

ИТОГОВАЯ АТТЕСТАЦИЯ

ВЫПОЛНИЛ: БОРИСОВ КИРИЛЛ

ГРУППА: КИСП-23(1)

MainActivity.java

```
1 package com.example.exam;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import android.view.View;
6 import android.widget.AdapterView;
7 import android.widget.Button;
8 import android.widget.ListView;
9 import android.widget.TextView;
10 import android.widget.Toast;
11 import androidx.appcompat.app.AppCompatActivity;
12 import java.util.List;
13
14 public class MainActivity extends AppCompatActivity {
15     private TextView textViewWelcome, textViewNoPasswords;
16     private Button buttonAddPassword;
17     private ListView listViewPasswords;
18     private DatabaseHelper db;
19     private String username;
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24         setContentView(R.layout.activity_main);
25
26         db = new DatabaseHelper(this);
27         username = getIntent().getStringExtra("USERNAME");
28         initView();
29         setupListeners();
30         loadPasswords();
31     }
32
33     private void initView() {
34         textViewWelcome = findViewById(R.id.textViewWelcome);
35         textViewNoPasswords = findViewById(R.id.textViewNoPasswords);
36         buttonAddPassword = findViewById(R.id.buttonAddPassword);
37         listViewPasswords = findViewById(R.id.listViewPasswords);
38
39         textViewWelcome.setText("Добро пожаловать, " + username + "!");
40     }
41
42     private void setupListeners() {
43         buttonAddPassword.setOnClickListener(new View.OnClickListener() {
44             @Override
45             public void onClick(View v) {
46                 Intent intent = new Intent(MainActivity.this, AddPasswordActivity.class);
47                 intent.putExtra("USERNAME", username);
48                 startActivity(intent);
49             }
50         });
51     }
52 }
```

```
listViewPasswords.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        PasswordEntry entry = (PasswordEntry) parent.getItemAtPosition(position);
        Toast.makeText(MainActivity.this,
            "Сервис: " + entry.getService() +
            "\nЛогин: " + entry.getLogin() +
            "\nПароль: " + entry.getPassword(),
            Toast.LENGTH_LONG).show();
    }
});

private void loadPasswords() {
    // Здесь нужно получить user_id по username, но для простоты будем передавать username
    List<PasswordEntry> passwordList = db.getAllPasswords(1); // В реальном приложении нужно получить правильный user_id

    if (passwordList.isEmpty()) {
        textViewNoPasswords.setVisibility(View.VISIBLE);
        listViewPasswords.setVisibility(View.GONE);
    } else {
        textViewNoPasswords.setVisibility(View.GONE);
        listViewPasswords.setVisibility(View.VISIBLE);

        PasswordAdapter adapter = new PasswordAdapter(this, passwordList);
        listViewPasswords.setAdapter(adapter);
    }
}

@Override
protected void onResume() {
    super.onResume();
    loadPasswords();
}
}
```

