Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледжимени Б.С. Галущака»

**РАЗРАБОТКА Мобильной версии видеохостинга**

Пояснительная записка к курсовому проекту

ПМ.01Разработка модулей программного обеспечения для компьютерныхсистем

МДК.01.03 Разработка мобильных приложений

НАТКиГ.202000.010.000ПЗ

Разработал:

студент группы ПР-21.106

Шестаков И.В.

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

Большую часть жизни современных людей занимает медиапространство, выступающих, как пространство для поиска медиафайлов. Мобильное приложение видеохостинга облегчает просмотр и загрузку видео для пользователя. В любой момент пользователь может загрузить видеои посмотреть его. Понравившиеся видеоролики пользователь может добавить в закладки, и в будущем, найти их в соответствующей вкладке.

Вопрос, касающийся нахождения видеохостинга, подходящего для потребностей пользователя, очень актуален. Ведь каждый видеохостинг предлагает свою реализацию задачи, и не может удовлетворить каждого пользователя.

Целью курсового проекта является создание мобильного приложения видеохостинга.

Задачами курсового проекта в связи с указанной целью являются:

* изучение предметной области мобильного приложения видеохостинга;
* рассмотрение приложения с точки зрения пользователя для выявления функций приложения;
* разработать и реализовать дизайн приложения;
* написать код приложения;
* протестировать полученный продукт.

Объект исследования – мобильное приложение видеохостинга.

Предмет исследования –изучение принципов функционирования и инструментов приложения.

# Исследовательский раздел

## Описание предметной области

Видеохостинг очень важная и нужная сфера развлечений. Не каждый может позволить хранить большой объем видеороликов на своих устройствах, к тому же, делится такими видео не очень удобно. К тому же устройство может сломаться, что приведет к потере данных. Именно поэтому видеохостинги очень важны и востребованы. А так как потребности пользователей бесчисленны, то и видеохостингов существует большое количество. Из-за этого возникает важный вопрос, какой видеохостинг выбрать.

Видеохостинги стали неотъемлемой частью интернета для большинства людей, ведь они являются удобным инструментом для хранения личных видеороликов и просмотра интересующего вас контента. К тому же, они позволяют сохранять просмотренные и понравившиеся пользователю видеоролики в отдельные списки.

Таким образом, можно с уверенностью сказать, что видеохостинги очень облегчают жизнь пользователей, давая неограниченное место под загрузку своих видеороликов, и позволяя хранить их, без угрозы их потери.

Приложение «VideoHost» – программный продукт, который всегда будет под рукой, так как телефон у современного человека всегда с собой. Приложение является библиотекой видеороликов пользователей с необходимостью авторизации.

## Образ клиента

Клиентами являются зарегистрированные пользователи видеохостинга. Приложение предназначено для людей различных возрастов, но основным контингентом будет молодёжь от 12 до 30 лет. Приложение удобно как для людей, которые часто делятся видеороликами с другими, так и для тех, кто желает удобно просматривать видеоролики.

## Сценарии

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

Молодой человек ведет социальную активность в интернете, и ему необходимо быстро загружать видеоролики, чтобы поделитьсяими. До этого он загружал видеоролики в мессенджер и отправлял их по отдельности каждому своему знакомому. Но ему будет гораздо легче делиться видеороликами, если на его устройстве будет приложение видеохостинга, в которое можно быстро их загружать и делиться со всеми знакомыми одной ссылкой.

В другом случае, молодой человек, наоборот, любит посмотреть видеоролики, загруженные в сеть другими пользователями и сохранять их. Но для этого приходится сидеть на нескольких сайтах, чтобы следить за всем, что интересует его. А для того, чтобы сохранять их, приходится скачивать их на свой ё устройство. Но куда удобнее иметь мобильное приложение, которое позволит собрать видеоролики по интересующим темам, и сохранять их в разделе «Избранное».

## Сбор и анализ прототипов

В Google Play и AppStore можно найти 2 популярных приложения, выполняющих роль видеохостинга. Первое – YouTube (Амереканское приложение), второе – RUTUBE (Русское приложение). Обы этих приложения имеют возможность просматривать видеоролики, сохранять понравившиеся и загружать собственные видеоролики с возможность поделиться ими. Рассмотрим эти два приложения подробнее и срравним их.

Сравнение по основным критериям данных приложений представлено в таблице 1.

