

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки

Кафедра МБІС

ЛАБОРАТОРНА РОБОТА №3

з дисципліни «Засоби програмування та захисту веб-додатків»
на тему: «Розробка web-додатку на Flask»

Студента 2 курсу групи ЗКІТС-236
напряму підготовки 125
спеціальності «Кібербезпека та захист
інформації»
Яцюка Дениса Володимировича
Перевірив: Присяжний Дмитро Петрович

Вінниця 2025

Мета роботи: *Створити повнофункціональний веб-додаток для ведення рейтингу хакерів з можливостями реєстрації, авторизації, управління профілями та адміністрування системи..*

1 РОЗРОБКА СТРУКТУРИ ДОДАТКУ

У процесі розробки було створено веб-додаток на основі фреймворку Flask. Основний файл додатку `app.py` містить усю логіку контролерів та маршрутизацію. Для стилізації застосовано кібер-панк дизайн з використанням пікселізованого шрифту "Press Start 2P" та неонових кольорів.

Створено базовий шаблон `base.html`, який містить навігацію та загальну структуру сторінок:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Hacker Leaderboard</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
  <link href="https://fonts.googleapis.com/css2?family=Press+Start+2P&display=swap"
rel="stylesheet">
</head>
<body>
  <header>
    <nav>
      <ul>
        <li><a href="{{ url_for('index') }}">Home</a></li>
        {% if session.get('logged_in') %}
          <li><a href="{{ url_for('edit_profile') }}">Edit Profile</a></li>
          <li><a href="{{ url_for('logout') }}">Logout</a></li>
        {% endif %}
      </ul>
    </nav>
  </header>
  ...
</body>
</html>
```

Головна сторінка index.html відображає таблицю рейтингу з можливістю надавання поваги (respect) іншим користувачам:

```
<table>
  <thead>
    <tr>
      <th>Username</th>
      <th>Country</th>
      <th>Rank</th>
      <th>Challenges Completed</th>
      <th>Respect</th>
    </tr>
  </thead>
  <tbody>
    {% for entry in leaderboard %}
    <tr>
      <td>{{ entry.username }}</td>
      <td>{{ entry.country }}</td>
      <td>{{ entry.rank }}</td>
      ...
    </tr>
    {% endfor %}
  </tbody>
</table>
```

Система ранжування реалізована через функцію get_rank(), яка визначає рівень користувача на основі кількості виконаних завдань:

```
def get_rank(challenges_completed):
    challenges_completed = int(challenges_completed)
    if challenges_completed >= 30:
        return "Legend"
    elif challenges_completed >= 20:
        return "Guru"
    elif challenges_completed >= 12:
        return "Elite Hacker"
    elif challenges_completed >= 6:
        return "Pro Hacker"
    elif challenges_completed >= 3:
        return "Hacker"
    else:
        return "Script Kiddie"
```

The screenshot shows a web application interface with a dark background. At the top, there is a navigation bar with three buttons: "Home", "Login", and "Register". The main content area features a central form titled "Register New Hacker". The form contains four input fields: "Username:", "Password:", "Confirm Password:", and "Country:". Below these fields is a "Register" button. The form is styled with a light gray border and a white background. The footer of the page displays the copyright notice "© 2025 Hacker Leaderboard".

Home Login Register

Register New Hacker

Username:

Password:

Confirm Password:

Country:

Register

© 2025 Hacker Leaderboard

Рисунок 1.1 – Вікно авторизації

This screenshot shows the same "Register New Hacker" form as in Figure 1.1, but with sample data entered into the input fields. The "Username" field contains "venator17", the "Password" and "Confirm Password" fields contain "*****", and the "Country" field contains "Ukraine". The "Register" button remains visible at the bottom of the form. The navigation bar and footer are identical to the previous figure.

Home Login Register

Register New Hacker

Username:

Password:

Confirm Password:

Country:

Register

© 2025 Hacker Leaderboard

Рисунок 1.2 – Вікно реєстрації



Рисунок 1.3 – Головна сторінка

На рисунках 1.1 – 1.3 зображено головні сторінки вебдодатку, які повністю працездатні та виконують поставлені на них функції.

2 НАПОВНЕННЯ ДОДАТКУ

Для цього проєкту була використана БД SQLite, оскільки вона не потребує окремого сервера та ідеально підходить для навчальних проєктів. Створено базу даних із назвою `hacker_leaderboard.db`. Таблиці бази даних повністю відповідають функціональним вимогам системи рейтингу хакерів.

Структура бази даних включає три основні моделі (`users`, `leaderboard`, `likes`):

