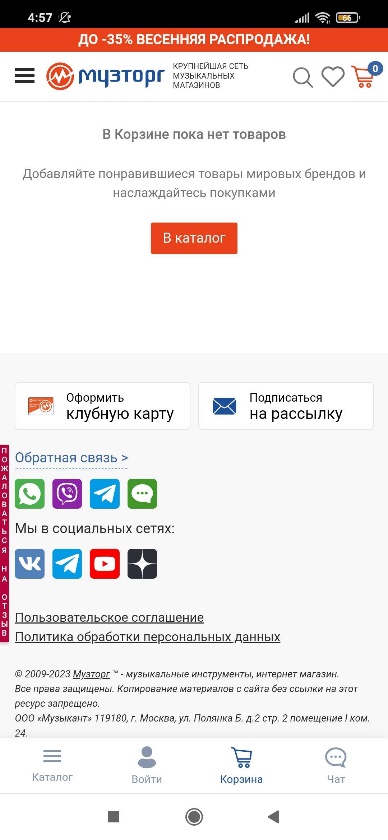


Рисунок 3 – Страница «Корзина»

Данная страница отличается от остальных своей пустотой. Так как на данный момент ничего не добавлено в «корзину» для покупки товара в будущем, страница остаётся пустой. На рисунке 4 представлена страница «Чат»

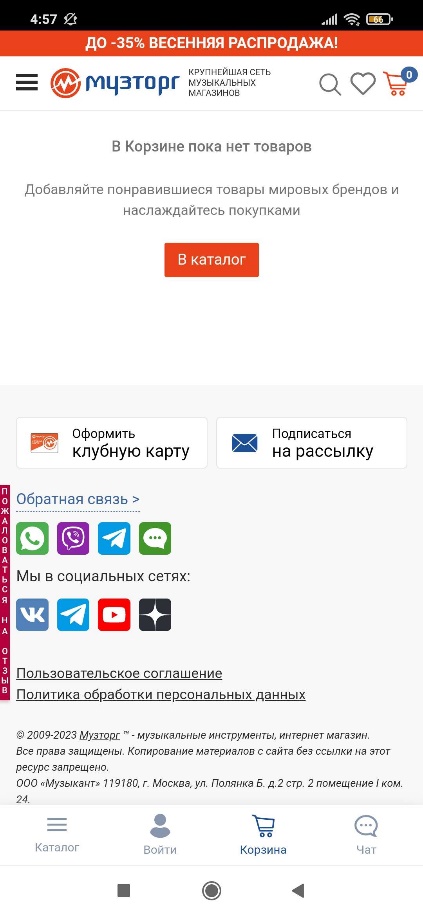


Рисунок 4 – Страница «Чат»

На данной странице имеется возможность общения с продавцом.

Вторым приложением стоит рассмотреть Music Store. Приложение имеет не самое приятное цветовое решение, так как ядовито красный цвет на белом фоне мешает просмотру приложения.

На главной странице расположено лента скидок сезона, а также каталог инструментов. В навигационном меню имеются кнопки для открытия дополнительных окон экрана, такие как «Shop», «Catalogs», «News», «Profile», «Cart». Внутри информационной части приложение предлагает выполнить вход, записаться в сервис и просмотреть напоминания.

На рисунке 1 представлена раздел «Shop».

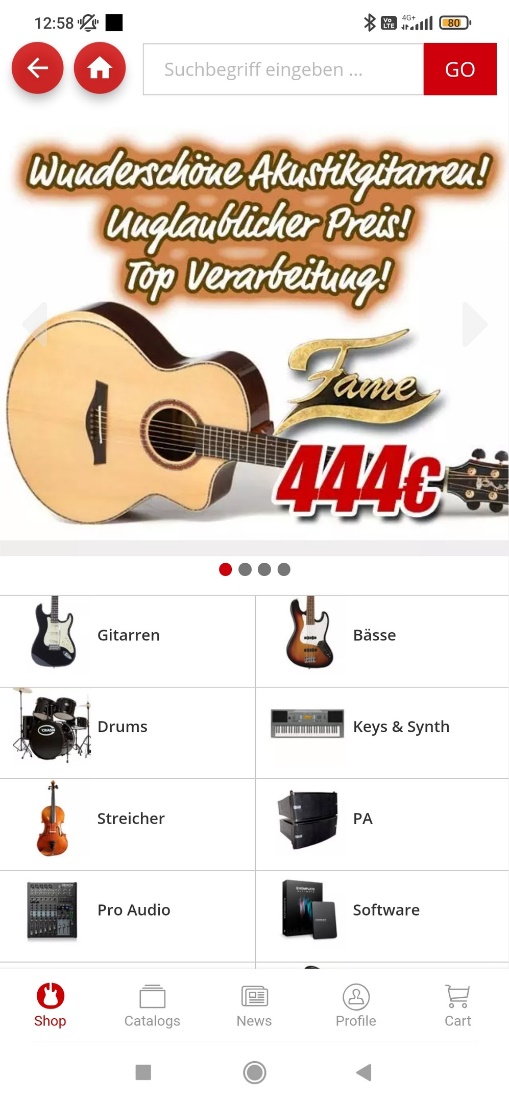


Рисунок 1 – Раздел «Shop»

На данной странице можно выбрать нужный раздел среди представленных для поиска нужных инструментов и их составляющих, а также тут представлена реклама, что будет обновляться каждую вторую неделю. На рисунке 2 представлена раздел «Catalogs».

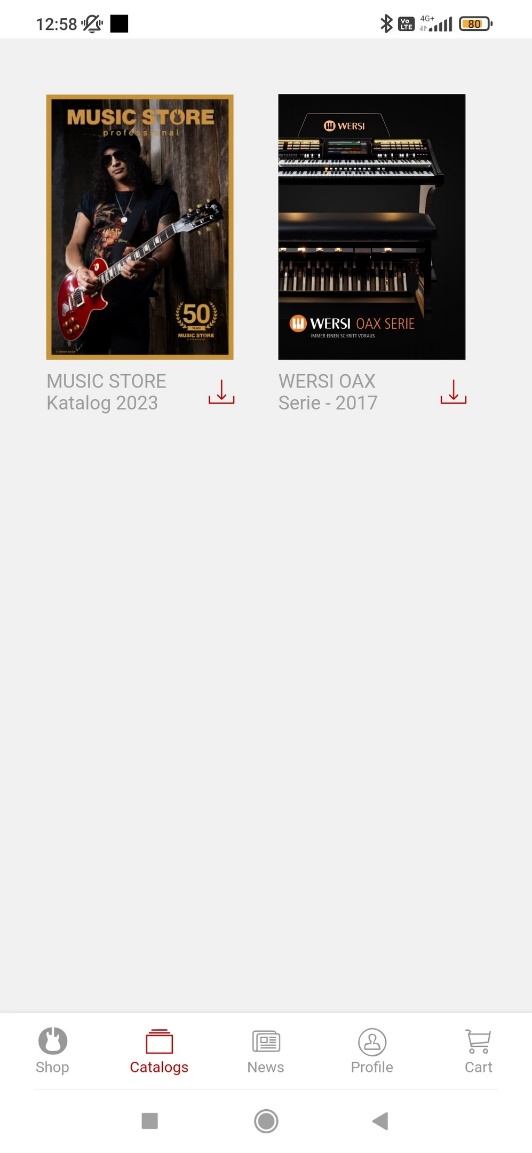


Рисунок 2 – Страница «Catalogs»