DatabaseHelper.java

```
1 package com.example.exan;
2
3 import android.content.ContentValues;
4 import android.content.Context;
5 import android.database.Cursor;
6 import android.database.sqlite.SQLiteDatabase;
7 import android.database.sqlite.SQLiteOpenHelper;
8 import android.os.Build;
9 import java.security.MessageDigest;
10 import java.security.NoSuchAlgorithmException;
11 import java.util.ArrayList;
12 import java.util.List;
13 import javax.crypto.Cipher;
14 import javax.crypto.spec.SecretKeySpec;
15
16 public class DatabaseHelper extends SQLiteOpenHelper { 8 usages
17     private static final String DATABASE_NAME = "PasswordManager.db"; 1 usage
18     private static final int DATABASE_VERSION = 1; 1 usage
19
20     // Таблица пользователей
21     private static final String TABLE_USERS = "users"; 6 usages
22     private static final String COLUMN_USER_ID = "id"; 4 usages
23     private static final String COLUMN_USERNAME = "username"; 4 usages
24     private static final String COLUMN_PASSWORD = "password"; 3 usages
25
26     // Таблица паролей
27     private static final String TABLE_PASSWORDS = "passwords"; 4 usages
28     private static final String COLUMN_PASSWORD_ID = "id"; 2 usages
29     private static final String COLUMN_USER_ID_FK = "user_id"; 4 usages
30     private static final String COLUMN_SERVICE = "service"; 4 usages
31     private static final String COLUMN_LOGIN = "login"; 3 usages
32     private static final String COLUMN_PASSWORD_VALUE = "password_value"; 3 usages
33     private static final String COLUMN_NOTES = "notes"; 3 usages
34
35     // Ключ для шифрования (в реальном приложении должен быть более безопасным)
36     private static final String ENCRYPTION_KEY = "MySuperSecretKey123"; 1 usage
37
38     public DatabaseHelper(Context context) { 4 usages
39         super(context, DATABASE_NAME, (Factory) null, DATABASE_VERSION);
40     }
41
42     @Override
43     public void onCreate(SQLiteDatabase db) {
44         // Создание таблиц пользователей
45         String CREATE_USERS_TABLE = "CREATE TABLE " + TABLE_USERS + "("
46             + COLUMN_USER_ID + " INTEGER PRIMARY KEY AUTOINCREMENT,"
47             + COLUMN_USERNAME + " TEXT UNIQUE,"
48             + COLUMN_PASSWORD + " TEXT" + ")";
49         db.execSQL(CREATE_USERS_TABLE);
50
51         // Создание таблицы паролей
```

```
16 public class DatabaseHelper extends SQLiteOpenHelper { 8 usages
43     public void onCreate(SQLiteDatabase db) {
51         // Создание таблицы паролей
52         String CREATE_PASSWORDS_TABLE = "CREATE TABLE " + TABLE_PASSWORDS + "("
53             + COLUMN_PASSWORD_ID + " INTEGER PRIMARY KEY AUTOINCREMENT,"
54             + COLUMN_USER_ID_FK + " INTEGER,"
55             + COLUMN_SERVICE + " TEXT,"
56             + COLUMN_LOGIN + " TEXT,"
57             + COLUMN_PASSWORD_VALUE + " TEXT,"
58             + COLUMN_NOTES + " TEXT,"
59             + "FOREIGN KEY(" + COLUMN_USER_ID_FK + ") REFERENCES "
60             + TABLE_USERS + "(" + COLUMN_USER_ID + ")" + ")";
61         db.execSQL(CREATE_PASSWORDS_TABLE);
62     }
63
64     @Override
65     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
66         db.execSQL("DROP TABLE IF EXISTS " + TABLE_USERS);
67         db.execSQL("DROP TABLE IF EXISTS " + TABLE_PASSWORDS);
68         onCreate(db);
69     }
70
71     // Регистрация пользователя
72     public boolean registerUser(String username, String password) { 1 usage
73         SQLiteDatabase db = this.getWritableDatabase();
74         ContentValues values = new ContentValues();
75
76         // Проверка на существование пользователя
77         if (checkUserExists(username)) {
78             return false;
79         }
80
81         values.put(COLUMN_USERNAME, username);
82         values.put(COLUMN_PASSWORD, hashPassword(password));
83
84         long result = db.insert(TABLE_USERS, null, values);
85         return result != -1;
86     }
87
88     // Проверка пользователя
89     public boolean checkUser(String username, String password) { 1 usage
90         SQLiteDatabase db = this.getReadableDatabase();
91         String[] columns = {COLUMN_USER_ID};
92         String selection = COLUMN_USERNAME + " = ?" + " AND " + COLUMN_PASSWORD + " = ?";
93         String[] selectionArgs = {username, hashPassword(password)};
94
95         Cursor cursor = db.query(TABLE_USERS, columns, selection, selectionArgs, null, null, null);
96         int count = cursor.getCount();
97         cursor.close();
98
99         return count > 0;
```

```

80
81 // Проверка существования пользователя
82 private boolean checkUserExists(String username) { 1 usage
83     SQLiteDatabase db = this.getReadableDatabase();
84     String[] columns = {COLUMN_USER_ID};
85     String selection = COLUMN_USERNAME + " = ?";
86     String[] selectionArgs = {username};
87
88     Cursor cursor = db.query(TABLE_USERS, columns, selection, selectionArgs, null, null, null, null);
89     int count = cursor.getCount();
90     cursor.close();
91
92     return count > 0;
93 }
94
95 // Хэширование пароля
96 @ private String hashPassword(String password) { 2 usages
97     try {
98         MessageDigest digest = MessageDigest.getInstance("SHA-256");
99         byte[] hash = digest.digest(password.getBytes());
100         return Base64.encodeToString(hash, Base64.DEFAULT);
101     } catch (NoSuchAlgorithmException e) {
102         e.printStackTrace();
103         return null;
104     }
105 }
106
107 // Добавление пароля
108 public boolean addPassword(int userId, String service, String login, String password, String notes) { 1 usage
109     SQLiteDatabase db = this.getWritableDatabase();
110     ContentValues values = new ContentValues();
111
112     try {
113         String encryptedPassword = encrypt(password);
114         values.put(COLUMN_USER_ID_FK, userId);
115         values.put(COLUMN_SERVICE, service);
116         values.put(COLUMN_LOGIN, login);
117         values.put(COLUMN_PASSWORD_VALUE, encryptedPassword);
118         values.put(COLUMN_NOTES, notes);
119
120         long result = db.insert(TABLE_PASSWORDS, null, values);
121         return result != -1;
122     } catch (Exception e) {
123         e.printStackTrace();
124         return false;
125     }
126 }
127
128 // Получение всех паролей пользователя
129 public List<PasswordEntry> getAllPasswords(int userId) { 1 usage