```
db.execute("""
    CREATE TABLE IF NOT EXISTS users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        username TEXT UNIQUE NOT NULL,
        password TEXT NOT NULL,
        is_admin INTEGER DEFAULT 0
    )
""")
db.execute("""
    CREATE TABLE IF NOT EXISTS leaderboard (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        username TEXT UNIQUE NOT NULL,
        country TEXT NOT NULL,
        challenges_completed INTEGER DEFAULT 0,
        respect INTEGER DEFAULT 0,
        user_id INTEGER,
        FOREIGN KEY (user_id) REFERENCES users (id)
    )
""")
db.execute("""
    CREATE TABLE IF NOT EXISTS likes (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        liker_user_id INTEGER NOT NULL,
        liked_leaderboard_id INTEGER NOT NULL,
        FOREIGN KEY (liker_user_id) REFERENCES users (id),
        FOREIGN KEY (liked_leaderboard_id) REFERENCES leaderboard (id),
        UNIQUE(liker_user_id, liked_leaderboard_id)
    )
""")
```

Ініціалізація бази даних відбувається через функцію `init_db()`, яка також створює адміністративний обліковий запис з логіном `"admin"` та паролем

"super_secure_admin_password". Для роботи з базою даних використовується контекстний менеджер Flask через функції `get_db()` та `close_connection()`. створити, яке кодування символів використовувати, які таблиці створити та з якими полями.

```
def get_db():
    db = getattr(g, '_database', None)
    if db is None:
        db = g._database = sqlite3.connect(DATABASE)
        db.row_factory = sqlite3.Row
    return db

@app.teardown_appcontext
def close_connection(exception):
    db = getattr(g, '_database', None)
    if db is not None:
        db.close()
```

3 РОЗРОБКА ФУНКЦІОНАЛУ ДОДАТКУ

3.1 CRUD

Система CRUD операцій реалізована для всіх основних сутностей додатку. Головний контролер `index()` забезпечує відображення рейтингової таблиці з сортуванням за кількістю поваги та виконаних завдань:

```
@app.route('/')
def index():
    db = get_db()
    leaderboard_entries = db.execute('SELECT * FROM leaderboard ORDER BY respect DESC,
challenges_completed DESC').fetchall()
    leaderboard_data = []
    current_user_id = session.get('user_id')

    for entry in leaderboard_entries:
        entry_dict = dict(entry)
        entry_dict['rank'] = get_rank(entry_dict['challenges_completed'])
        if current_user_id:
            has_liked = db.execute(
                'SELECT 1 FROM likes WHERE liker_user_id = ? AND liked_leaderboard_id = ?',
                (current_user_id, entry_dict['id'])
            ).fetchone()
            entry_dict['has_liked'] = bool(has_liked)
        leaderboard_data.append(entry_dict)
```

3.2 Авторизація та реєстрація

Форма реєстрації `register.html` включає валідацію паролів та створення записів у двох таблицях одночасно:

```
<form method="post">
  <div>
    <label for="username">Username:</label>
    <input type="text" id="username" name="username" required>
  </div>
  <div>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required>
  </div>
</div>
```



```

        <label for="confirm_password">Confirm Password:</label>
        <input type="password" id="confirm_password" name="confirm_password" required>
    </div>
    <div>
        <label for="country">Country:</label>
        <input type="text" id="country" name="country" required>
    </div>
    <button type="submit">Register</button>
</form>

```