Внутри информационной части данной страницы имеется возможность просмотра предложенного каталога музыки, а также можно скачать ее себе на телефон.

На рисунок 3 представлена страница «News»

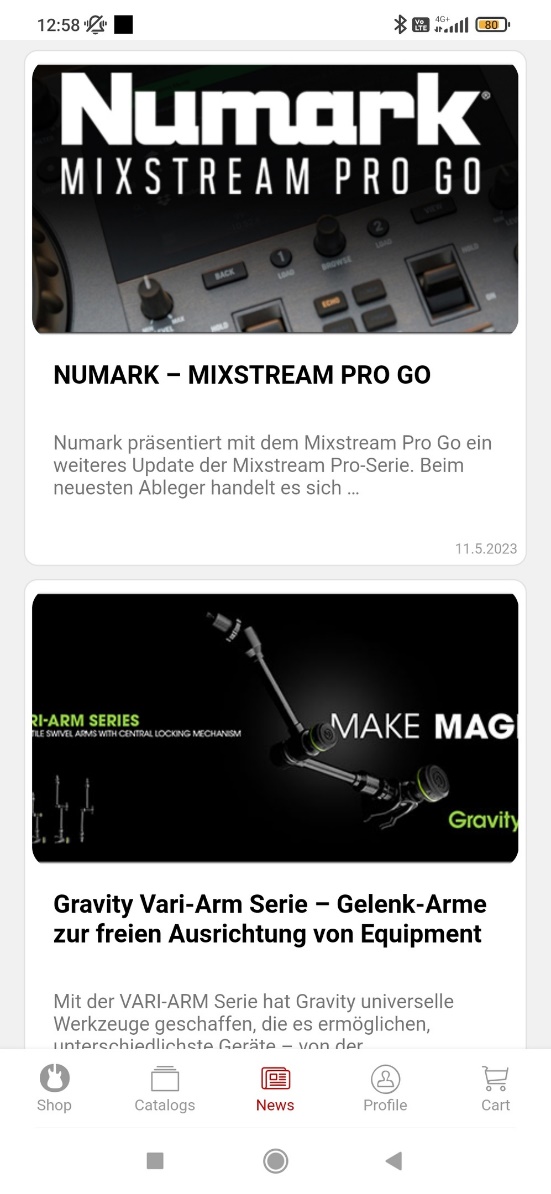


Рисунок 3 – Страница «News»

Данная страница отличается от остальных тем, что она никак не связана с покупкой и продажей музыкальных инструментов, на ней ест лишь информация, связанная с музыкой. На рисунке 4 представлена страница «Profile».

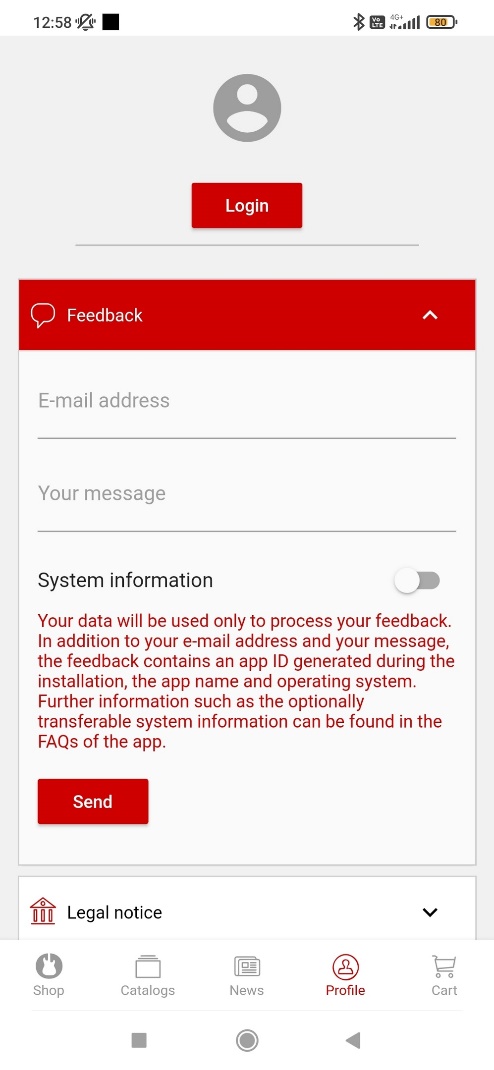


Рисунок 4 – Страница «Profile»

На данной странице имеется возможность зарегистрироваться в приложение и узнать основную информацию о самом приложении. На рисунке 5 представлена страница «Cart».



Рисунок 5 – Страница «Cart»

Данная страница не открывается, если не зарегистрироваться в приложение, что не было сделано при изучении.

Так как других приложений, в которых можно купить музыкальные инструменты, отсутствуют, то в заключении можно сказать, что трендами в разработке являются: возможность регистрации и авторизации, возможность просмотра имеющихся товаров, добавление товаров в корзину.

# Проектирование приложения

## UI/UX дизайн проекта

Дизайн проекта разработан в программе Figma.

Для проекта определены основные экраны:

* регистрация;
* авторизация;
* корзина;
* личный кабинет;
* домашняя страница;
* каталог;
* добавление товара.

Большинство музыкальных инструментов изготавливаются из дерева. По этой причине цветовая гамма состоит из коричневого цвета, с которым приятно сочетается зеленый цвет.

Потому в разработке макета отдано предпочтение спокойным цветам: охра коричневый и зеленый. Но стоит заметить, что данные цвета, разбавленные изображениями, не выглядят скучно, а весьма лаконично.

На рисунке 6 представлена страница главного меню.



Рисунок 6 – Страница «Главное меню»

На данной странице мобильного приложения предоставляется возможность войти в приложение. На рисунке 7 представлена основная страница.