```

```

130 public class DatabaseHelper extends SQLiteOpenHelper { 8 usages
131     // Получение всех паролей пользователя
132     public List<PasswordEntry> getAllPasswords(int userId) { 1 usage
133         List<PasswordEntry> passwordList = new ArrayList<>();
134         SQLiteDatabase db = this.getReadableDatabase();
135
136         String[] columns = {COLUMN_PASSWORD_ID, COLUMN_SERVICE, COLUMN_LOGIN, COLUMN_PASSWORD_VALUE, COLUMN_NOTES};
137         String selection = COLUMN_USER_ID_FK + " = ?";
138         String[] selectionArgs = {String.valueOf(userId)};
139
140         Cursor cursor = db.query(TABLE_PASSWORDS, columns, selection, selectionArgs, null, null, null, null);
141
142         if (cursor.moveToFirst()) {
143             do {
144                 try {
145                     PasswordEntry entry = new PasswordEntry();
146                     entry.setId(cursor.getInt(0));
147                     entry.setService(cursor.getString(1));
148                     entry.setLogin(cursor.getString(2));
149                     entry.setPassword(decrypt(cursor.getString(3)));
150                     entry.setNotes(cursor.getString(4));
151                     passwordList.add(entry);
152                 } catch (Exception e) {
153                     e.printStackTrace();
154                 }
155             } while (cursor.moveToNext());
156         }
157         cursor.close();
158         return passwordList;
159     }
160
161     // Шифрование
162     @ private String encrypt(String data) throws Exception { 1 usage
163         SecretKeySpec key = generateKey();
164         Cipher cipher = Cipher.getInstance("AES");
165         cipher.init(Cipher.ENCRYPT_MODE, key);
166         byte[] encryptedData = cipher.doFinal(data.getBytes());
167         return Base64.encodeToString(encryptedData, Base64.DEFAULT);
168     }
169
170     // Дешифрование
171     @ private String decrypt(String encryptedData) throws Exception { 1 usage
172         SecretKeySpec key = generateKey();
173         Cipher cipher = Cipher.getInstance("AES");
174         cipher.init(Cipher.DECRYPT_MODE, key);
175         byte[] decodedData = Base64.decode(encryptedData, Base64.DEFAULT);
176         byte[] decryptedData = cipher.doFinal(decodedData);
177         return new String(decryptedData);
178     }
179 }

