МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт кибербезопасности и цифровых технологий

Кафедра КБ-14 «Цифровые технологии обработки данных»

ОТЧЕТ
по практической работе №6

Выполнил Бурмистров И.Г.
фамилия, имя, отчество

шифр 22Б0616 группа БСБО-07-22

Проверил _____ Изергин Д.А.
ученая степень, должность *фамилия, имя, отчество*

Москва 2025г.

В ходе данной работы были созданы модули «app», «employeedb», «internalfilestorage», «notebook» и «securesharedpreferences» (см. Рисунок 1).

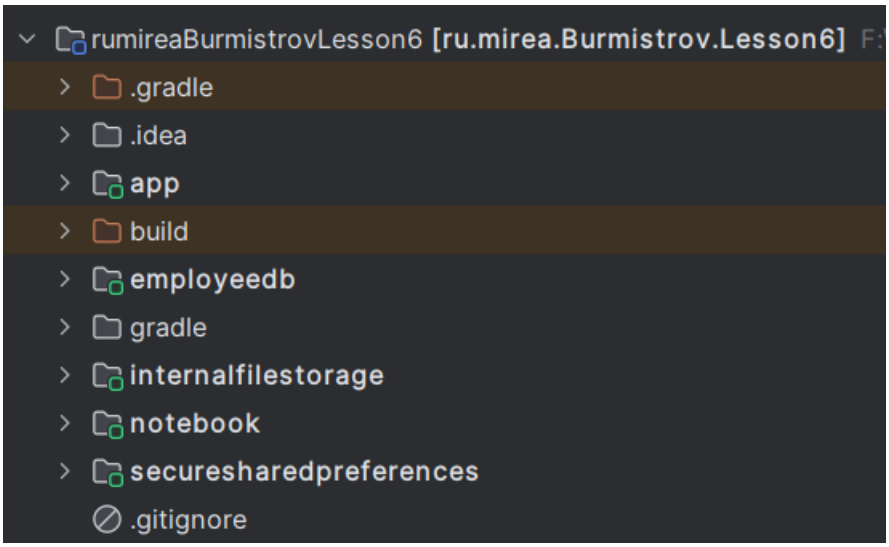

Рисунок 1. Модули проекта

В первом модуле «app» было создано 3 поля для ввода. При нажатии на кнопку «Отправить» данные сохраняются на устройстве (см. Рисунки 2-3 и Листинг 1).