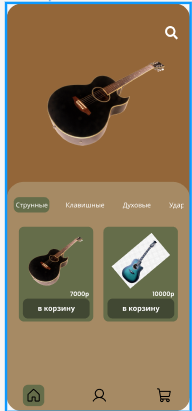


Рисунок 7 – Страница «Основная»

На данной странице можно увидеть реализацию каталога приложения, а также информацию о некоторых инструментах.

На рисунке 8 представлена страница «Профиля» пользователя.

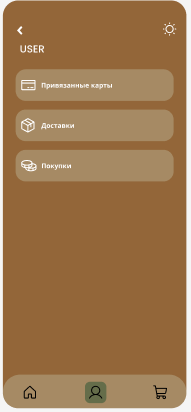


Рисунок 8 – Страница «Профиль»

На данной странице клиент имеет возможность просмотра своих покупок, а также узнать баланс и дату доставки заказа. На рисунке 9 представлена страница «Корзина».



Рисунок 9 – Страница «Корзина»

На данной странице должен располагаться список отложенных товаров, однако эта страница на данный момент пустая, так как пользователь еще не зарегистрировался.

Трендами в разработке являются: возможность регистрации и авторизации, возможность просмотра имеющихся товаров, добавление товаров в корзину.

## Выбор технологии, языка и среды программирования

Средой программирования выбрана программа Android Studio версии. Выбор был сделан, основываясь на том, что в нем редактор кода, с которым удобно работать, а также есть предварительная проверка уже созданного приложения на предмет ошибок в нем.

Языком программирования является Java. Java это много платформенный язык, благодаря которому можно реализовать различные функции в приложении. Безопасность —в большой степени достигается из-за строгости статичной типизации. Этот язык также является многофункциональным и имеет несложный синтаксис.

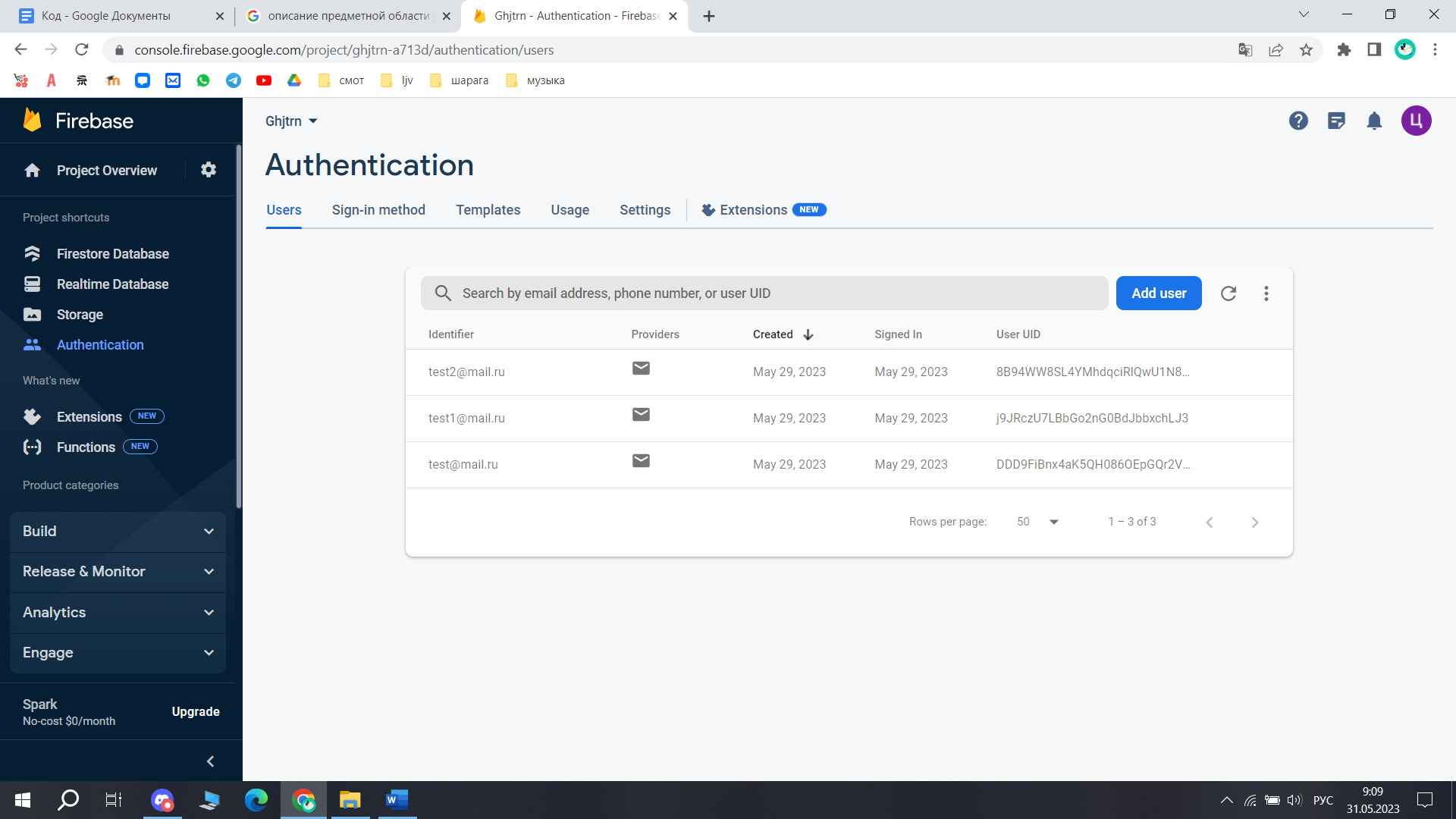
Используемый сервис для базы данных и аутентификации – Firebase. Плюсом которого является ее безопасный и быстрый хостинг. Хостинг от Firebase поддерживает все типы контента, включая веб-приложения, динамический и статический контент.

# Разработка мобильного приложения

## Разработка базы данных

К проекту был подключён сервис Firebase посредством помощника Firebaseв программе Android Studio и прописывания кода в файлах проекта.

С помощью Authentication Firebase реализована аутентификация пользователей на рисунке 10, условием которой является наличие логина и пароля у пользователя.

 Рисунок 10 – Аутентификация пользователей

В ходе анализа выявлены сущности: пользователи, товары.

На рисунке 11 ниже представлена таблица «Пользователи», которая включает в себя email, password, role, username.