Таблица 1 – Сравнение приложений

|  |  |  |
| --- | --- | --- |
| Параметры | YouTube | RUTUBE |
| Стоимость | Бесплатно | Бесплатно |
| Основной язык | Английский | Русский |
| Просмотр видеороликов | Есть | Есть |
| Возможность сохранять видеоролики в «Избранное» | Есть | Есть |
| Подробное описание к контенту | Есть | Есть |
| Наличие поиска | Есть | Есть |
| Выбор темы интерфейса | Есть | Есть |
| Список просмотренного | Есть | Есть |
| Оценки видеороликов | Есть | Есть |
| Возможность загружать свои видеоролики | Есть | Есть |
| Возможность продавать свой контент | Есть | Отсутвует |

Рассмотрев несколько приложений, выполняющих похожие задачи, было решено написать приложение, которое имело бы простой интерфейс без лишних функций и с акцентом внимания на важных для пользователя элементах, в котором можно смотреть видеоролики и добавлять их в «Избранное».

## UI/UX дизайн проекта

Дизайн проекта разработан в программе Figma.

Для проекта были определены основные экраны:

* экран регистрации;
* экран авторизации;
* главный экран с видеороликами;
* экран личного кабинета;
* экран с понравившимися видеороликами;
* экран загрузки нового видеоролика;
* экран загруженных видеороликов;
* экран просмотра видеоролика.

Цветовая гамма была выбрана путём изучения трендов. В наше время популярны тёмные темы приложений, так как они довольно приятны для глаз. Это подтверждается тем, что большинство приложений и сайтов сейчас имеют тёмную тему.

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

Ниже на рисунке (Рисунок 4) представлен логотип видеохостинга.



Рисунок 4 – Логотип видеохостинга

На рисунке (Рисунок 5) ниже представлена страница регистрации.

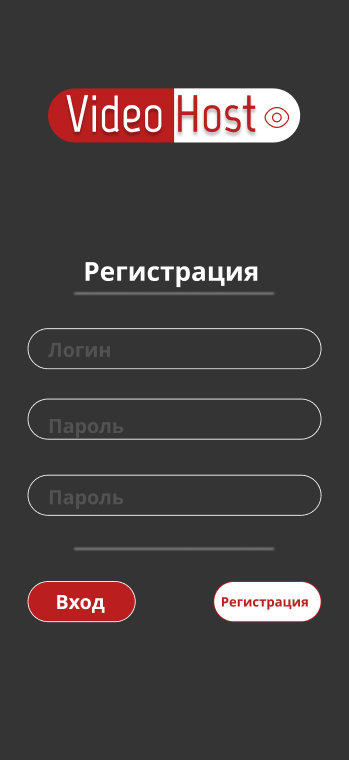


Рисунок 5 – Страница «Регистрации»

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 6) ниже представлена страница входа в приложение.

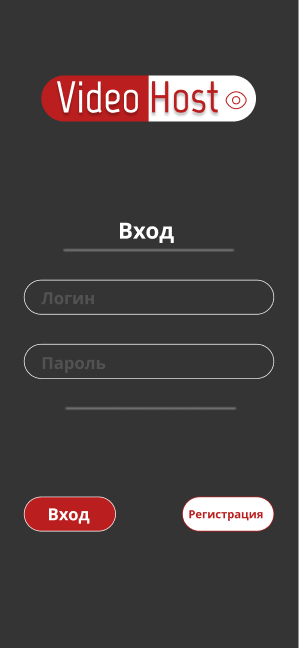


Рисунок 6 – Страница «Входа»

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

На рисунке (Рисунок 7) ниже представлена главная страница с видеороликами.



Рисунок 7 – Главная страница

На данной странице пользователь будет видеть все видеоролики, загруженные на хостинг. Данную страницу можно листать до тех пор, пока видеоролики не закончатся.

На рисунке (Рисунок 8) ниже представлена страница личного кабинета.

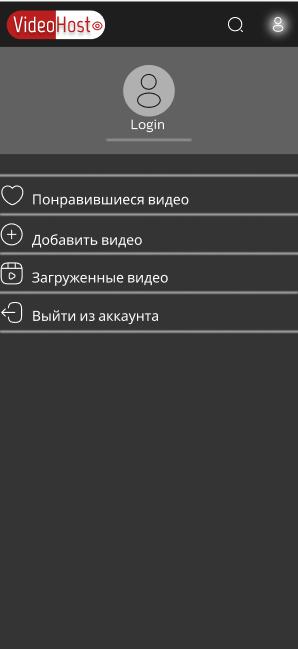


Рисунок 8 – Страница «Личный кабинет»

На данной странице имеется возможность выйти из аккаунта, или перейти на другие экраны.

На рисунке (Рисунок 9) ниже представлена страница понравившихся видеороликов.



Рисунок 9 – Страница «Понравившиеся»

На данной странице пользователь может посмотреть понравившиеся ему видеоролики.

На рисунке (Рисунок 10) ниже представлена страница добавления нового видеоролика.