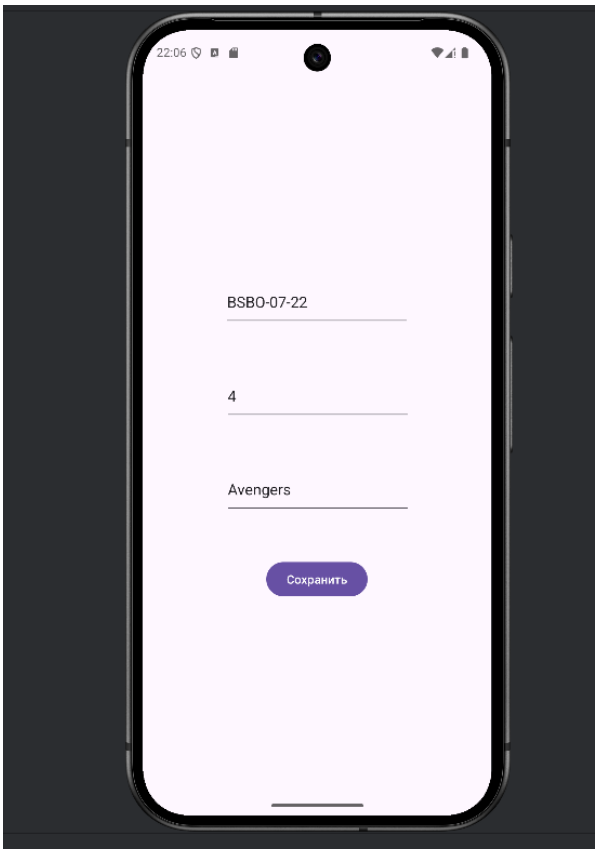

Рисунок 2. Окно приложения

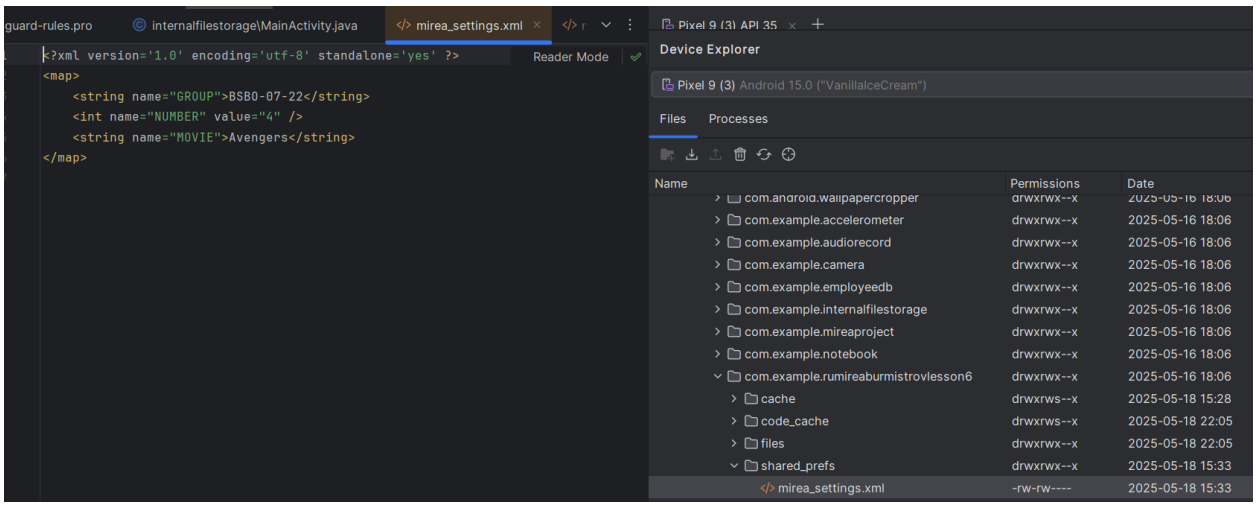

Рисунок 3. Сохранённые данные

```
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;
    private EditText editTextGroup, editTextList, editMovie;
    private SharedPreferences sharedPreferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        sharedPreferences = getSharedPreferences("mirea_settings",
MODE_PRIVATE);

        editTextGroup = binding.editTextText;
        editTextList = binding.editTextText2;
        editMovie = binding.editTextText3;

        Button buttonSave = binding.button;

        editTextGroup.setText(sharedPreferences.getString("GROUP", ""));

editTextList.setText(String.valueOf(sharedPreferences.getInt("NUMBER", 0)));
        editMovie.setText(sharedPreferences.getString("MOVIE", ""));

        buttonSave.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v){
                String group = editTextGroup.getText().toString();
                int number =
Integer.parseInt(editTextList.getText().toString());
                String movie = editMovie.getText().toString();

                SharedPreferences.Editor editor = sharedPreferences.edit();
                editor.putString("GROUP", group);
```

```
                editor.putInt("NUMBER", number);
                editor.putString("MOVIE", movie);
                editor.apply();

                Toast.makeText(MainActivity.this, "Данные сохранены!",
Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Листинг 1. Метод для сохранения

Далее был создан модуль «securesharedpreferences», в котором был создан экран отображения имени и фото писателя. При нажатии на кнопку «Сохранить» данные сохраняются на устройстве через secureSharedPreferences (см. Рисунки 4-5 и Листинг 2).

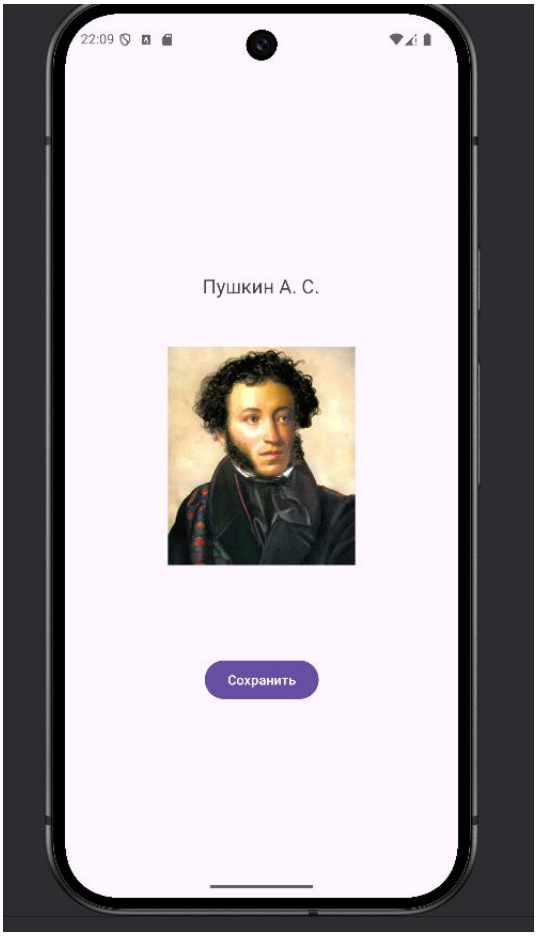

Рисунок 4. Экран приложения

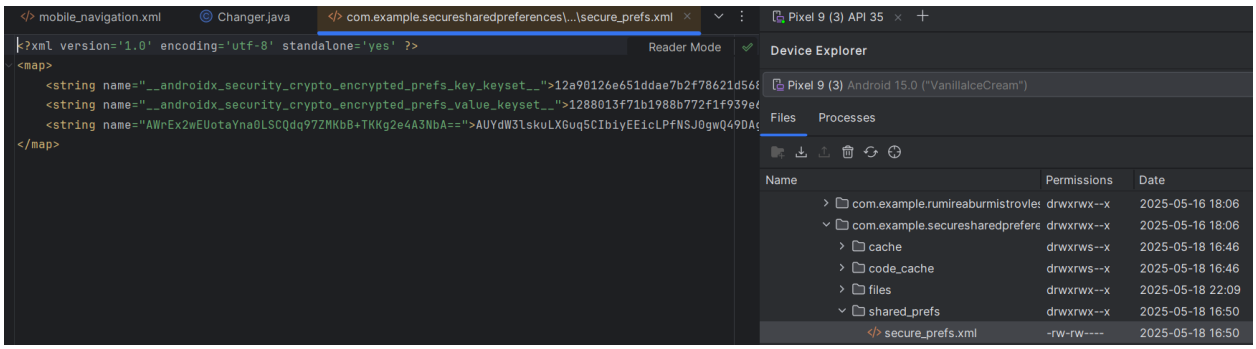

Рисунок 5. Пример данных