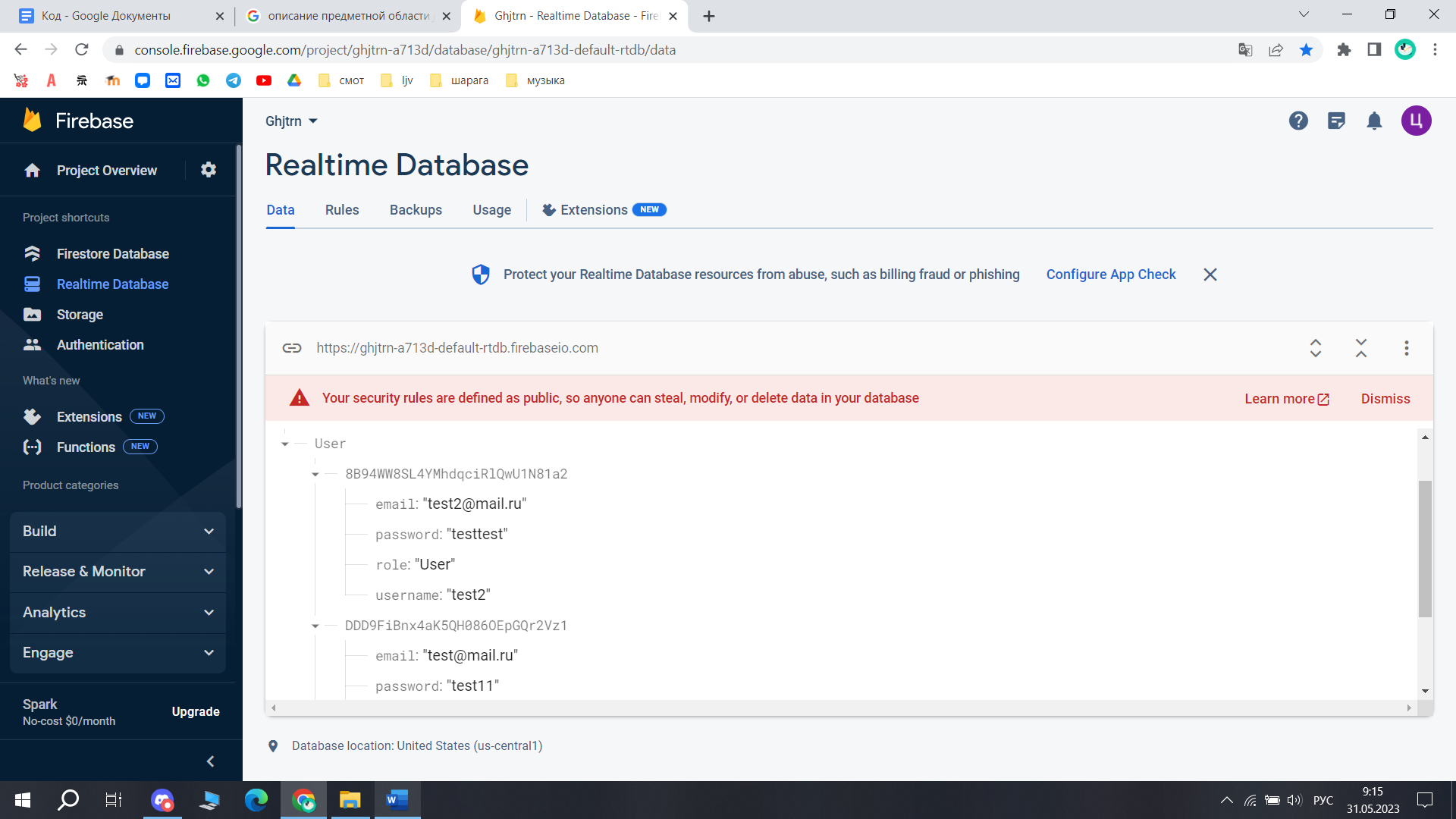


Рисунок 11 – Таблица «Пользователи»

## Разработка используемых плагинов

На рисунке 12 изображено название эмулятора для запуска приложения в AndroidStudio.