```

```

@ // Генерация ключа
private SecretKeySpec generateKey() throws Exception { 2 usages
    byte[] key = ENCRYPTION_KEY.getBytes(charsetName: "UTF-8");
    MessageDigest sha = MessageDigest.getInstance(algorithm: "SHA-256");
    key = sha.digest(key);
    return new SecretKeySpec(key, algorithm: "AES");
}
}

```

LoginActivity.java

```
package com.example.exam;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class LoginActivity extends AppCompatActivity {
    private EditText editTextUsername, editTextPassword;
    private Button buttonLogin;
    private TextView textViewRegister;
    private DatabaseHelper db;

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

        db = new DatabaseHelper(this);
        initView();
        setupListeners();
    }

    private void initView() {
        editTextUsername = findViewById(R.id.editTextUsername);
        editTextPassword = findViewById(R.id.editTextPassword);
        buttonLogin = findViewById(R.id.buttonLogin);
        textViewRegister = findViewById(R.id.textViewRegister);
    }

    private void setupListeners() {
        buttonLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = editTextUsername.getText().toString().trim();
                String password = editTextPassword.getText().toString().trim();

                if (username.isEmpty() || password.isEmpty()) {
                    Toast.makeText(LoginActivity.this, "Username or password is empty", Toast.LENGTH_SHORT).show();
                    return;
                }

                boolean checkUser = db.checkUser(username, password);
                if (checkUser) {
                    Toast.makeText(LoginActivity.this, "Login Successful", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(LoginActivity.this, MainActivity.class);
                    intent.putExtra("username", username);
                    startActivity(intent);
                } else {
                    Toast.makeText(LoginActivity.this, "User not found", Toast.LENGTH_SHORT).show();
                }
            }
        });

        textViewRegister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(LoginActivity.this, RegisterActivity.class));
            }
        });
    }
}
```

```
textViewRegister.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        startActivity(new Intent(LoginActivity.this, RegisterActivity.class));
    }
});
}
```

PasswordAdapter.java

```
DatabaseHelper.java LoginActivity.java MainActivity.java PasswordAdapter.java PasswordAdapter.java

package com.example.exam;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
import java.util.List;

public class PasswordAdapter extends ArrayAdapter<PasswordEntry> {
    public PasswordAdapter(Context context, List<PasswordEntry> passwords) {
        super(context, 0, passwords);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        PasswordEntry entry = getItem(position);

        if (convertView == null) {
            convertView = LayoutInflater.from(getContext())
                .inflate(android.R.layout.simple_list_item_1, parent, false);
        }

        TextView textView = convertView.findViewById(android.R.id.text1);
        textView.setText(entry.getService() + " (" + entry.getLogin() + ")");

        return convertView;
    }
}
```

PasswordEntry.java

```

1 package com.example.exam;
2
3 public class PasswordEntry { 10 usages
4     private int id; 2 usages
5     private String service; 2 usages
6     private String login; 2 usages
7     private String password; 2 usages
8     private String notes; 2 usages
9
10    // Геттеры и сеттеры
11    public int getId() { return id; }
12    public void setId(int id) { this.id = id; }
13    public String getService() { return service; } 2 usages
14    public void setService(String service) { this.service = service; } 1 usage
15    public String getLogin() { return login; } 2 usages
16    public void setLogin(String login) { this.login = login; } 1 usage
17    public String getPassword() { return password; } 1 usage
18    public void setPassword(String password) { this.password = password; } 1 usage
19    public String getNotes() { return notes; } no usages
20    public void setNotes(String notes) { this.notes = notes; } 1 usage
21 }