```
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;
    private TextView textView;
    private SharedPreferences secureSharedPreferences;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        textView = binding.textView;
        Button buttonSave = binding.button;

        try {
            String masterKey =
MasterKeys.getOrCreate(MasterKeys.AES256_GCM_SPEC);
            secureSharedPreferences = EncryptedSharedPreferences.create(
                    "secure_prefs",
                    masterKey,
                    this,

EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,

EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
            );
        } catch (GeneralSecurityException | IOException e) {
            throw new RuntimeException(e);
        }

        String WritterName = secureSharedPreferences.getString("WRITE_NAME",
"Пушкин А. С.");
        textView.setText(WritterName);

        buttonSave.setOnClickListener(v ->{
            secureSharedPreferences.edit().putString("WRITE_NAME",
textView.getText().toString()).apply();
            Toast.makeText(this, "Данные сохранены!",
Toast.LENGTH_SHORT).show();
```

```
        });

    }
}
```

Листинг 2. Код модуля

Далее был создан модуль «InternalFileStorage», в котором был создан экран для сохранения данных в «txt» файле (см. Рисунки 6-7 и Листинг 3)

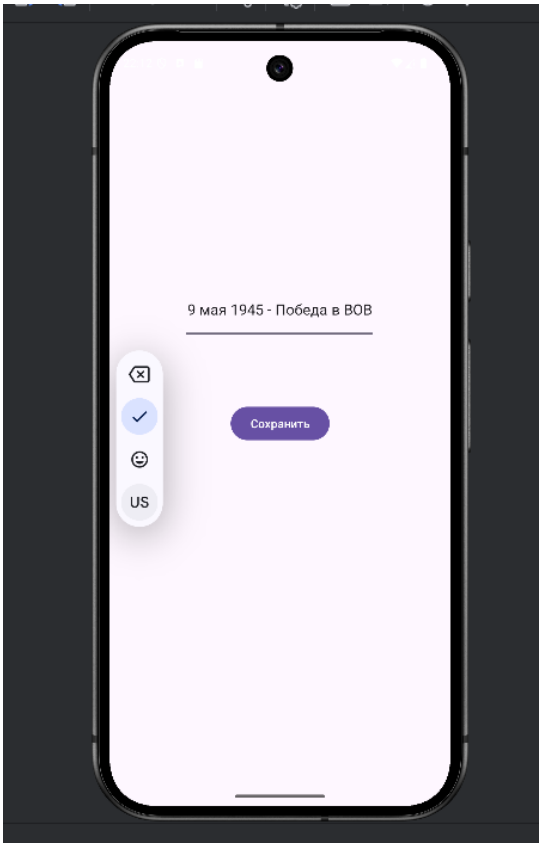

Рисунок 6. Окно приложения

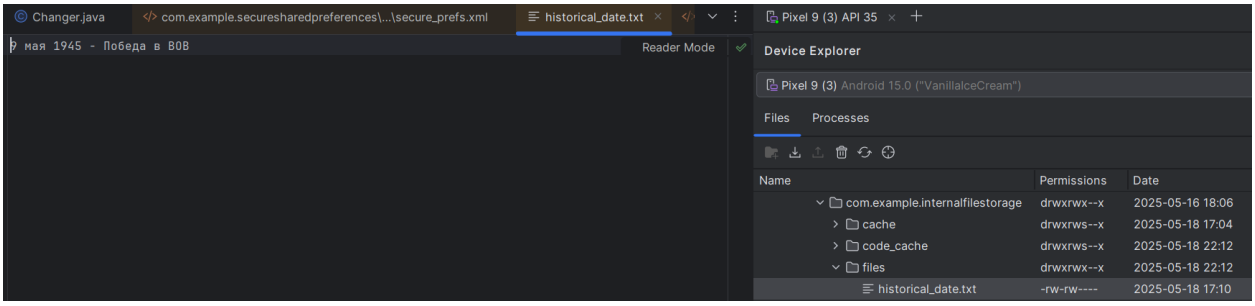

Рисунок 7. Пример сохранённых данных