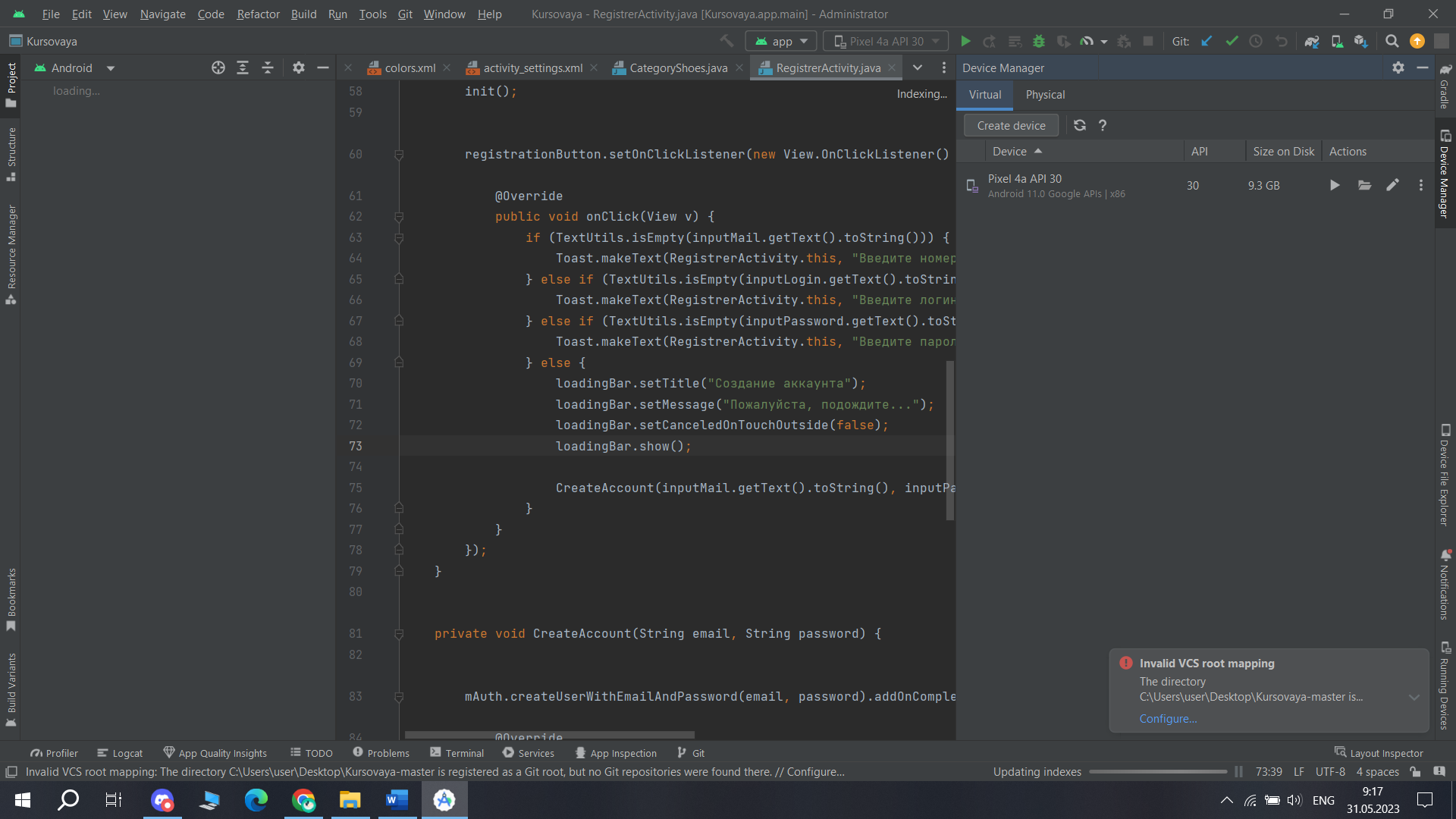


Рисунок 12 – Эмулятор

Так же были использованы, как встраиваемые плагины, так и дополнительные. На рисунке 13 изображены используемых плагинов.

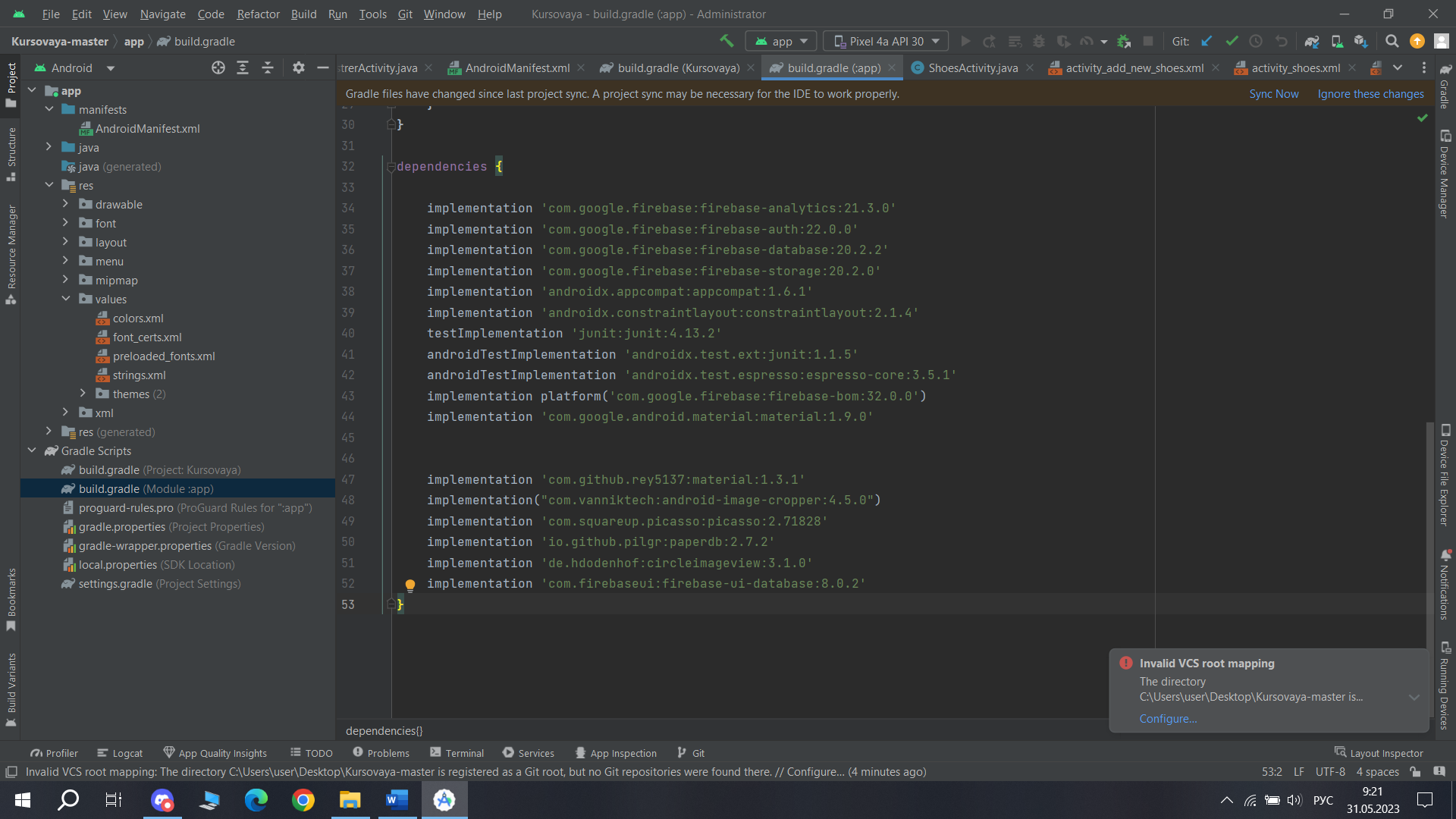


Рисунок 13 –Плагины

Добавленные плагины предназначаются для подключения Firebase, RealTimeDatabase и навигационного меню.

## Описание разработанных процедур и функций

В приложении разработаны следующие функции:

* регистрация и авторизация;
* настройка административных параметров;
* переход между страницами с помощью навигационного меню;
* добавление товара в корзину.

На листинге 1 представлен код, отвечающий за настройки административных параметров в приложении.

Листинг 1 – Настройки административных возможностей

public class AdminSettings extends AppCompatActivity { // Определение публичного класса AdminSettings, который наследуется от класса AppCompatActivity

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState); // Вызов метода onCreate родительского класса AppCompatActivity

setContentView(R.layout.activity\_admin\_settings);

activity\_admin\_settings // Установка макета для текущей активности

}

}

На листинге 2 представлен код, который отвечает за представление текущего авторизованного пользователя и хранение его ключей доступа к данным, таким как электронная почта и пароль.

Листинг 2 – Представление текущего пользователя

public class Prevalent { // Определение публичного класса Prevalent

public static Users currentOnlineUser; // Объявление статической переменной currentOnlineUser типа Users

public static final String UserEmailKey = «UserEmail»; // Объявление статической константы UserEmailKey типа String со значением «UserEmail»

public static final String UserPasswordKey = «UserPassword»; // Объявление статической константы UserPasswordKey типа String со значением «UserPassword»

}

На листинге 3 представлен код, отвечающий за отображение содержимого «Корзины». Это связано с функционалом, связанный с покупками или заказами в приложении.

