

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

ассистент

должность, уч. степень, звание

подпись, дата

Н. И. Чулочникова

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 8

СИСТЕМА АУТЕНТИФИКАЦИИ. FIREBASE AUTHENTICATION

по курсу:

КРОССПЛАТФОРМЕННОЕ ПРОГРАММИРОВАНИЕ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №

4326

подпись, дата

Г. С. Томчук

инициалы, фамилия

Санкт-Петербург 2025

1 Цель работы

Цель работы: создание мобильного приложения на Kotlin для ОС Android с использованием платформы Firebase для реализации системы аутентификации и создания пользовательской базы данных.

2 Задание

Вариант работы: № 17 (мотоциклисты). Создайте систему аутентификации для приложения, используя Firebase Authentication. Реализуйте возможность регистрации новых пользователей, входа в систему и управления учетной записью. Вариант брать из ЛР6.

3 Листинг программы

Листинг программного кода приложения представлен в Приложении А.

4 Результаты работы программы

На рисунках 1–5 изображены результаты тестирования программы.

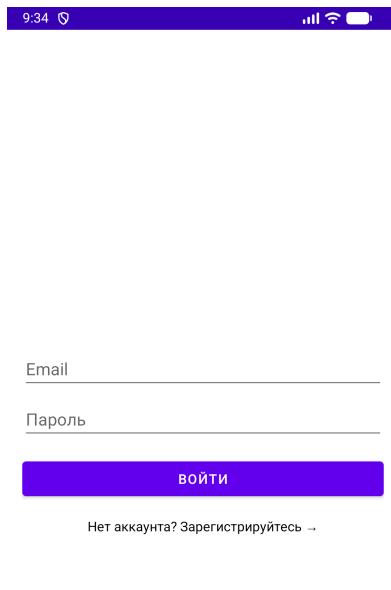


Рисунок 1 — Экран входа в аккаунт



Рисунок 2 — Экран редактирования данных учетной записи

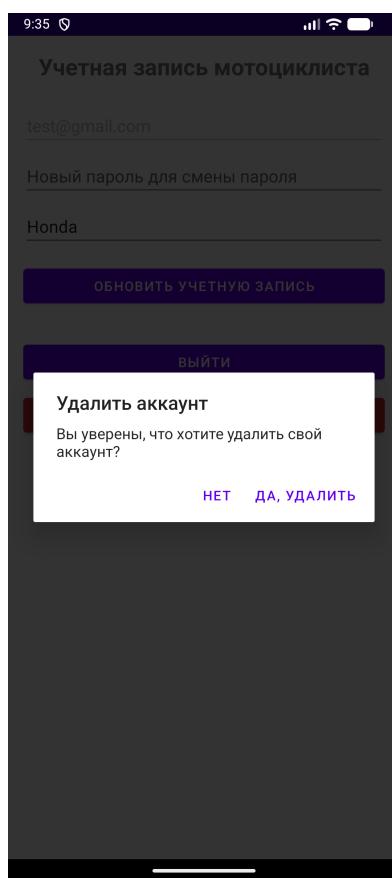


Рисунок 3 — Диалоговое окно подтверждения удаления аккаунта



Email _____

Пароль _____

Модель мотоцикла _____

СОЗДАТЬ АККАУНТ

Уже есть аккаунт? Войдите →



Рисунок 4 — Экран регистрации



test@yandex.ru _____

*** _____

Kawasaki R1000 _____

СОЗДАТЬ АККАУНТ

Уже есть аккаунт? Войдите →

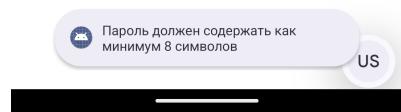


Рисунок 5 — Ошибка регистрации. Ввод пароля недостаточной длины

5 Выводы

В ходе выполнения лабораторной работы была разработана система аутентификации для Android-приложения в соответствии с темой «Мотоциклисты». В качестве основы была использована платформа Firebase, что позволило реализовать современный и безопасный механизм работы с учетными записями пользователей без необходимости создания собственного серверного решения. В процессе разработки была освоена работа с сервисом Firebase Authentication, с помощью которого реализованы регистрация новых пользователей и вход в приложение по адресу электронной почты и паролю. Механизмы аутентификации обеспечивают проверку корректности вводимых данных, а также обработку возможных ошибок, возникающих при регистрации и авторизации.

Для хранения пользовательских данных, не входящих напрямую в учетную запись Firebase, была использована база данных Firestore. В ней сохраняется дополнительная информация о пользователе, в частности модель мотоцикла. Был реализован механизм синхронизации данных между аутентификацией и базой данных.

Также в рамках работы реализованы функции управления учетной записью пользователя. Приложение предоставляет возможность изменения пароля, обновления информации о модели мотоцикла и полного удаления аккаунта, включая связанные данные в базе Firestore. Это позволило закрепить навыки работы с жизненным циклом пользовательских данных и взаимодействием с облачными сервисами.

