Министерство образования Новосибирской области

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

**разработка** **мобильного приложения Социальной сети**

Пояснительная записка к курсовому проекту

ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

МДК.01.03 Разработка мобильных приложений

НАТКиГ.200100.010.000ПЗ

Разработал:

студент группы ПР-20.106

Новиков М.А.

2022

**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

Мобильное приложение для СТО делает взаимодействие с предприятием более удобным для клиента: в случае возникновения проблемы с автомобилем у пользователя не должно возникнуть трудности, куда обратиться. Клиент может записаться на обслуживание с помощью приложения, выбрав нужные услуги, дату и время. …

Вопрос, касающийся нахождения автосервиса, предоставляющего качественные услуги, является очень актуальным, ведь как машин, так и автосервисов с каждым днём становится всё больше, что затрудняет поиск необходимого.

Целью курсового проекта является создание мобильного приложения социальной сети «Bigeny».

Задачами курсового проекта в связи с указанной целью являются:

* изучение предметной области темы;
* рассмотрение приложения с точки зрения пользователя для выявления необходимых функций приложения;
* изучения процесса создания своего собственного API для социальной сети;

Объект исследования – приложения социальной сети.

Предмет исследования – изучение принципов функционирования и инструментов приложения.

# Исследовательский раздел

## Описание предметной области

Количество пользователей сети интернет растет с каждым годом, в связи с этим требуются приложения для общения. За последние 20 лет на рынке появилось множество социальных сетей, которые борется за трафик пользователей. Основным методом конкуренции в таком случае является либо предоставление чего-то нового, либо обеспечение удобства использования.

Основные функции социальных сетей обмен информацией, как массовый, (публикации в каналах, группах), так и личный (чаты, беседы и т.д.).

Из этого можно определить, что социальная сеть позволит упростить коммуникацию между людьми.

## Образ клиента

Клиентами являются обычные пользователи сети интернет.

Приложение будет рассчитано как на мужскую, так и на женскую аудиторию.

Как правило, в новые социальные сети приходит молодая аудитория, поэтому ожидается, что средний возраст пользователя будет 20-30 лет.

## Сценарии

Прежде всего, при составлении сценария выявляется цель, которой будут достигать пользователи. В первую очередь целью является простота и удобство. Например:

Молодой человек договорился о встречи с девушкой и подходя, он решил уточнить место встречи. В данном случае важна скорость и удобство, с которой он это сделает. Приложение должно быстро открыться и предоставить удобною навигацию по чату, чтобы молодой человек смог быстро предать информацию.

Можно взять противоположную ситуацию и представить, что человек не очень социальный, но все еще имеет потребность в общение. В таком случае социальная сеть должна предоставить ему удобный интерфейс, чтобы опубликовать пост или оставить под ним комментарий или прочитать комментарии.

Так же можно представить ситуацию, когда группа людей работают совместно над каким-то проектом. В таком случае приложение предоставляет удобную систему обсуждения и оперативного оповещения участников проекта.

## Сбор и анализ прототипов

У данного мобильного приложения много конкурентов. Из самых известных:

* Telegram;
* ВКонтакте;
* Одноклассники;
* Twitter;
* Facebook;
* Instagram;
* Pinterest;
* Tumbler;

Рассмотрим дизайн, ошибки и достижения приложения Telegram.

Telegram имеет настраиваемый дизайн, а так же темную и светлую тему.

Ниже приведено 2 рисунка, темной и светлой темы соответственно.