```

return render_template('index.html', leaderboard=leaderboard_data)

```

Контролер реєстрації включає валідацію та хешування паролів:

```

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        confirm_password = request.form['confirm_password']
        country = request.form['country']

        if password != confirm_password:
            flash('Passwords do not match. Please try again.', 'danger')
            return render_template('register.html')

        hashed_password = hash_password(password)
        db = get_db()

        try:
            cursor = db.execute("INSERT INTO users (username, password) VALUES (?, ?)",
                                (username, hashed_password))
            user_id = cursor.lastrowid

            db.execute("INSERT INTO leaderboard (username, country, challenges_completed,
            respect, user_id) VALUES (?, ?, ?, ?, ?)",
                        (username, country, 0, 0, user_id))
            db.commit()
            flash('Registration successful! Please log in.', 'success')
            return redirect(url_for('login'))
        except sqlite3.IntegrityError:
            flash('Username already exists. Please choose a different one.', 'danger')

```

Username already exists. Please choose a different one.

Register New Hacker

Username:

Password:

Confirm Password:

Country:

Register

Рисунок 1.4 – Верифікація існуючого користувача

Авторизація реалізована через перевірку хешованих паролів та створення сесії:

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        db = get_db()
        user = db.execute('SELECT * FROM users WHERE username = ?', (username,)).fetchone()
        if user and check_password(user['password'], password):
            session['logged_in'] = True
            session['user_id'] = user['id']
            session['username'] = user['username']
            session['is_admin'] = bool(user['is_admin'])
            flash('Logged in successfully!', 'success')
            return redirect(url_for('index'))
        else:
            flash('Invalid username or password.', 'danger')
```

Функціональність надання поваги реалізована через систему лайків з унікальними обмеженнями:

```
@app.route('/respect/<int:leaderboard_entry_id>', methods=['POST'])
def respect_entry(leaderboard_entry_id):
    if not session.get('logged_in'):
        flash('You need to be logged in to give respect.', 'danger')
        return redirect(url_for('login'))
    liker_user_id = session['user_id']
```

```

db = get_db()
existing_like = db.execute(
    'SELECT 1 FROM likes WHERE liker_user_id = ? AND liked_leaderboard_id = ?',
    (liker_user_id, leaderboard_entry_id)
).fetchone()
if existing_like:
    flash('You have already given respect to this entry.', 'warning')
else:
    db.execute('INSERT INTO likes (liker_user_id, liked_leaderboard_id) VALUES (?, ?)',
        (liker_user_id, leaderboard_entry_id))
    db.execute('UPDATE leaderboard SET respect = respect + 1 WHERE id = ?',
        (leaderboard_entry_id,))
    db.commit()
    flash('Respect given!', 'success')

@Repository
public interface EventRepository extends JpaRepository<Event, Long>
    Optional<Event> findById(Long id);
    @Query("SELECT e FROM Event e ORDER BY e.createdAt DESC")
    List<Event> findAll();
    List<Event> findAllByUser_Id(Long userId);
}

```

Respect given!					
Hacker Leaderboard					
Username	Country	Rank	Challenges Completed	Respect	Actions
▼	NUSA	Legend	55	1	👍
venator17	Ukraine	Script Kiddie	0	1	❤️
Bobby Quine	Sprawl	Pro Hacker	10	0	👍

Рисунок 1.5 – Система Поваги

4 РОБОТА З UI КОМПОНЕНТАМИ

Адміністративна панель включає повний набір CRUD операцій для управління користувачами та записами рейтингу. Захист адмін-панелі реалізовано через декоратор `@app.before_request`:

```
@app.before_request
def check_admin_privileges():
    admin_endpoints = [
        'admin_panel', 'admin_add_leaderboard', 'admin_edit_leaderboard',
        'admin_delete_leaderboard', 'admin_add_user', 'admin_edit_user', 'admin_delete_user'
    ]
    if request.endpoint and request.endpoint in admin_endpoints and not
    session.get('is_admin'):
        flash('Access denied. Admin privileges required.', 'danger')
        return redirect(url_for('index'))
```