```
public class MainActivity extends AppCompatActivity {

    private EditText editTextDate;
    private static final String FILENAME = "historical_date.txt";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        editTextDate = findViewById(R.id.editTextText);
        Button buttonSave = findViewById(R.id.button);

        buttonSave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String date = editTextDate.getText().toString();
                String data = date;

                // Запись в файл
                try (FileOutputStream fos = openFileOutput(FILENAME,
MODE_PRIVATE)) {
                    fos.write(data.getBytes());
                    Toast.makeText(MainActivity.this, "Файл сохранён!",
Toast.LENGTH_SHORT).show();
                } catch (IOException e) {
                    Toast.makeText(MainActivity.this, "Ошибка записи!",
Toast.LENGTH_SHORT).show();
                    e.printStackTrace();
                }
            }
        });
    }
}
```

Листинг 3. Код для сохранения в «txt» файле

Далее был создан модуль «Notebook», в котором было реализовано сохранение данных в файле и загрузка данных (см. Рисунки 8-9 и Листинг 4).

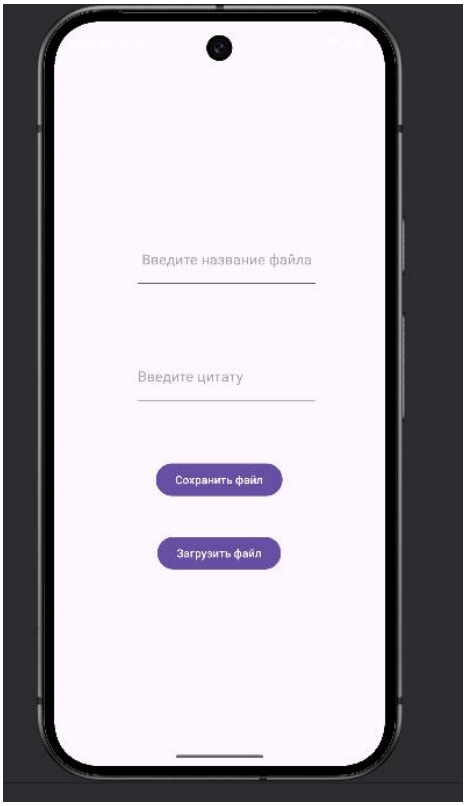

Рисунок 8. Пример загрузки файла

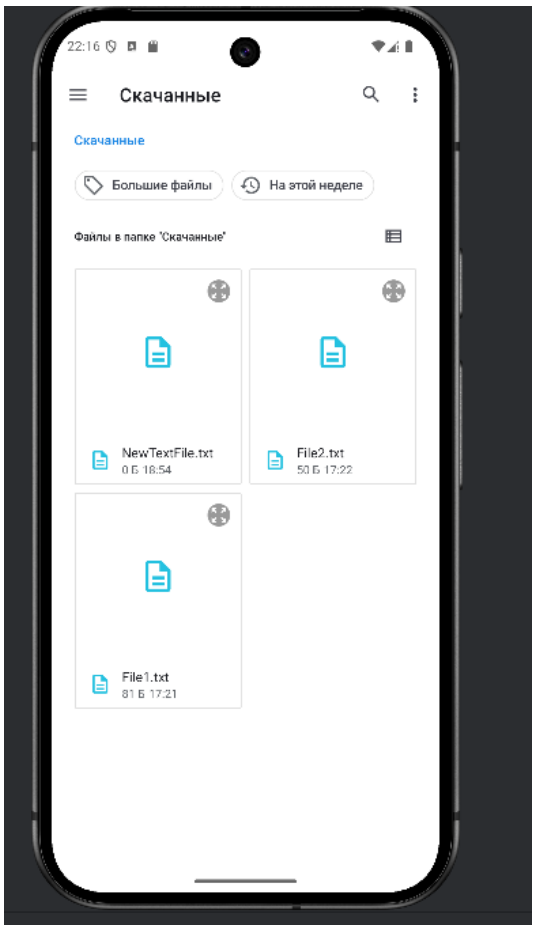

Рисунок 9. Загруженные файлы

```
public class MainActivity extends AppCompatActivity {
    private EditText editTextFileName, editTextQuote;
    private ActivityResultLauncher<Intent> saveFileLauncher;
    private ActivityResultLauncher<Intent> loadFileLauncher;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editTextFileName = findViewById(R.id.editTextText);
        editTextQuote = findViewById(R.id.editTextText2);
        Button buttonSave = findViewById(R.id.button);
        Button buttonLoad = findViewById(R.id.button2);

        saveFileLauncher = registerForActivityResult(
                new ActivityResultContracts.StartActivityForResult(),
                result -> {
                    if (result.getResultCode() == Activity.RESULT_OK &&
result.getData() != null) {
                        Uri uri = result.getData().getData();
                        if (uri != null) {
                            writeFileToUri(uri);
                        }
                    }
                }
        );
        loadFileLauncher = registerForActivityResult(
                new ActivityResultContracts.StartActivityForResult(),
                result -> {
                    if (result.getResultCode() == Activity.RESULT_OK &&
result.getData() != null) {
```