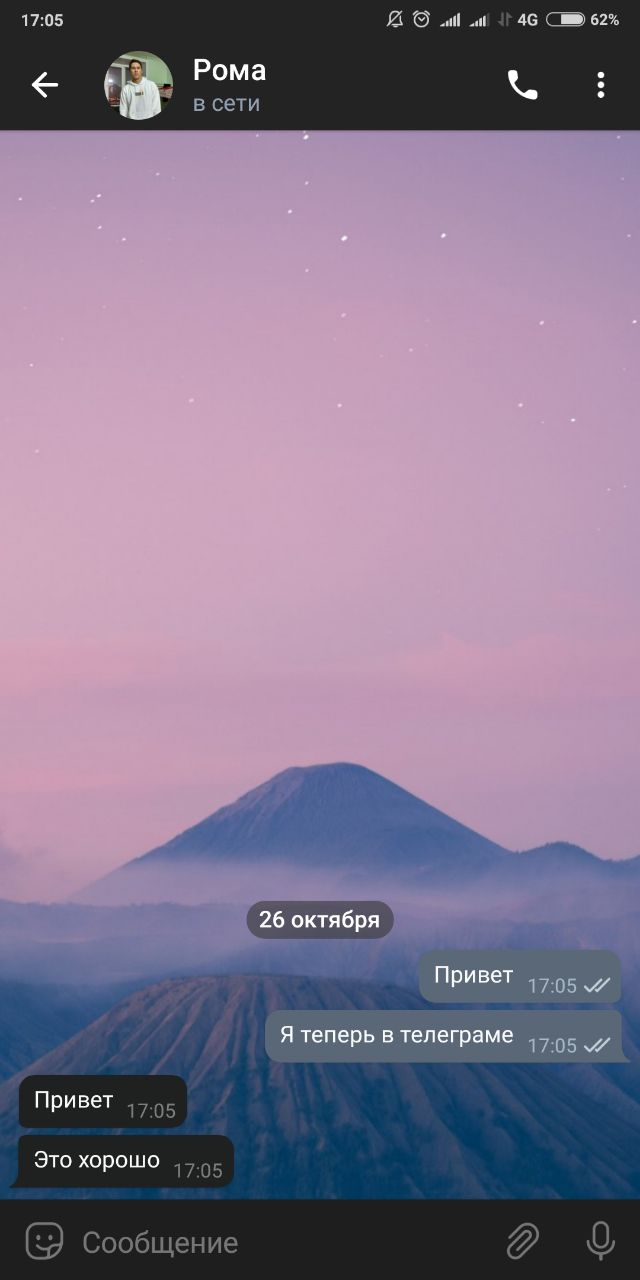


Рисунок 1 — Темная тема

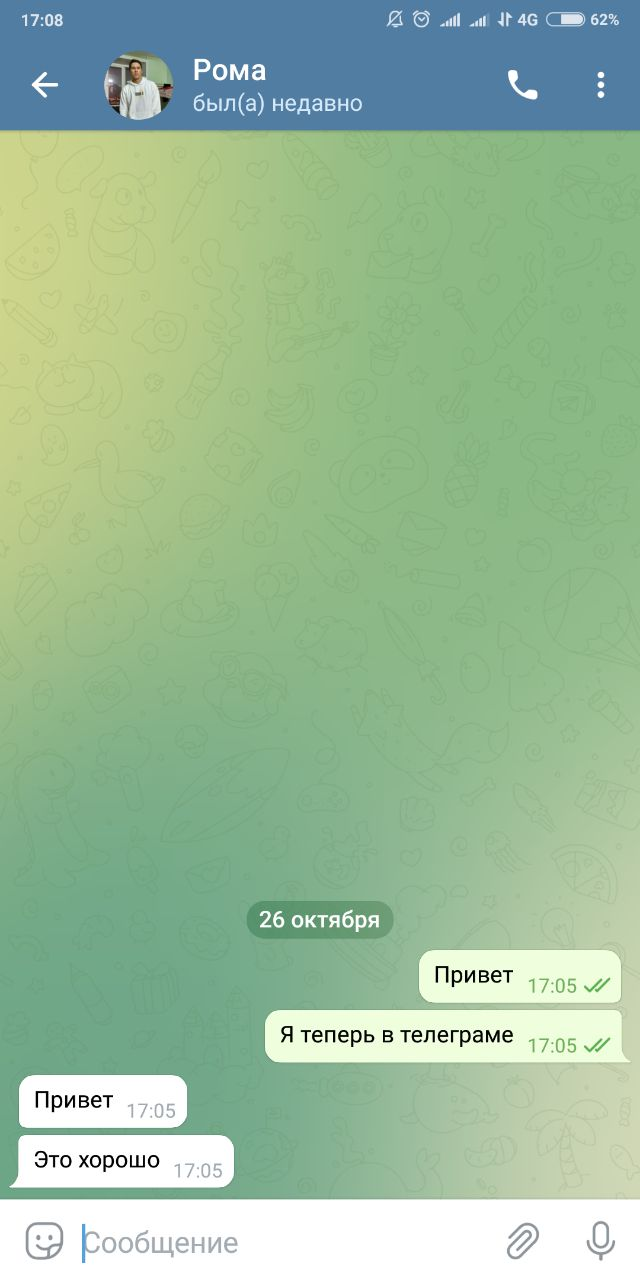


Рисунок 2 — Светлая тема

У приложения есть шапка

Исходя из рассмотрения данного приложения можно сделать вывод:

Плюсы приложения:

* оснащенность главными функциями;
* цветовую гамму можно настраивать.