```

RegisterActivity.java

```

11 public class RegisterActivity extends AppCompatActivity {
12     private EditText editTextUsername, editTextPassword, editTextConfirmPassword; 2 usages
13     private Button buttonRegister; 2 usages
14     private DatabaseHelper db; 2 usages
15
16     @Override
17     protected void onCreate(Bundle savedInstanceState) {
18         super.onCreate(savedInstanceState);
19         setContentView(R.layout.activity_register);
20
21         db = new DatabaseHelper( context, this);
22         initView();
23         setupListeners();
24     }
25
26     private void initView() { 1 usage
27         editTextUsername = findViewById(R.id.editTextUsername);
28         editTextPassword = findViewById(R.id.editTextPassword);
29         editTextConfirmPassword = findViewById(R.id.editTextConfirmPassword);
30         buttonRegister = findViewById(R.id.buttonRegister);
31     }
32
33     private void setupListeners() { 1 usage
34         buttonRegister.setOnClickListener(new View.OnClickListener() {
35             @Override
36             public void onClick(View v) {
37                 String username = editTextUsername.getText().toString().trim();
38                 String password = editTextPassword.getText().toString().trim();
39                 String confirmPassword = editTextConfirmPassword.getText().toString().trim();
40
41                 if (username.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {
42                     Toast.makeText( context, RegisterActivity.this, text: "Заполните все поля", Toast.LENGTH_SHORT).show();
43                     return;
44                 }
45
46                 if (!password.equals(confirmPassword)) {
47                     Toast.makeText( context, RegisterActivity.this, text: "Пароли не совпадают", Toast.LENGTH_SHORT).show();
48                     return;
49                 }
50
51                 boolean registered = db.registerUser(username, password);
52                 if (registered) {
53                     Toast.makeText( context, RegisterActivity.this, text: "Регистрация успешна", Toast.LENGTH_SHORT).show();
54                     finish();
55                 } else {
56                     Toast.makeText( context, RegisterActivity.this, text: "Пользователь уже существует", Toast.LENGTH_SHORT).show();
57                 }
58             }
59         });
60     }

```

activity_add_password.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical"
6     android:padding="16dp">
7
8     <TextView
9         android:layout_width="wrap_content"
10        android:layout_height="wrap_content"
11        android:text="Роботенне Hoboro napona"
12        android:textSize="18sp"
13        android:layout_marginBottom="16dp"/>
14
15     <TextView
16         android:layout_width="wrap_content"
17         android:layout_height="wrap_content"
18         android:text="Cepace/call:"/>
19
20     <EditText
21         android:id="@+id/editTextService"
22         android:layout_width="match_parent"
23         android:layout_height="wrap_content"
24         android:hint="Hapomepi Google"
25         android:layout_marginBottom="16dp"/>
26
27     <TextView
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content"
30         android:text="Honyu/email:"/>
31
32     <EditText
33         android:id="@+id/editTextLogin"
34         android:layout_width="match_parent"
35         android:layout_height="wrap_content"
36         android:hint="Bacure Honyu"
37         android:layout_marginBottom="16dp"/>
38
39     <TextView
40         android:layout_width="wrap_content"
41         android:layout_height="wrap_content"
42         android:text="Napona:"/>
43
44     <EditText
45         android:id="@+id/editTextPassword"
46         android:layout_width="match_parent"
47         android:layout_height="wrap_content"
48         android:hint="Bacure napona"
49         android:inputType="textPassword"
50         android:layout_marginBottom="16dp"/>
```

activity_login.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical"
6     android:padding="16dp"
7     android:gravity="center">
8
9     <TextView
10        android:layout_width="wrap_content"
11        android:layout_height="wrap_content"
12        android:text="Honyu napona"
13        android:textSize="24sp"
14        android:layout_marginBottom="32dp"/>
15
16     <EditText
17         android:id="@+id/editTextUsername"
18         android:layout_width="match_parent"
19         android:layout_height="wrap_content"
20         android:hint="Honyu"
21         android:layout_marginBottom="16dp"/>
22
23     <EditText
24         android:id="@+id/editTextPassword"
25         android:layout_width="match_parent"
26         android:layout_height="wrap_content"
27         android:hint="Napona"
28         android:inputType="textPassword"
29         android:layout_marginBottom="16dp"/>
30
31     <Button
32         android:id="@+id/buttonLogin"
33         android:layout_width="match_parent"
34         android:layout_height="wrap_content"
35         android:text="BoiTy"
36         android:layout_marginBottom="16dp"/>
37
38     <TextView
39         android:id="@+id/textViewRegister"
40         android:layout_width="wrap_content"
41         android:layout_height="wrap_content"
42         android:text="Ewe het ankayita? Bapercapupvitec"
43         android:textColor="@color/blue"
44         android:textStyle="bold"/>
45 </LinearLayout>
```