```
                        Uri uri = result.getData().getData();
                        if (uri != null) {
                            readFileFromUri(uri);
                        }
                    }
                }
        );
        buttonSave.setOnClickListener(v -> {
            String fileName = editTextFileName.getText().toString().trim();
            if (fileName.isEmpty()) {
                Toast.makeText(this, "Введите название файла!",
Toast.LENGTH_SHORT).show();
                return;
            }
            Intent intent = new Intent(Intent.ACTION_CREATE_DOCUMENT);
            intent.addCategory(Intent.CATEGORY_OPENABLE);
            intent.setType("text/plain");
            intent.putExtra(Intent.EXTRA_TITLE, fileName + ".txt");
            saveFileLauncher.launch(intent);
        });

        buttonLoad.setOnClickListener(v -> {
            Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
            intent.addCategory(Intent.CATEGORY_OPENABLE);
            intent.setType("text/plain");
            loadFileLauncher.launch(intent);
        });
    }
    private void writeFileToUri(Uri uri) {
        try {
            String quote = editTextQuote.getText().toString();
            OutputStream outputStream =
getContentResolver().openOutputStream(uri);
            outputStream.write(quote.getBytes());
            outputStream.close();
            Toast.makeText(this, "Файл сохранён!",
Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            Toast.makeText(this, "Ошибка записи!",
Toast.LENGTH_SHORT).show();
        }
    }
    private void readFileFromUri(Uri uri) {
        try {
            InputStream inputStream =
getContentResolver().openInputStream(uri);
            BufferedReader reader = new BufferedReader(new
InputStreamReader(inputStream));
            StringBuilder stringBuilder = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                stringBuilder.append(line).append("\n");
            }
            editTextQuote.setText(stringBuilder.toString().trim());
            Toast.makeText(this, "Файл загружен!",
Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            Toast.makeText(this, "Ошибка чтения!",
Toast.LENGTH_SHORT).show();
        }
    }
}
```

Листинг 4. Класс для сохранения и загрузки файлов

Затем был создан модуль «EmployeeDB», в котором была реализовано база данных для хранения информации о сотрудниках, записанные данные были выведены на экран приложения (см. Рисунок 10 и Листинги 5-8).

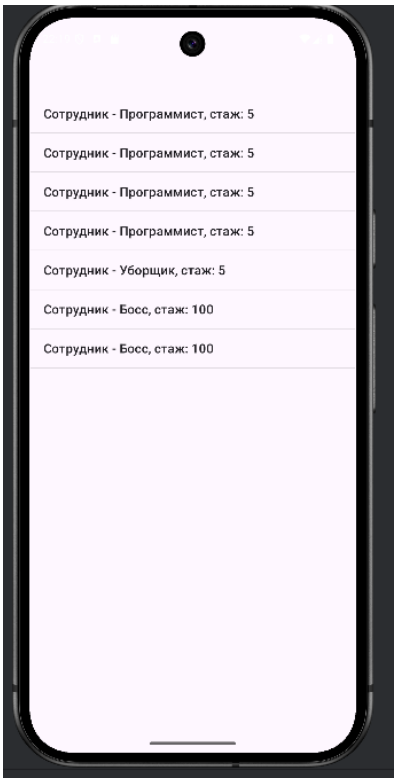

Рисунок 10. Сохранённые сотрудники

```
public class MainActivity extends AppCompatActivity {
    private AppDatabase db;
    private EmployeeDAO employeeDAO;
    private ListView listViewEmployees;
    private ArrayAdapter<String> adapter;
    private ArrayList<String> employeeStrings;

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        listViewEmployees = findViewById(R.id.list_view);

        db = App.getInstance().getDatabase();
        employeeDAO = db.employeeDAO();

        Employee employee = new Employee("Сотрудник", "Босс", 100);
        employeeDAO.insert(employee);

        List<Employee> employees = employeeDAO.getAll();
        Toast.makeText(this, "Сохранено сотрудников: " + employees.size(),
Toast.LENGTH_SHORT).show();

        employeeStrings = new ArrayList<>();
        for (Employee e : employees) {
            String s = e.name + " - " + e.profession + ", стаж: " + e.stage;
            employeeStrings.add(s);
```

```
        }

        adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, employeeStrings);
        listViewEmployees.setAdapter(adapter);
    }
}
```

Листинг 5. Добавление сотрудников и вывод на экран

```
@Entity(tableName = "employee_db")
public class Employee {
    @PrimaryKey(autoGenerate = true)
    public int id;
    public String name;
    public String profession;
    public int stage;

    public Employee(String name, String profession, int stage) {
        this.name = name;
        this.profession = profession;
        this.stage = stage;
    }
}
```

Листинг 6. Класс сотрудника