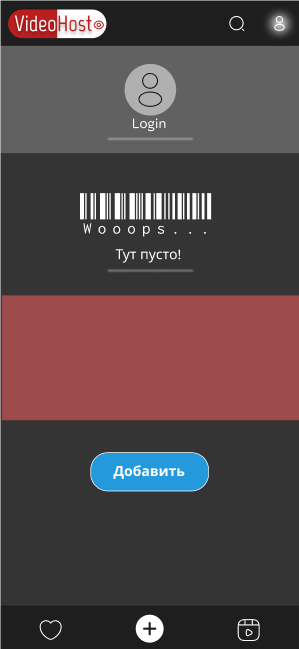


Рисунок 10 – Страница добавления нового видеоролика

На данной странице пользователь может добавить новый видеоролик на свой аккаунт.

На рисунке (Рисунок 11) ниже представлена страница загруженных пользователем видеороликов.

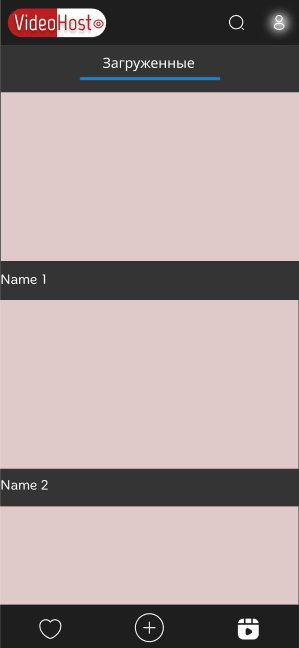


Рисунок 11 – Страница загруженных видеороликов

На данной странице пользователь может посмотреть загруженные им видеоролики.

На рисунке (Рисунок 12) ниже представлена страница для просмотра видеороликов.



Рисунок 12 – Страница загруженных видеороликов

На данной странице пользователь может посмотреть видеоролик.

## Выбор технологии, языка и среды программирования

Средой программирования выбрана программа AndroidStudio, так как она довольно удобна для создания приложения и является довольно популярной, поэтому в случае возникновения ошибки, легче найти способ её решения. Языком программирования является Java, так как это старый и развитый язык программирования, в котором уже реализовано множество важных функция, которые могут пригодиться при разработке приложения.

Используемый сервис для базы данных и аутентификации – Firebase. Это бесплатная база данных, которая подходит под поставленную мною задачу. Она довольно быстрая и легка для понимания.

# Разработка мобильного приложения

## Разработка базы данных

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

С помощью AuthenticationFirebaseбыла 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 VINего автомобиля.



Рисунок 14 – Таблица «Пользователи»

На рисунке (Рисунок 15) ниже представлена таблица «Запись на приём», которая включает в себя название услуги, дату записи и VINобслуживаемого 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 идополнительные. Ниже приведен листинг, (Листинг 2) используемых плагинов.



Листинг 2 –Плагины

Добавленные плагины предназначаются для подключения Firebase, RealTimeDatabase, 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 {  
  
EditTextinputemail, inputpassword, cconformpassword;  
 Button registerBtn, vxodBtn;  
 String emailPattern = "[a-zA-z0-9.\_-]+@[a-z]+\\.+[a-z]+";  
ProgressDialogprogressDialog;  
  
FirebaseAuthmAuth;  
FirebaseUsermUser;  
  
  
  
 @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("Введитепароль, неменее 6 символов");  
 } else if(!password.equals(conformpassword))  
 {  
cconformpassword.setError("Паролинесовпадают");  
 }  
 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, "Регистрацияпрошлауспешно", 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 {  
  
EditTextinputemail, inputpassword;  
 Button loginBtn;  
 Button createNewAccount;  
 String emailPattern = "[a-zA-z0-9.\_-]+@[a-z]+\\.+[a-z]+";  
ProgressDialogprogressDialog;  
  
FirebaseAuthmAuth;  
FirebaseUsermUser;  
  
 @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("Введитепароль, неменее 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, "Входвыполнен", 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 FirebaseAuth.

# Тестирование

## Протокол тестирования дизайна приложения

Для проведения тестирования дизайна приложение было выбрано устройство Honor 10ic 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 booleanvalidatePassword(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:открытьстраницу 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 результатеработыбыло 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. AndroidDevelopers [электронный ресурс]: ДокументацияAndroidStusio – https://developer.android.com/docs/
2. HelpCenter [электронныйресурс]: ДокументацияJava: https://docs.oracle.com/en/java/
3. Автостат[электронный ресурс]:Структура российских автовладельцев по возрасту и маркамhttps://www.autostat.ru/infographics/40710/
4. Firebase [электронныйресурс]: Документация Firebase - https://firebase.google.com/docs?hl=uk
5. Material Design [электронныйресурс]: BottomNavigation: https://material.io/components/bottom-navigation/android
6. Git[электронныйресурс]: ДокументацияGit: https://git-scm.com/
7. 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.