Листинг 3 – Отображение «Корзины»

public class BagActivity extends AppCompatActivity { // Определение публичного класса BagActivity, который наследуется от класса AppCompatActivity

@Override

protected void onCreate(Bundle savedInstanceState) { // Переопределение метода onCreate с параметром savedInstanceState типа Bundle

super.onCreate(savedInstanceState); // Вызов метода onCreate родительского класса AppCompatActivity

setContentView(R.layout.activity\_bag); // Установка макета activity\_bag для текущей активности

}

}

На листинге 4 представлен код, который необходим для аутентификации пользователя с помощью электронной почты и пароля. После успешного входа происходит проверка роли пользователя и в зависимости от роли выполняется переход на соответствующий профиль (администратора или пользователя).

Листинг 4 – Авторизация пользователя

public class LoginActivity extends AppCompatActivity {

private FirebaseAuth mAuth;

private DatabaseReference mDataBase;

private Button LoginButton;

private EditText InputEmail;

private EditText InputPassword;

private ProgressDialog loadingBar;

private TextView ForgetPassword;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

init();

mAuth = FirebaseAuth.getInstance();

// Обработчик нажатия кнопки "Войти"

LoginButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (TextUtils.isEmpty(InputEmail.getText().toString())) {

// Если поле ввода почты пустое, отображается сообщение о необходимости ввода почты

Toast.makeText(LoginActivity.this, "Введите почту!", Toast.LENGTH\_SHORT).show();

} else if (TextUtils.isEmpty(InputPassword.getText().toString())) {

// Если поле ввода пароля пустое, отображается сообщение о необходимости ввода пароля

Toast.makeText(LoginActivity.this, "Введите пароль!", Toast.LENGTH\_SHORT).show();

} else {

loadingBar.setTitle("Авторизация");

loadingBar.setMessage("Выполняется вход...");

loadingBar.setCanceledOnTouchOutside(false);

loadingBar.show();

loginUser(InputEmail.getText().toString(), InputPassword.getText().toString());

}

}

});

// Обработчик нажатия ссылки "Забыли пароль?"

ForgetPassword.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// Переход к активности сброса пароля

Intent resetPassIntent = new Intent(LoginActivity.this, ForgetPasswordActivity.class);

startActivity(resetPassIntent);

}

});

}

// Метод, вызываемый при старте активности

protected void onStart() {

super.onStart();

FirebaseUser currentUser = mAuth.getCurrentUser();

// Проверка, если текущий пользователь не равен null, то выполнен вход

// В данном коде закомментированы действия, которые могут быть выполнены при наличии или отсутствии пользователя

}

// Метод для выполнения входа пользователя

private void loginUser(String email, String password) {

mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task) {

if (task.isSuccessful()) {

DatabaseReference roleRef;

String userUid = mAuth.getCurrentUser().getUid();

roleRef = FirebaseDatabase.getInstance().getReference("User").child(userUid);

roleRef.addListenerForSingleValueEvent(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot snapshot) {

String roleName = snapshot.child("role").getValue(String.class);

if (roleName != null && roleName.equals("Admin")) {

// Переход к профилю администратора

Intent adminIntent = new Intent(LoginActivity.this, AdminProfile.class);

startActivity(adminIntent);

} else {

loadingBar.dismiss();

Toast.makeText(LoginActivity.this, "Вы успешно вошли!", Toast.LENGTH\_SHORT).show();

// Переход к профилю пользователя

Intent succesLogin = new Intent(LoginActivity.this, Profile.class);

startActivity(succesLogin);

}

}

@Override

public void onCancelled(@NonNull DatabaseError error) {

// Обработка ошибки запроса к базе данных

}

});

} else {

loadingBar.dismiss();

Toast.makeText(LoginActivity.this, "Неправильный логин или пароль!", Toast.LENGTH\_SHORT).show();

}

}

});

}

// Метод инициализации компонентов интерфейса

private void init() {

LoginButton = findViewById(R.id.login\_btn);

InputEmail = findViewById(R.id.login\_email\_input);

InputPassword = findViewById(R.id.login\_password\_input);

loadingBar = new ProgressDialog(this);

mDataBase = FirebaseDatabase.getInstance().getReference();

Paper.init(this);

ForgetPassword = findViewById(R.id.forget\_password\_link);

}

}

На листинге 5 представлен код, который отображает информацию о пользователе, такую как имя, изображение профиля и предоставляет навигационное меню с опциями для просмотра корзины, категорий и других функций.

Листинг 5 – Отображение данных пользователя и навигация по приложению

public class Profile extends AppCompatActivity {

private ImageView logoutBtn;

private ImageView settingsBtn;

private CircleImageView ProfileImage;

private BottomNavigationView NavigationMenu;

private String currentUserId;

private TextView HelloUser;

private Button toFavouriteBtn;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_profile);

// Инициализация элементов пользовательского интерфейса

init();

// Получение ссылки на базу данных Firebase

DatabaseReference UsersRef = FirebaseDatabase.getInstance().getReference().child("User").child(currentUserId);

UsersRef.addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot snapshot) {

// Получение имени пользователя из базы данных и отображение его в приветствии

String username = snapshot.child("username").getValue() != null ? snapshot.child("username").getValue().toString() : "";

HelloUser.setText("Пользователь " + username);

// Получение URL-адреса изображения пользователя из базы данных и отображение его с помощью библиотеки Picasso

String imageUrl = snapshot.child("image").getValue() != null ? snapshot.child("image").getValue().toString() : "";

if (!imageUrl.equals("")) {

Picasso.get().load(imageUrl).into(ProfileImage);

} else {

Picasso.get().load(R.drawable.profile).into(ProfileImage);

}

}

@Override

public void onCancelled(@NonNull DatabaseError error) {

// Обработка ошибки получения данных из базы данных

}

});

// Обработчик нажатия для кнопки "Выйти"

logoutBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// Удаление данных пользователя из локального хранилища

Paper.book().destroy();

// Выход из учетной записи Firebase

FirebaseAuth.getInstance().signOut();

// Переход на главный экран (MainActivity)

Intent logoutIntent = new Intent(Profile.this, MainActivity.class);

startActivity(logoutIntent);

}

});

// Обработчик нажатия для кнопки "Настройки"

settingsBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// Переход на экран настроек (SettingsActivity)

Intent settingsIntent = new Intent(Profile.this, SettingsActivity.class);

startActivity(settingsIntent);

}

});