В результате выполнения лабораторной работы были получены практические навыки интеграции Firebase Authentication и Firestore в Android-приложение, а также опыт проектирования системы аутентификации и управления учетными записями. Разработанное приложение соответствует поставленным требованиям и демонстрирует применение облачных технологий в мобильной разработке.

ПРИЛОЖЕНИЕ А

Листинг 1 — MainActivity.kt

```
package com.grigorijtomczuk.authuser

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.navigation.findNavController
import androidx.navigation.ui.setupActionBarWithNavController
import com.grigorijtomczuk.authuser.databinding.ActivityMainBinding

class MainActivity : AppCompatActivity() {

    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }

    override fun onSupportNavigateUp(): Boolean {
        val navController = findNavController(R.id.nav_host_fragment)
        return navController.navigateUp() || super.onSupportNavigateUp()
    }
}
```

Листинг 2 — LoginFragment.kt

```
package com.grigorijtomczuk.authuser

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Toast
import androidx.fragment.app.Fragment
import androidx.navigation.fragment.findNavController
import com.google.firebase.auth.FirebaseAuth
import com.grigorijtomczuk.authuser.databinding.FragmentLoginBinding

class LoginFragment : Fragment() {

    private var _binding: FragmentLoginBinding? = null
    private val binding get() = _binding!!
    private lateinit var auth: FirebaseAuth

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        _binding = FragmentLoginBinding.inflate(inflater, container, false)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        auth = FirebaseAuth.getInstance()

        if (auth.currentUser != null) {

```

```

        findNavController().navigate(R.id.action_loginFragment_to_profileFragment)
    }

    binding.btnLogin.setOnClickListener {
        val email = binding.etEmail.text.toString()
        val password = binding.etPassword.text.toString()

        if (email.isNotEmpty() && password.isNotEmpty()) {
            auth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener { task ->
                    if (task.isSuccessful) {

findNavController().navigate(R.id.action_loginFragment_to_profileFragment)
                } else {
                    Toast.makeText(
                        context,
                        "Ошибка авторизации: ${task.exception?.message}",
                        Toast.LENGTH_SHORT
                    ).show()
                }
            }
        } else {
            Toast.makeText(context, "Заполните все поля",
                Toast.LENGTH_SHORT).show()
        }
    }

    binding.tvGoToRegister.setOnClickListener {
        findNavController().navigate(R.id.action_loginFragment_to_registerFragment)
    }
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null
}
}

```

Листинг 3 — ProfileFragment.kt

```

package com.grigorijtomczuk.authuser

import android.app.AlertDialog
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Toast
import androidx.fragment.app.Fragment
import androidx.navigation.fragment.findNavController
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.firestore.FirebaseFirestore
import com.grigorijtomczuk.authuser.databinding.FragmentProfileBinding

class ProfileFragment : Fragment() {

    private var _binding: FragmentProfileBinding? = null
    private val binding get() = _binding!!
    private lateinit var auth: FirebaseAuth
    private lateinit var db: FirebaseFirestore

```

```

override fun onCreateView(
    inflater: LayoutInflater, container: ViewGroup?,
    savedInstanceState: Bundle?
): View {
    _binding = FragmentProfileBinding.inflate(inflater, container, false)
    return binding.root
}

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    auth = FirebaseAuth.getInstance()
    db = FirebaseFirestore.getInstance()

    loadUserData()

    binding.btnExit.setOnClickListener {
        updateUserData()
    }

    binding.btnExitLogout.setOnClickListener {
        auth.signOut()
    }

    findNavController().navigate(R.id.action_profileFragment_to_loginFragment)
}

binding.btnExitDeleteAccount.setOnClickListener {
    showDeleteAccountConfirmation()
}
}

private fun loadUserData() {
    val user = auth.currentUser
    if (user != null) {
        binding.etEmail.setText(user.email)
        db.collection("users").document(user.uid).get()
            .addOnSuccessListener { document ->
                if (document != null) {

binding.etMotorcycleModel.setText(document.getString("motorcycleModel"))
                }
            }
            .addOnFailureListener {
                Toast.makeText(
                    context,
                    "Ошибка загрузки данных пользователя",
                    Toast.LENGTH_SHORT
                ).show()
            }
    }
}

private fun updateUserData() {
    val user = auth.currentUser
    binding.etEmail.text.toString()
    val newPassword = binding.etPassword.text.toString()
    val newMotorcycleModel = binding.etMotorcycleModel.text.toString()

    if (user != null) {
        // Обновление Email
        if (newEmail.isNotEmpty() && newEmail != user.email) {
            user.updateEmail(newEmail).addOnCompleteListener { task ->