Минусы:

* отсутствие главной ленты и системы рекомендации каналов;

В заключении трендами в разработке являются: возможность регистрации и авторизации, возможность создание каналов, чатов и бесед, удобная возможность оставлять и просматривать комментарии к записям, а так же единая система оценивания записей.

# Проектирование приложения

## UI/UX дизайн проекта

Дизайн проекта был разработан в программе Figma.

Для проекта были определены основные экраны:

* экран регистрации;
* экран авторизации;
* экран записи в автосервис;
* экран личного кабинета;
* экран чата с оператором;
* экран отзывов клиентов.

Цветовая гамма была выбрана посредством изучения информации о цветовосприятии цветов, как мужчин, так и женщин. Мужчины предпочитают монохромность или сочетание цветов, расположенных в спектре рядом. А еще для них привлекательны черный и оттенки серого. Женщины более серьезно относятся к подбору цветов и их сочетаемости и более чувствительны к явлению цветовой гармонии, выбирая сочетания дополнительных цветов, неприемлемые для мужчин. Например, оранжевого и фиолетового или желтого и синего.

Потому было принято решение в разработке макета отдать предпочтение спокойным цветам: синий, черный, белый. Но стоит заметить, что данные цвета, разбавленные изображениями, не выглядят скучно, а весьма лаконично.

Ниже на рисунке (Рисунок 6) представлен логотип автосервиса.



Рисунок 6 – Логотип автосервиса

На логотипе изображена тематическая приложению иллюстрация.

На рисунке (Рисунок 7) ниже представлена страница регистрации.

Рисунок 7 – Страница «Регистрации»

На данной странице мобильного приложения предоставлена возможность зарегистрироваться новому пользователю, а в случае существования аккаунта у пользователя, он может перейти на страницу входа.

На рисунке (Рисунок 8) ниже представлена страница входа в приложение.

Рисунок 8 – Страница «Входа»

На данной странице имеется возможность входа в приложение, если пользователь ещё не зарегистрирован, он может перейти к странице регистрации.

На рисунке (Рисунок 9) ниже представлена страница записи в автосервис.

Рисунок 9 – Страница «Услуги»

На данной странице клиент имеет возможность записаться в автосервис. Для этого ему достаточно заполнить поле с VIN автомобиля, выбрать необходимую услугу, выбрать свободную дату и записаться.

На рисунке (Рисунок 10) ниже представлена страница чата с оператором.



Рисунок 10 – Страница «Чата»

На данной странице имеется возможность общения с оператором по интересующим вопросам на соответствующие автосервису темы.

На рисунке (Рисунок 11) ниже представлена страница отзывов.

Рисунок 11 – Страница «Отзывы»

На данной странице клиент имеет возможность просмотреть оставленные другими клиентами отзывы.

На рисунке (Рисунок 12) ниже представлена страница личного кабинета.

Рисунок 12 – Страница «Личный кабинет»

На данной странице представлена возможность поиска услуг, перехода на другие страницы приложения.

## Выбор технологии, языка и среды программирования

Средой программирования была выбрана программа Android Studio версии. Языком программирования является Java.

Используемый сервис для базы данных и аутентификации – Firebase.

# Разработка мобильного приложения

## Разработка базы данных

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

С помощью Authentication Firebase была реализована аутентификация пользователей (Рисунок 13), условием которой является наличие пароля и электронной почты у пользователя. Так же при заполнении полей регистрации происходит проверка заполнения: валидация пароля и электронной почты.

Рисунок 13 – Аутентификация пользователей

В ходе анализа были выявлены сущности: запись на приём, чат, пользователи, услуги.

На рисунке (Рисунок 14) ниже представлена таблица «Пользователи», которая включает в себя контакт пользователя и VIN его автомобиля.



Рисунок 14 – Таблица «Пользователи»

На рисунке (Рисунок 15) ниже представлена таблица «Запись на приём», которая включает в себя название услуги, дату записи и VIN обслуживаемого автомобиля.