activity_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical"
6     android:padding="16dp">
7
8     <TextView
9         android:id="@+id/textViewWelcome"
10        android:layout_width="wrap_content"
11        android:layout_height="wrap_content"
12        android:textSize="18sp"
13        android:layout_marginBottom="16dp"/>
14
15     <Button
16         android:id="@+id/buttonAddPassword"
17         android:layout_width="match_parent"
18         android:layout_height="wrap_content"
19         android:text="Добавить пароль"
20         android:layout_marginBottom="16dp"/>
21
22     <ListView
23         android:id="@+id/listViewPasswords"
24         android:layout_width="match_parent"
25         android:layout_height="0dp"
26         android:layout_weight="1"/>
27
28     <TextView
29         android:id="@+id/textViewNoPasswords"
30         android:layout_width="wrap_content"
31         android:layout_height="wrap_content"
32         android:text="В базе пока нет сохраненных паролей"
33         android:visibility="gone"
34         android:layout_gravity="center"/>
35 </LinearLayout>
```

activity_register.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical"
6     android:padding="16dp"
7     android:gravity="center">
8
9     <TextView
10        android:layout_width="wrap_content"
11        android:layout_height="wrap_content"
12        android:text="Регистрация"
13        android:textSize="24sp"
14        android:layout_marginBottom="32dp"/>
15
16     <EditText
17         android:id="@+id/editTextUsername"
18         android:layout_width="match_parent"
19         android:layout_height="wrap_content"
20         android:hint="Имя"
21         android:layout_marginBottom="16dp"/>
22
23     <EditText
24         android:id="@+id/editTextPassword"
25         android:layout_width="match_parent"
26         android:layout_height="wrap_content"
27         android:hint="Пароль"
28         android:inputType="textPassword"
29         android:layout_marginBottom="16dp"/>
30
31     <EditText
32         android:id="@+id/editTextConfirmPassword"
33         android:layout_width="match_parent"
34         android:layout_height="wrap_content"
35         android:hint="Подтвердите пароль"
36         android:inputType="textPassword"
37         android:layout_marginBottom="16dp"/>
38
39     <Button
40         android:id="@+id/buttonRegister"
41         android:layout_width="match_parent"
42         android:layout_height="wrap_content"
43         android:text="Зарегистрироваться"/>
44 </LinearLayout>
```


Результаты

Менеджер паролей

Логин

Пароль

Войти

[Ещё нет аккаунта? Зарегистрируйтесь](#)

Регистрация

BorisovKirill

...

...

Зарегистрироваться

Добро пожаловать, BorisovKirill!

Добавить пароль

Google (dota2govno@gmail.com)

Добавление нового пароля

Сервис/сайт:

Google

Логин/email:

javatop1@gmail.com

Пароль:

...

Заметки (необязательно):

123

Сохранить

✕

←

😊

≡

Добро пожаловать, BorisovKirill!

Добавить пароль

Google (dota2govno@gmail.com)

Google (javatop1@gmail.com)