```

```

        //                     if (task.isSuccessful) {
        //                         Toast.makeText(context, "Email обновлен",
        Toast.LENGTH_SHORT).show()
        //                     } else {
        //                         Log.d("TAG", task.exception?.message.toString())
        //                         Toast.makeText(
        //                             context,
        //                             "Ошибка обновления Email: ${task.exception?.message}",
        //                             Toast.LENGTH_SHORT
        //                         ).show()
        //                     }
        //                 }
        //             }
        //         }

        // Обновление пароля
        if (newPassword.isNotEmpty()) {
            user.updatePassword(newPassword).addOnCompleteListener { task ->
                if (task.isSuccessful) {
                    Toast.makeText(context, "Пароль обновлен",
                    Toast.LENGTH_SHORT).show()
                } else {
                    Toast.makeText(
                        context,
                        "Ошибка обновления пароля: ${task.exception?.message}",
                        Toast.LENGTH_SHORT
                    ).show()
                }
            }
        }

        // Обновление модели мотоцикла
        val userData = hashMapOf<String, Any>(
            "motorcycleModel" to newMotorcycleModel
        )
        db.collection("users").document(user.uid).update(userData)
            .addOnSuccessListener {
                Toast.makeText(context, "Данные обновлены",
                Toast.LENGTH_SHORT).show()
            }
            .addOnFailureListener {
                Toast.makeText(context, "Ошибка обновления данных",
                Toast.LENGTH_SHORT).show()
            }
        }

private fun showDeleteAccountConfirmation() {
    AlertDialog.Builder(context)
        .setTitle("Удалить аккаунт")
        .setMessage("Вы уверены, что хотите удалить свой аккаунт?")
        .setPositiveButton("Да, удалить") { _, _ ->
            deleteAccount()
        }
        .setNegativeButton("Нет", null)
        .show()
}

private fun deleteAccount() {
    val user = auth.currentUser
    user?.let { u ->
        db.collection("users").document(u.uid).delete()
            .addOnSuccessListener {

```

```
        u.delete().addOnCompleteListener { task ->
            if (task.isSuccessful) {
                Toast.makeText(context, "Аккаунт удален",
Toast.LENGTH_SHORT).show()

            findNavController().navigate(R.id.action_profileFragment_to_loginFragment)
            } else {
                Toast.makeText(
                    context,
                    "Ошибка удаления аккаунта:
${task.exception?.message}",
                    Toast.LENGTH_SHORT
                ).show()
            }
        }
    }
    .addOnFailureListener {
        Toast.makeText(
            context,
            "Ошибка удаления данных пользователя",
            Toast.LENGTH_SHORT
        ).show()
    }
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null
}
```

Листинг 4 — RegisterFragment.kt

```
package com.grigorijtomczuk.authuser

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Toast
import androidx.fragment.app.Fragment
import androidx.navigation.fragment.findNavController
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.auth.FirebaseAuthWeakPasswordException
import com.google.firebase.firestore.FirebaseFirestore
import com.grigorijtomczuk.authuser.databinding.FragmentRegisterBinding

class RegisterFragment : Fragment() {

    private var _binding: FragmentRegisterBinding? = null
    private val binding get() = _binding!!
    private lateinit var auth: FirebaseAuth
    private lateinit var db: FirebaseFirestore

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        _binding = FragmentRegisterBinding.inflate(inflater, container, false)
        return binding.root
    }
}
```

```

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    auth = FirebaseAuth.getInstance()
    db = FirebaseFirestore.getInstance()

    binding.btnRegister.setOnClickListener {
        val email = binding.etEmail.text.toString()
        val password = binding.etPassword.text.toString()
        val motorcycleModel = binding.etMotorcycleModel.text.toString()

        if (email.isNotEmpty() && password.isNotEmpty() &&
motorcycleModel.isNotEmpty()) {
            auth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener { task ->
                    if (task.isSuccessful) {
                        val user = auth.currentUser
                        val userData = hashMapOf(
                            "motorcycleModel" to motorcycleModel
                        )
                        user?.let {
                            db.collection("users").document(it.uid).set(userData)
                                .addOnSuccessListener {
                                    Toast.makeText(
                                        context,
                                        "Успешная регистрация",
                                        Toast.LENGTH_SHORT
                                    ).show()
                                }
                        }
                    } else {
                        var errorMessage = ""
                        when (task.exception) {
                            is FirebaseAuthWeakPasswordException -> errorMessage =
                                "Пароль должен содержать как минимум 8 символов"
                            else -> errorMessage =
                                "Ошибка регистрации: ${task.exception?.message}"
                        }
                        Toast.makeText(
                            context,
                            errorMessage,
                            Toast.LENGTH_SHORT
                        ).show()
                    }
                }
            } else {
                Toast.makeText(context, "Заполните все поля",
                Toast.LENGTH_SHORT).show()
            }
        }
    }
}

```

```
        binding.tvGoToLogin.setOnClickListener {
            findNavController().popBackStack()
        }

    }

    override fun onDestroyView() {
        super.onDestroyView()
        _binding = null
    }
}
```