```
@Database(entities = {Employee.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
    public abstract EmployeeDAO employeeDAO();

    private static volatile AppDatabase INSTANCE;

    public static AppDatabase getInstance(Context context) {
        if (INSTANCE == null) {
            synchronized (AppDatabase.class) {
                if (INSTANCE == null) {
                    INSTANCE = Room.databaseBuilder(
                            context.getApplicationContext(),
                            AppDatabase.class,
                            "employee_db"
                    ).allowMainThreadQueries().build();
                }
            }
        }
        return INSTANCE;
    }
}
```

Листинг 7. Класс «AppDatabase»

```
public class App extends Application {
    private static App instance;
    private AppDatabase database;

    @Override
    public void onCreate() {
        super.onCreate();
        instance = this;
        database = AppDatabase.getInstance(this);
    }

    public static App getInstance() {
        return instance;
    }
```

```
    public AppDatabase getDatabase() {
        return database;
    }
}
```

Листинг 8. Класс «App»

Далее в проекте «MireaProject» **(B Lesson3)** было добавлено 2 фрагмента. В первом фрагменте «Профиль» было реализовано сохранение данных с шифрованием (см. Рисунок 11).

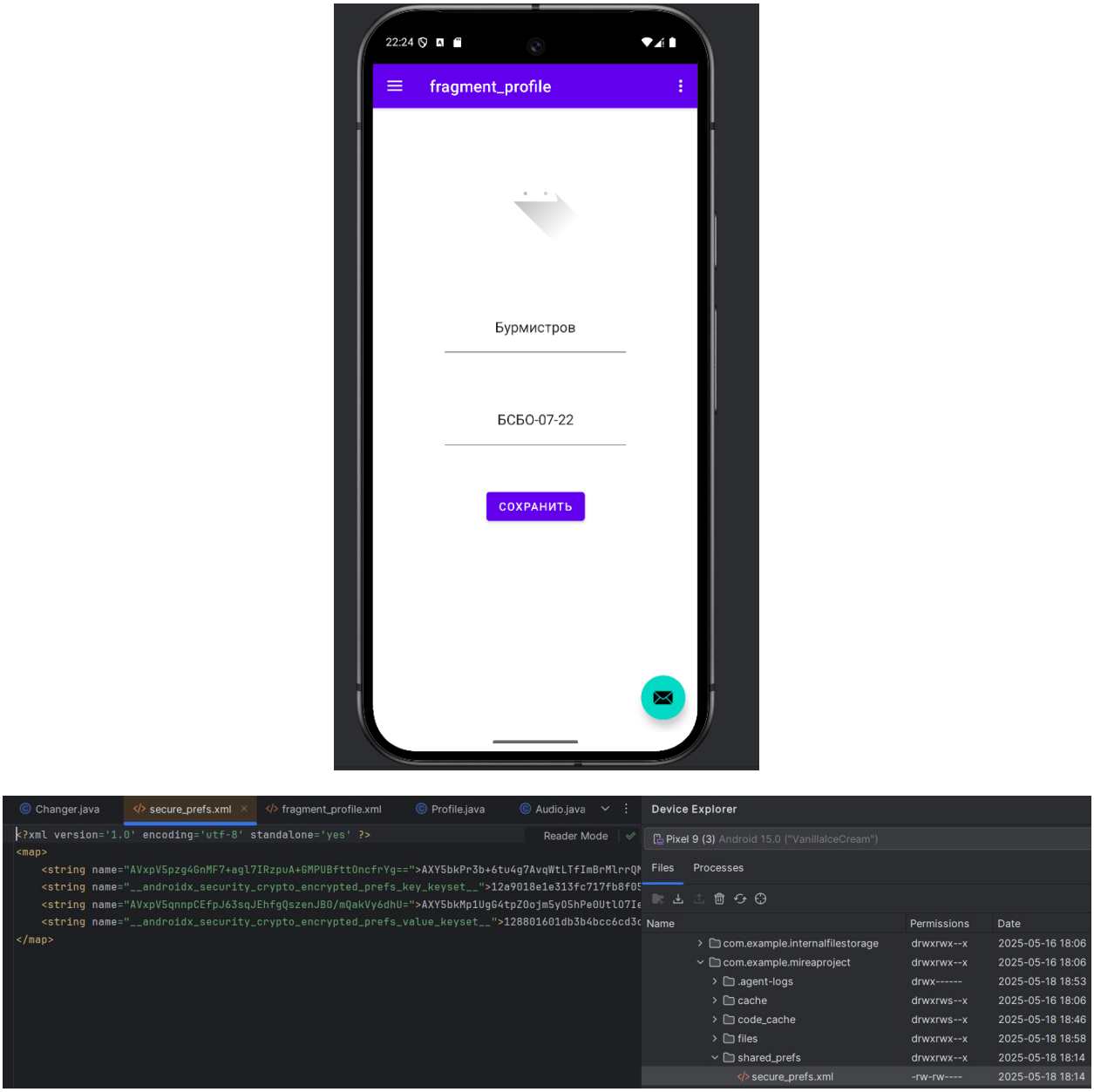

Рисунок 11. Пример данных

Во втором фрагменте была реализован функционал, связанный с обработкой файлов. При нажатии на кнопку «+» пользователь может выбрать файл, и далее в окне фрагмента изменить его имя и расширение (см. Рисунки 12-13 и Листинг 9).

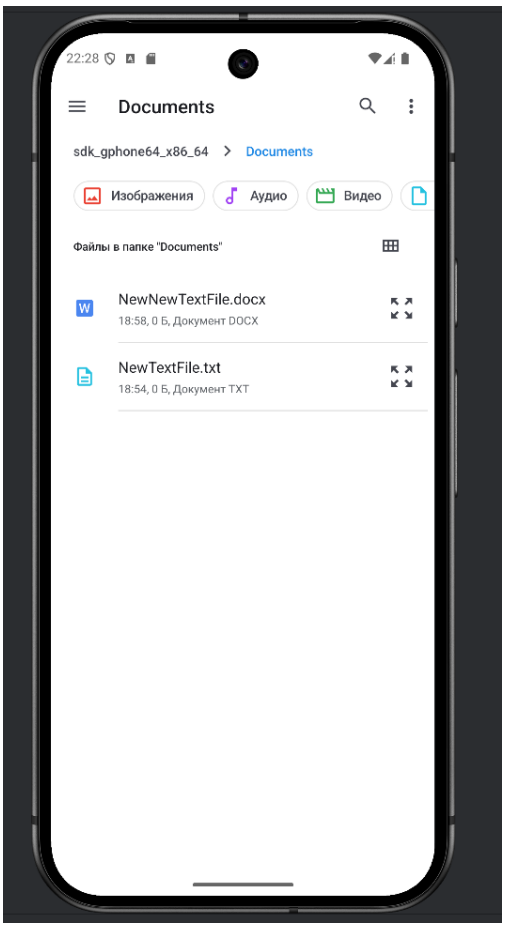

Рисунок 12. Выбор файла

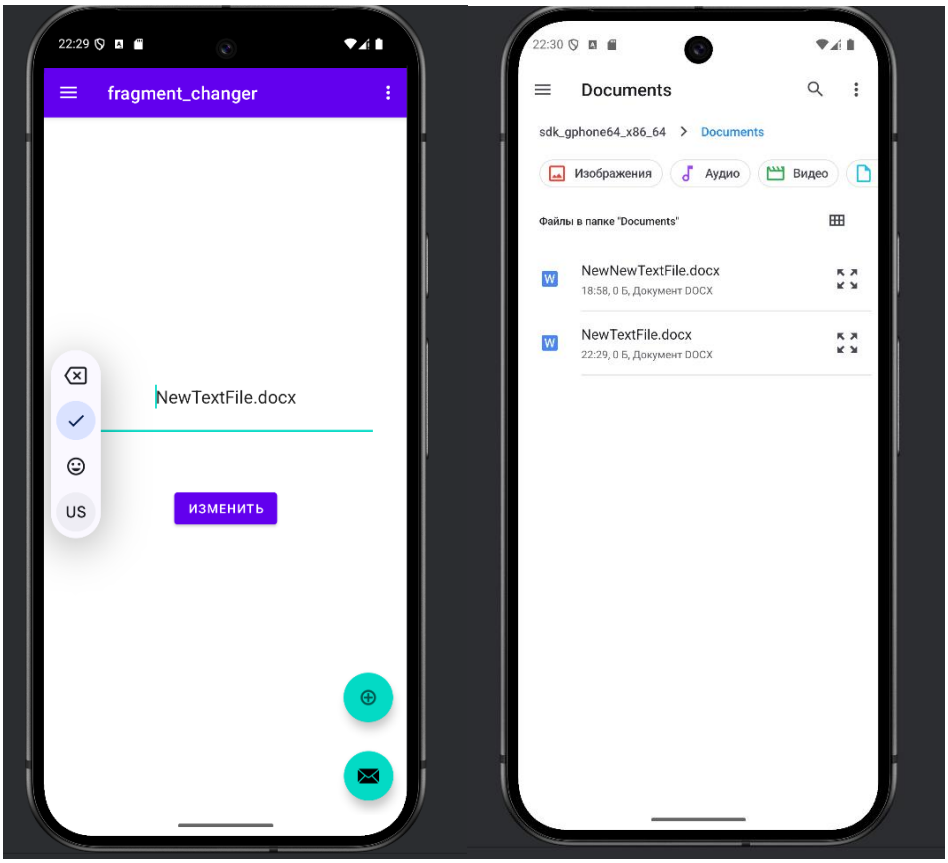

Рисунок 13. Изменение расширения файла

```
public class Changer extends Fragment {

    private FragmentChangerBinding binding;

    private Uri pickedFolderUri = null;
    private Uri currentFileUri = null;

    private ActivityResultLauncher<Uri> pickFolderLauncher;
    private ActivityResultLauncher<String[]> pickFileLauncher;