Рисунок 15 – Таблица «Запись на приём»

## Разработка используемых плагинов

Ниже приведен листинг (Листинг 1), на котором изображено название эмулятора для запуска приложения в Android Studio.



Листинг 1 – Эмулятор

Так же были использованы, как встраиваемые плагины, так и дополнительные. Ниже приведен листинг, (Листинг 2) используемых плагинов.



Листинг 2 – Плагины

Добавленные плагины предназначаются для подключения Firebase, RealTime Database, Auth и навигационного меню.

## Описание разработанных процедур и функций

В приложении разработаны следующие функции:

* регистрация и авторизация;
* переход между страницами с помощью навигационного меню;
* запись на приём;
* просмотр услуг автосервиса;
* появление календаря по нажатию на после с выбором даты;
* чат.

Ниже представлен листинг (Листинг 4) кода, который отвечает за регистрацию пользователя в системе.

public class MainActivity extends AppCompatActivity {  
  
 EditText inputemail, inputpassword, cconformpassword;  
 Button registerBtn, vxodBtn;  
 String emailPattern = "[a-zA-z0-9.\_-]+@[a-z]+\\.+[a-z]+";  
 ProgressDialog progressDialog;  
  
 FirebaseAuth mAuth;  
 FirebaseUser mUser;  
  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 vxodBtn = findViewById(R.id.*vxodBtn*);  
 inputemail = findViewById(R.id. *inputemail*);  
 inputpassword = findViewById(R.id. *inputpassword*);  
 cconformpassword = findViewById(R.id.*cconformpassword*);  
 registerBtn = findViewById(R.id. *registerBtn*);  
 progressDialog = new ProgressDialog(this);  
  
 mAuth = FirebaseAuth.*getInstance*();  
 mUser = mAuth.getCurrentUser();  
  
 vxodBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 startActivity(new Intent(MainActivity.this, login.class));  
 }  
 });  
  
  
 registerBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 PerforAuth();  
 }  
 });  
 }  
  
  
  
 private void PerforAuth() {  
 String email = inputemail.getText().toString();  
 String password = inputpassword.getText().toString();  
 String conformpassword = cconformpassword.getText().toString();  
  
 if(password.isEmpty() || password.length()<6)  
 {  
 inputpassword.setError("Введите пароль, не менее 6 символов");  
 } else if(!password.equals(conformpassword))  
 {  
 cconformpassword.setError("Пароли не совпадают");  
 }  
 else{  
 progressDialog.setMessage("Пожалуйста, подождите");  
 progressDialog.setTitle("Регистрация");  
 progressDialog.setCanceledOnTouchOutside(false);  
 progressDialog.show();  
  
  
  
 mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 if (task.isSuccessful())  
 {  
 progressDialog.dismiss();  
 sendUserToNextActivity();  
 Toast.*makeText*(MainActivity.this, "Регистрация прошла успешно", Toast.*LENGTH\_LONG*).show();  
  
 }  
 else {  
 progressDialog.dismiss();  
 Toast.*makeText*(MainActivity.this, ""+task.getException(), Toast.*LENGTH\_LONG*).show();  
  
 }  
 }  
 });  
 }  
 }  
  
 private void sendUserToNextActivity() {  
 Intent intent = new Intent(MainActivity.this, Home.class);  
 intent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK*| Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(intent);  
 }  
  
  
}

Листинг 4 – Регистрация пользователя

Во время регистрации происходит проверка пароля пользователя на соответствие: пароль состоит не менее из 6 символов и совпадает с повторным ведением пароля. А также, если пользователь с данным адресом электронной почты уже зарегистрирован, пользователь получит предупреждение об этом.