Стилізація використовує кастомні CSS змінні для підтримки кібер-панк тематики:

```
:root {
    --pixel-font: 'Press Start 2P', monospace;
    --bg-black: #000000;
    --main-cyan: #00FFFF;
    --dark-cyan: #008080;
    --medium-grey: #444444;
    --dark-grey: #222222;
}

body {
    font-family: var(--pixel-font);
    background-color: var(--bg-black);
    color: var(--main-cyan);
    font-size: 0.85em;
    line-height: 1.6;
}
```

Таблиці мають спеціальне оформлення з неоновими кольорами та ефектами наведення:

```
table {
    width: 100%;
    border-collapse: collapse;
```

```

margin-top: 20px;
border: 2px solid var(--main-cyan);
overflow: hidden;
}

th {
background-color: var(--main-cyan);
font-weight: bold;
color: var(--bg-black);
padding: 12px 10px;
font-size: 0.9em;
}

tr:hover {
background-color: var(--medium-grey);
}

```

Кнопки поваги мають інтерактивну зміну стану з емодзі:

```

.respect-button {
background: none;
border: none;
color: var(--main-cyan);
cursor: pointer;
font-size: 1.8em;
padding: 0;
transition: color 0.2s ease;
}

.respect-button.respected {
color: #00FF00;
}

```

Інтеграція jQuery UI для роботи з датами та модальними вікнами:

```

$(function() {
$("#some_date").datepicker({
dateFormat: "yy-mm-dd"
});
});

function openModal(modalId) {
document.getElementById(modalId).style.display = "block";
}

function closeModal(modalId) {
document.getElementById(modalId).style.display = "none";
}

```

Admin Panel

Leaderboard Entries

Add New Leaderboard Entry

ID	Username	Country	Challenges	Respect	User ID	Actions
1	v	NUSA	55	1	2	<div>Edit</div> <div>Delete</div>
2	Bobby Quine	Sprawl	10	0	3	<div>Edit</div> <div>Delete</div>
3	venator17	Ukraine	0	1	4	<div>Edit</div> <div>Delete</div>

Users

Add New User

ID	Username	Is Admin	Actions
1	admin	Yes	<div>Edit</div> <div>Delete</div>

Рисунок 1.6 – Панель Адміністратора

Add Leaderboard Entry

Username:

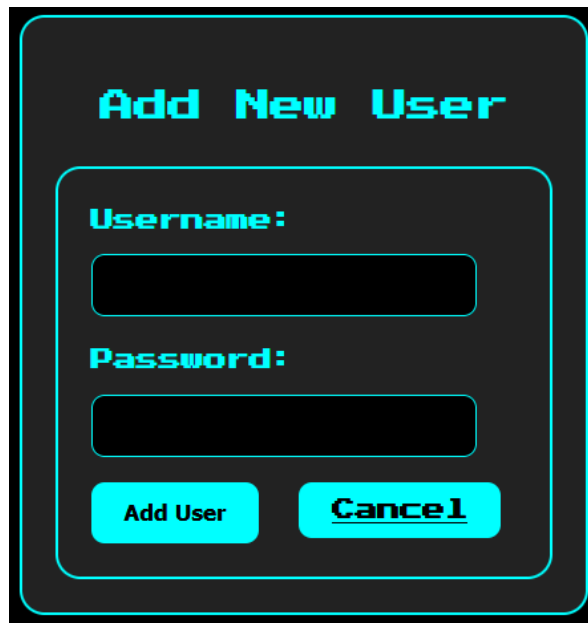
Country:

Challenges Completed:

Respect:

Add Entry

Рисунок 1.7 – Додання нової доски лідерів



The image shows a 'Add New User' dialog box with a dark gray background and rounded corners. At the top, the title 'Add New User' is displayed in a bold, black, monospace-style font. Below the title, there is a section with a thin gray border containing two input fields. The first field is labeled 'Username:' in a bold, black, monospace-style font. The second field is labeled 'Password:' in the same font style. Both fields are empty and have a light gray border. At the bottom of the dialog box, there are two buttons: 'Add User' and 'Cancel'. The 'Add User' button is light gray with black text, and the 'Cancel' button is light gray with black text and a red underline.