    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        binding = FragmentChangerBinding.inflate(inflater, container, false);
        return binding.getRoot();
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        pickFolderLauncher = registerForActivityResult(
                new ActivityResultContracts.OpenDocumentTree(),
                uri -> {
                    if (uri != null) {
                        pickedFolderUri = uri;

requireContext().getContentResolver().takePersistableUriPermission(uri,
                                Intent.FLAG_GRANT_READ_URI_PERMISSION |
Intent.FLAG_GRANT_WRITE_URI_PERMISSION);
                        Toast.makeText(requireContext(), "Папка выбрана",
Toast.LENGTH_SHORT).show();
                    }
                }
        );

        pickFileLauncher = registerForActivityResult(
                new ActivityResultContracts.OpenDocument(),
                uri -> {
                    if (uri != null) {
                        currentFileUri = uri;

requireContext().getContentResolver().takePersistableUriPermission(uri,
                                Intent.FLAG_GRANT_READ_URI_PERMISSION |
Intent.FLAG_GRANT_WRITE_URI_PERMISSION);

                        DocumentFile docFile =
DocumentFile.fromSingleUri(requireContext(), uri);
                        if (docFile != null && docFile.getName() != null) {

binding.editTextFileName.setText(docFile.getName());
                        }
                    }
                }
        );
    }

    @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle
savedInstanceState) {
        super.onViewCreated(view, savedInstanceState);

        binding.Select.setOnClickListener(v -> {
```

```
            if (pickedFolderUri == null) {

                pickFolderLauncher.launch(null);
            } else {

                pickFileLauncher.launch(new String[]{"*/*"});
            }
        });

        binding.buttonRename.setOnClickListener(v -> {
            if (pickedFolderUri == null) {
                Toast.makeText(requireContext(), "Сначала выберите папку",
Toast.LENGTH_SHORT).show();
                return;
            }
            if (currentFileUri == null) {
                Toast.makeText(requireContext(), "Сначала выберите файл",
Toast.LENGTH_SHORT).show();
                return;
            }
            String newName =
binding.editTextFileName.getText().toString().trim();
            if (TextUtils.isEmpty(newName)) {
                Toast.makeText(requireContext(), "Введите новое имя файла",
Toast.LENGTH_SHORT).show();
                return;
            }
            renameFileInFolder(pickedFolderUri, currentFileUri, newName);
        });
    }

    private void renameFileInFolder(Uri folderUri, Uri fileUri, String
newName) {
        try {
            DocumentFile pickedDir =
DocumentFile.fromTreeUri(requireContext(), folderUri);
            DocumentFile oldFile =
DocumentFile.fromSingleUri(requireContext(), fileUri);

            if (pickedDir == null || oldFile == null) {
                Toast.makeText(requireContext(), "Ошибка доступа к файлам",
Toast.LENGTH_SHORT).show();
                return;
            }

            if (!pickedDir.canWrite()) {
                Toast.makeText(requireContext(), "Нет доступа на запись в
выбранную папку", Toast.LENGTH_SHORT).show();
                return;
            }


            String extension = "";
            if (newName.contains(".")) {
                extension = newName.substring(newName.lastIndexOf('.') + 1);
            }
            String mimeType = getMimeTypeFromExtension(extension);
            if (mimeType == null) {
                mimeType = "application/octet-stream";
            }

            DocumentFile newFile = pickedDir.createFile(mimeType, newName);
            if (newFile == null) {
                Toast.makeText(requireContext(), "Не удалось создать новый
```

```
файл", Toast.LENGTH_SHORT).show();
                return;
            }

            try (InputStream in =
requireContext().getContentResolver().openInputStream(fileUri);
                 OutputStream out =
requireContext().getContentResolver().openOutputStream(newFile.getUri())) {
                if (in == null || out == null) {
                    Toast.makeText(requireContext(), "Ошибка открытия
потоков", Toast.LENGTH_SHORT).show();
                    return;
                }
                byte[] buffer = new byte[8192];
                int length;
                while ((length = in.read(buffer)) > 0) {
                    out.write(buffer, 0, length);
                }
            }

            boolean deleted = oldFile.delete();
            if (!deleted) {
                Toast.makeText(requireContext(), "Не удалось удалить старый
файл", Toast.LENGTH_SHORT).show();
            }

            Toast.makeText(requireContext(), "Файл переименован
(копированием)", Toast.LENGTH_SHORT).show();
            binding.editTextFileName.setText(newName);
            currentFileUri = newFile.getUri();

        } catch (Exception e) {
            Log.e("FileConverter", "Ошибка при переименовании", e);
            Toast.makeText(requireContext(), "Ошибка при переименовании
файла", Toast.LENGTH_SHORT).show();
        }
    }

    private String getMimeTypeFromExtension(String ext) {
        if (ext == null || ext.isEmpty()) return null;
        return
android.webkit.MimeTypeMap.getSingleton().getMimeTypeFromExtension(ext.toLowe
rCase());
    }
}
```

Листинг 9. Изменение файла