Ниже на листинге (Листинг 5) кода представлен метод перехода на другую активность.

vxodBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 startActivity(new Intent(MainActivity.this, login.class));  
 }  
 });  
  
  
 registerBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 PerforAuth();  
 }  
 });  
}

Листинг 5 – Переход на другие активности

Ниже на листинге (Листинг 6) представлен код, отвечающий за авторизацию пользователя в системе.

public class login extends AppCompatActivity {  
  
 EditText inputemail, inputpassword;  
 Button loginBtn;  
 Button createNewAccount;  
 String emailPattern = "[a-zA-z0-9.\_-]+@[a-z]+\\.+[a-z]+";  
 ProgressDialog progressDialog;  
  
 FirebaseAuth mAuth;  
 FirebaseUser mUser;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
  
 inputemail = findViewById(R.id.*inputemail*);  
 inputpassword = findViewById(R.id.*inputpassword*);  
 loginBtn = findViewById(R.id.*loginBtn*);  
 createNewAccount = findViewById(R.id.*createNewAccount*);  
  
 progressDialog = new ProgressDialog(this);  
  
 mAuth = FirebaseAuth.*getInstance*();  
 mUser = mAuth.getCurrentUser();  
  
  
 createNewAccount.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 startActivity(new Intent(login.this, MainActivity.class));  
 }  
 });  
  
 loginBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 perforLogin();  
 }  
 });  
  
  
 }  
  
 private void perforLogin() {  
  
 String email = inputemail.getText().toString();  
 String password = inputpassword.getText().toString();  
  
 if (password.isEmpty() || password.length() < 6) {  
 inputpassword.setError("Введите пароль, не менее 6 символов");  
 } else {  
 progressDialog.setMessage("Пожалуйста, подождите");  
 progressDialog.setTitle("Вход");  
 progressDialog.setCanceledOnTouchOutside(false);  
 progressDialog.show();  
  
 mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 if (task.isSuccessful()) {  
 progressDialog.dismiss();  
 Toast.*makeText*(login.this, "Вход выполнен", Toast.*LENGTH\_LONG*).show();  
 }else  
 {  
 progressDialog.dismiss();  
 Toast.*makeText*(login.this, ""+task.getException(), Toast.*LENGTH\_LONG*).show();  
  
 }  
 }  
 });  
 }  
  
 }  
  
 private void sendUserToNextActivity() {  
 Intent intent = new Intent(login.this, Home.class);  
 intent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK*| Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(intent);  
 }  
  
}

Листинг 6 – Авторизация пользователя

Авторизация была выполнена с помощью сервиса Firebase Auth.

# 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 (Листинг 3):

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 boolean validatePassword(String password) {  
 return (password != null && password.matches(*VALID\_PASSWORD\_REGEX*));  
 }  
}

Листинг 3 – UnitTests

Так же для тестирования функционала были разработаны TestCase.

Test-Case №1

Название тест кейса: Регистрация

Предусловия тест кейса: приложение открыто02

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.

Заключение

В результате работы было разработано клиентское мобильное приложение для автосервиса.

Во время разработки возникли трудности с подключением сервиса Firebase к проекту, проблема была решена изменением Android на AndroidX. AndroidX – это переработанная библиотека, чтобы сделать имена пакетов более понятными.

В дальнейшем планируется совершенствовать приложение и добавить уже к имеющемуся функционал.

Библиография

1. Android Developers [электронный ресурс]: Документация Android Stusio – https://developer.android.com/docs/
2. Help Center [электронный ресурс]: Документация Java: https://docs.oracle.com/en/java/
3. Автостат [электронный ресурс]: Структура российских автовладельцев по возрасту и маркам https://www.autostat.ru/infographics/40710/
4. Firebase [электронный ресурс]: Документация Firebase - https://firebase.google.com/docs?hl=uk
5. Material Design [электронный ресурс]: Bottom Navigation: https://material.io/components/bottom-navigation/android
6. Git [электронный ресурс]: Документация Git: https://git-scm.com/
7. Figma [электронный ресурс]: Прототипирование проектов https://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

1 Назначения разработки 35

2 Требования к 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

3 Требования к программной документации 2338

4 Технико-экономические показатели 2338

5 Стадии и этапы разработки 23638

6 Порядок контроля и приёмки 2339

**Введение**

Настоящее техническое задание распространяется на разработку информационной системы «Разработка мобильного приложения для автосервиса», 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».

**1** **Назначение 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.

**2 Требования 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.

**3 Требования к программной документации**

Состав программной документации должен включать в себя:

* техническое задание;
* проектную документацию.

**4 Технико-экономические показатели**

Экономические преимущества разработки и ориентировочная экономическая эффективность не рассчитывается.

**5 Стадии 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 |  |

**6 Порядок контроля и приёмки**

Виды испытаний – защита курсового проекта.

Общее требования к приёмке:

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