Add New User

Username:

Password:

Add User **Cancel**

Рисунок 1.8 – Додавання нового користувача

5 ПРЕДСТАВЛЕННЯ РОЗРОБЛЕНОГО ПРОЕКТУ

5.1 Опис розробленого проєкту

Розроблено повнофункціональний веб-додаток системи рейтингу хакерів "Hacker Leaderboard" з використанням Python Flask, SQLite, HTML/CSS/JavaScript та jQuery UI. Dodatok присвячений веденню рейтингу учасників хакерських змагань та CTF турнірів.

Система включає наступні сторінки: головна сторінка з рейтинговою таблицею, форми реєстрації та авторизації, сторінка редагування профілю, адміністративна панель з можливостями управління користувачами та записами рейтингу.

Головна функціональність включає систему ранжування на основі виконаних завдань (від "Script Kiddie" до "Legend"), можливість надавання поваги іншим учасникам, адміністративне управління всіма аспектами системи.

Дизайн виконано в стилі кібер-панк з використанням неонових кольорів (блакитний, зелений, червоний), пікселізованого шрифту "Press Start 2P" та тематичного оформлення елементів інтерфейсу.

База даних містить три взаємопов'язані таблиці з належними зовнішніми ключами та обмеженнями унікальності. Безпека забезпечується хешуванням паролів за допомогою SHA-256 та сесійною авторизацією.

5.2 Сторінки

Сторінка авторизації (рис. 1.1), сторінка реєстрації (рис. 1.2), головна сторінка (1.3), панель адміністратора (рис. 1.6).

The screenshot shows a web application interface with a dark theme. At the top, there is a navigation bar with three buttons: "Home", "Login", and "Register". The main content area features a central form titled "Register New Hacker". The form contains four input fields: "Username:", "Password:", "Confirm Password:", and "Country:". Below these fields is a "Register" button. The footer of the page displays the copyright notice "© 2025 Hacker Leaderboard".

Home Login Register

Register New Hacker

Username:

Password:

Confirm Password:

Country:

Register

© 2025 Hacker Leaderboard

Рисунок 1.1 – Вікно реєстрації

This screenshot shows the same "Register New Hacker" form as in Figure 1.1, but with sample data entered into the input fields. The "Username" field contains "venator17", the "Password" and "Confirm Password" fields contain "*****", and the "Country" field contains "Ukraine". The "Register" button remains visible at the bottom of the form. The navigation bar and footer are identical to the previous figure.

Home Login Register

Register New Hacker

Username:

Password:

Confirm Password:

Country:

Register

© 2025 Hacker Leaderboard

Рисунок 1.2 – Вікно авторизації



Рисунок 1.3 – Головна сторінка



Рисунок 1.6 – Панель адміністратора

Висновки: в результаті виконання даної роботи було розроблено повнофункціональний веб-додаток системи рейтингу хакерів з адміністративною панеллю, системою авторизації та інтерактивним користувацьким інтерфейсом у стилі кібер-панк.

Під час виконання роботи я навчився працювати з фреймворком Flask для створення веб-додатків, проектувати та реалізовувати реляційні бази даних SQLite, створювати системи авторизації з хешуванням паролів, розробляти адміністративні панелі з повним набором CRUD операцій, стилізувати веб-інтерфейси за

допомогою сучасних CSS технологій та інтегрувати JavaScript бібліотеки для покращення користувацького досвіду.