// Обработчик выбора элементов нижней панели навигации

NavigationMenu.setOnNavigationItemSelectedListener(new BottomNavigationView.OnNavigationItemSelectedListener() {

@Override

public boolean onNavigationItemSelected(@NonNull MenuItem item) {

int ItemId = item.getItemId();

if (ItemId == R.id.navigation\_profile) {

// Действия при выборе элемента "Профиль"

} else if (ItemId == R.id.navigation\_bag) {

// Переход на экран корзины (BagActivity)

startActivity(new Intent(Profile.this, BagActivity.class));

} else if (ItemId == R.id.navigation\_home) {

// Переход на экран категорий (CategoryActivity)

startActivity(new Intent(Profile.this, CategoryActivity.class));

}

return true;

}

});

// toFavouriteBtn.setOnClickListener(new View.OnClickListener() {

// @Override

// public void onClick(View v) {

// Intent toFavouriteIntent = new Intent(Profile.this, FavouriteActivity.class);

// startActivity(toFavouriteIntent);

// }

// });

}

private void init() {

// Инициализация элементов пользовательского интерфейса

logoutBtn = (ImageView) findViewById(R.id.logout\_btn);

settingsBtn = (ImageView) findViewById(R.id.settings\_btn);

ProfileImage = (CircleImageView) findViewById(R.id.user\_profile\_image);

// Получение идентификатора текущего пользователя Firebase

currentUserId = FirebaseAuth.getInstance().getCurrentUser().getUid();

HelloUser = (TextView) findViewById(R.id.username);

NavigationMenu = (BottomNavigationView) findViewById(R.id.navigation\_panel);

// toFavouriteBtn = findViewById(R.id.to\_favourite\_button);

}

}

# Тестирование

## Протокол тестирования дизайна приложения

Для проведения тестирования дизайна приложение было выбрано устройство Honor 10i c разрешение экрана 2340×1080.

Проверка была проведена на:

* оптимальный размер кнопок;
* читабельный размер шрифта;
* приятное глазу цветовое сочетание;
* корректное расположение элементов.

## Протокол тестирования функционала приложения

Для проверки правильности ввода пароля при регистрации были проведены UnitTests, которые изображены ниже на листинге 6.

Листинг 6–UnitTests

public class PasswordUtils {  
  
 private static final String *VALID\_PASSWORD\_REGEX* = "((?=.\*\\d)(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[@#$%=:\\?]).{8,12})";  
  
 @Test  
 public void validatePassword\_EmptyString() {  
*// setup*String password = "";  
  
*// execute*boolean actual = PasswordUtils.*validatePassword*(password);  
  
*// assert  
assertFalse*(actual);  
 }  
  
 @Test  
 public void validatePassword\_Null() {  
*// setup*String password = null;  
  
*// execute*boolean actual = PasswordUtils.*validatePassword*(password);  
  
*// assert  
assertFalse*(actual);  
 }  
  
 @Test  
 public void validatePassword\_Missing\_OneNumber() {  
*// setup*String password = "Abcdefg#";  
  
*// execute*boolean actual = PasswordUtils.*validatePassword*(password);  
  
*// assert  
assertFalse*(actual);  
 }  
  
 @Test  
 public void validatePassword\_LengthTooLong() {  
*// setup*String password = "Abcdefg5#abcdefgabcd";  
  
*// execute*boolean actual = PasswordUtils.*validatePassword*(password);  
  
*// assert  
assertFalse*(actual);  
 }  
  
 @Test  
 public void validatePassword\_Missing\_OneUpperCaseLetter() {  
*// setup*String password = "abcdefg5#";  
  
*// execute*boolean actual = PasswordUtils.*validatePassword*(password);  
  
*// assert  
assertFalse*(actual);  
 }  
  
  
  
 public static booleanvalidatePassword(String password) {  
 return (password != null &&password.matches(*VALID\_PASSWORD\_REGEX*));  
 }  
}

Так же для тестирования функционала были разработаны TestCase.

Test-Case №1

Название тест кейса: Регистрация

Предусловия тест кейса: приложение открыто

Шаги тест кейса:

* 1. шаг тест кейс №1: открыть страницу регистрации;
  2. шаг тест кейса №2: заполнить все поля;
  3. шаг тест кейса №3: нажать на кнопку «регистрация».

Ожидаемый результат тест кейса:

Пользователь зарегистрирован в системе.

Статус кейса: выполнено.

Test-Case №2

Название тест кейса: Авторизация

Предусловия тест кейса: пользователь зарегистрирован

Шаги тест кейса:

* 1. шаг тест кейс №1: открыть страницу авторизации;
  2. шаг тест кейса №2: заполнить все поля;
  3. шаг тест кейса №3: нажать на кнопку «Вход».

Ожидаемый результат тест кейса:

Пользователь войдёт в систему.

Статус кейса: выполнено.

Test-Case №3

Название тест кейса: Переход на другую активность

Предусловия тест кейса: пользователь вошёл в систему

Шаги тест кейса:

1. шаг тест кейс №1: открыть страницу с навигацией;
2. шаг тест кейса №2: нажать на желаемую кнопку.

Ожидаемый результат тест кейса:

Пользователь перешёл на необходимую ему страницу.

Статус кейса: выполнено.

Test-Case №4

Название тест кейса: Добавление товара в корзину

Предусловия тест кейса: пользователь вошёл в систему

Шаги тест кейса:

1. шаг тест кейс №1: открыть страницу каталога;
2. шаг тест кейс №2: выбрать нужные товары;
3. шаг тест кейс №3: нажать на кнопку «Добавить».

Ожидаемый результат тест кейса:

Пользователь отложил в корзину товар, что хочет приобрести.

Статус кейса: выполнено.

Test-Case №5

Название тест кейса: Настройка административных параметров

Предусловия тест кейса: администратор вошёл в систему

Шаги тест кейса:

1. шаг тест кейс №1: открыть товар;
2. шаг тест кейс №2: изменить информацию о нем;
3. шаг тест кейс №3: сохранить изменения.

Ожидаемый результат тест кейса:

Администратор изменил информацию о товаре.

Статус кейса: выполнено.

Заключение

В ходе разработки мобильного приложения для магазина музыкальных инструментов учтены предпочтения и потребности пользователей, в формате комфортного магазина музыкальных инструментов.

Интерфейс приложения разработан с учетом удобства использования и минимального количества действий для достижения пользовательской цели. Основные экраны спроектированы таким образом, чтобы не перегружать пользователей лишней информацией, и добавлены полезные функции, такие как просмотр товаров и возможность добавления его в список избранных товаров.

Важным элементом концепции приложения является возможность отложить понравившийся товар в корзину для дальнейшей покупки, которая позволяет пользователям сократить время на поиск и покупку музыкальных инструментов.

Важным этапом в разработке приложения было тестирование дизайна, которое подтвердило его корректность и успешное функционирование. Это гарантирует, что разработанное приложение соответствует поставленным целям и требованиям.

В целом разработка мобильной версии магазина музыкальных инструментов позволила создать функциональное и удобное приложение, учитывающее потребности пользователей. Проделанная работа успешно достигла поставленной цели курсового проекта и является основой для дальнейшего развития и улучшения приложения.

Библиография

1. AndroidDevelopers [электронный ресурс]: Документация AndroidStusio – https://developer.android.com/docs/
2. HelpCenter [электронныйресурс]: ДокументацияJava: https://docs.oracle.com/en/java/
3. Firebase [электронныйресурс]: Документация Firebase - https://firebase.google.com/docs?hl=uk
4. Material Design [электронныйресурс]: BottomNavigation: https://material.io/components/bottom-navigation/android
5. Git[электронныйресурс]: ДокументацияGit: https://git-scm.com/
6. Figma[электронныйресурс]: Прототипирование проектовhttps://www.figma.com/

Приложение А

Техническое задание

Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледж

имени Б.С. Галущака»

**разработка мобильного приложения для автосервиса**

НАТКиГ.202100.010.000ПЗ

Выполнил:

студент группы

ПР-20.106

Петрищева А.В.

2022

**Содержание**

Введение 34

1Назначения разработки 35

2Требованияк мобильному приложению 36

2.1 Требования к функциональным характеристикам 36

2.2 Требования к надёжности 36

2.3 Условия эксплуатации 37

2.4 Требования к составу и параметрам технических средств 2337

2.5 Требования к информационной и программной совместимости 2337

2.6 Требования к защите информации 38

2.7 Требования к маркировке и упаковке 2338

3Требования к программной документации 2338

4Технико-экономические показатели 2338

5Стадии и этапы разработки 23638

6Порядок контроля и приёмки 2339

**Введение**

Настоящее техническое задание распространяется на разработку информационной системы «Разработкамобильногоприложениядляавтосервиса», используемого для записи на приём, просмотра услуг, чата с оператором.

Наименование приложения: «Garage54».

Краткая характеристика области применения: приложение для автосервиса – полезное оснащение телефона автовладельца. При необходимости пользователь может просмотреть имеющиеся в автосервисе услуги, записаться на приём и обратиться за помощью к оператору.

Условные обозначения и сокращения:

БД – База данных;

ИС – Информационная система;

ОС – Операционная система;

СТО – Станция технического обслуживания.

Основанием для проведения разработки является Протокол №6 от 21 февраля 2022 года.

Наименование темы разработки – «Разработка мобильного приложения для автосервиса».

Условное обозначение темы разработки – «Автосервис – Garage54».

**1Назначение разработки**

Основное назначение мобильного приложения заключается в:

* осведомлении клиентов об услугах;
* возможности клиента записи на приём.

Лица, которые могут работать с данной Системой:

пользователь–просматривает услуги, регистрируется, авторизуется, оформляет запись;

администратор–добавляет и редактирует услуги, просматривает записи.

**2Требования к мобильному приложению**

2.1 Требования к функциональным характеристикам

Требования к составу выполняемых функций:

* регистрация и авторизация пользователей;
* просмотр имеющихся услуг;
* запись на приём;
* чат с оператором.

Входные данные должны быть организованы в виде вводимого, в специальную форму, текста, соответствующего определённому шаблону. Данные, вводимые вручную, проверяются на корректность.

**2.2 Требования к надёжности**

Обеспечение устойчивого функционирования должно выполняться несколькими действиями:

* организация бесперебойного питания оборудования пользователя;
* использование лицензионного программного обеспечения;
* организация стабильного интернет-соединения.

Приложение должно контролировать входную информацию:

* соблюдение типов данных при заполнении полей;
* операции изменения, удаления и сохранения.

Время восстановления после отказа, вызванного сбоем электропитания технических средств, не фатальным сбоем ОС или файловой системы, не должно превышать 15 минут при соблюдении условий эксплуатации технических и программных средств и правильной настройки операционной системы.

**2.3 Условия эксплуатации**

Обслуживание ИС включает в себя:

1. информационное обслуживание – ввод и редактирование информации БД;
2. системное администрирование БД ИС.

**2.4 Требования к составу и параметрам технических средств**

Для работы приложения на устройстве требуется: объем свободной памяти не менее 3 ГБ, версия устройства Android 10 и выше.

**2.5 Требования к информационной и программной совместимости**

Проектирование взаимодействия с файловой системой должно быть выполнено в рамках разработки курсового проекта. При разработке взаимодействия с файловой системой должен быть использован язык программирования Java.

**2.6 Требования к защите информации**

Доступ к информации БД предоставляется только администратору базы данных.

**2.7** **Требования к маркировке и упаковке**

Требования к маркировке и упаковке не предъявляются.

**3Требования к программной документации**

Состав программной документации должен включать в себя:

* техническое задание;
* проектную документацию.

**4Технико-экономические показатели**

Экономические преимущества разработки и ориентировочная экономическая эффективность не рассчитывается.

**5Стадии и этапы разработки**

Таблица 1 – Стадии разработки

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название этапа | Срок,даты | Отчётность |
| 1 | Исследование предметной области | 25.03.2022-28.03.2022 | Пояснительная записка |
| 2 | Выбор моделей, описывающих предметную область | 01.04.2022-03.04.2022 | Пояснительная записка |
| 3 | Разработка технического задания | 03.04.2022-05.04.2022 | Техническое задание |
| 4 | Изучение Java и Firebase | 01.04.2022-  18.04.2022 | Пояснительная записка |
| 5 | Анализ требований и уточнение спецификаций | 14.04.2022 | Спецификации программного обеспечения |
| 6 | Проектирование структуры мобильного приложения, проектирование компонентов (технический проект) | 15.04.2022-29.04.2022 | Схема структурная системы и спецификации компонентов |
| 8 | Кодирование клиентской части | 01.05.2022-20.06.2022 | Программный  Продукт |
| 9 | Тестирование компонентов.  Сборка и комплексное тестирование | 20.06.2022 | Тексты программных компонентов |
| 10 | Разработка программной  документации | 24.06.2022 – 26.06.2022 | Программная  документация |
| 12 | Защита | 27.06.2022 |  |

**6Порядок контроля и приёмки**

Виды испытаний – защита курсового проекта.

Общее требования к приёмке:

* техническое задание;
* пояснительная записка;
* программный продукт;
* презентация.