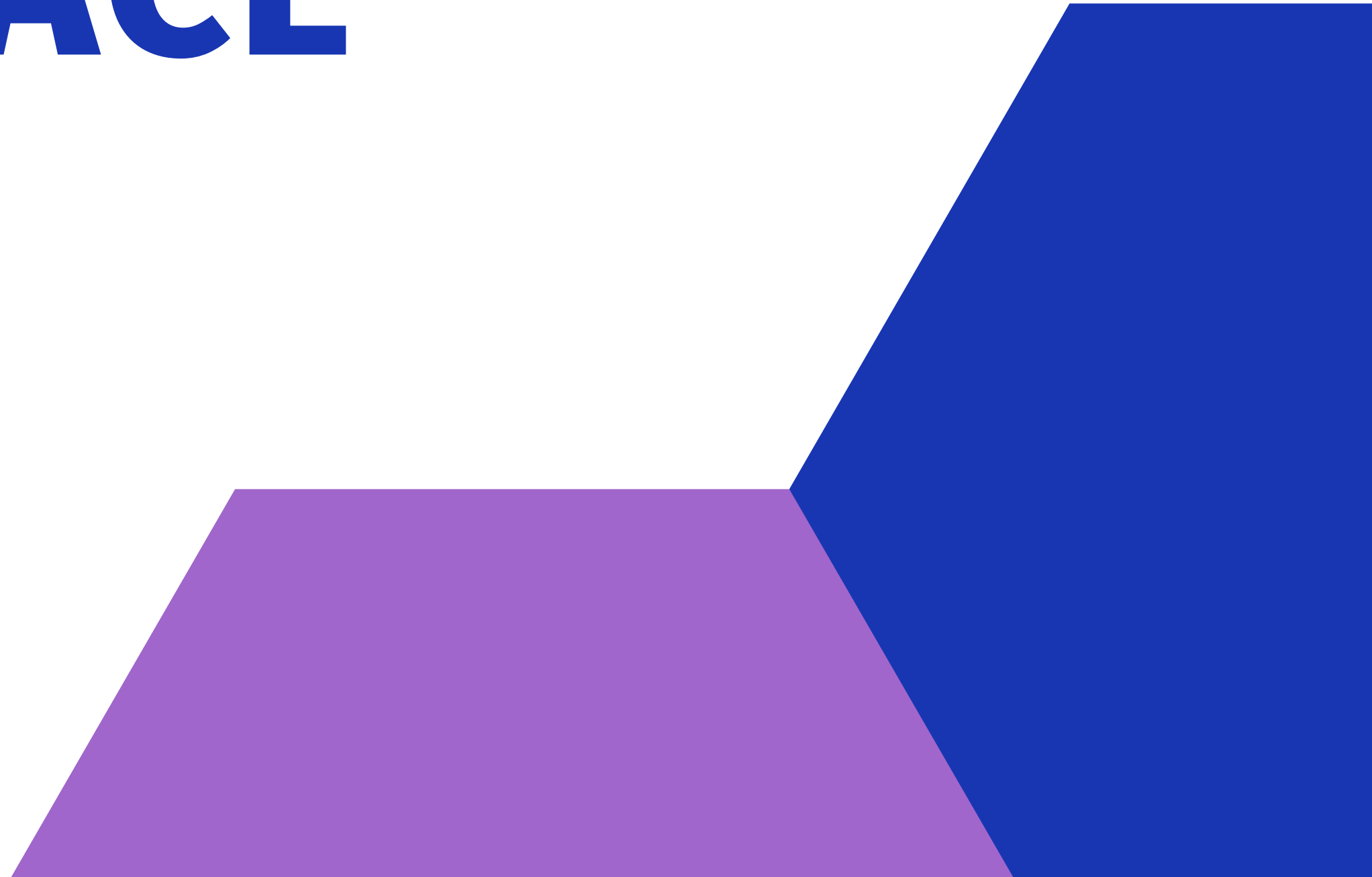


# FRIENDSPACE

Bolatbek Ermekov & Ilyas Abulov  
Group CS-2215



# Outline

- ◆ Main Goal
- ◆ Objectives of Project
- ◆ Future Possibilities
- ◆ Project Revelance
- ◆ Explaining of Code
- ◆ Advantages of Project

# Main Goal

The main goal of our project is to create a domestic social networking system. As a basis, we took such giants as Facebook and Twitter



# Project Revelance

It is impossible to deny the relevance of social networks, as they have recently gained enormous popularity. Every day millions of people use their services: some are looking for friends, clients, employers, others are just wasting their time.

Social networks allow people to stay in touch. You can fully communicate with your friend, exchange information, share news, interests, experience, while being in different cities or countries.

Social networks make it possible to make new acquaintances, find people with the same interests.

# Objectives of Project

01

Help people exchange information

02

Help people find new friends

03

Help people express themselves

04

Help people share their news, experiences



# Explaining of Code

Let's begin!



# User Registration

```
2 usages  ⚙ Bolatbekermekov
@app.route(rule: '/user/add', methods=['GET', 'POST'])
def add_user():
    name = None
    form = UserForm()
    our_users = Users.query.order_by(Users.date_added)
    hashed_pw = None

    if form.validate_on_submit():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        about_author = form.about_author.data

        if form.password_hash.data != form.password_hash2.data:
            flash(message: "Passwords do not match. Please try again.", category='error')
            return redirect(url_for('add_user'))
        else:
            # Hash the password
            hashed_pw = generate_password_hash(form.password_hash.data)

            user = Users.query.filter_by(email=email).first()
            if user:
                flash(message: "User with this email already exists.", category="error")
            else:
                new_user = Users(username=username, name=name, email=email, password_hash=hashed_pw, about_author=about_author)
```

FriendSpace Add Blog Post Posts Login Register Dashboard Search

## User List:

Name  
itachi

Username  
itachi uchiha

Email  
itachi.uchiha@bk.ru

About Author  
amaterasu

Password  
\*\*\*\*

Confirm Password

The `add_user()` function is used in web applications to create new user accounts. It takes user information as input and stores it in a database.

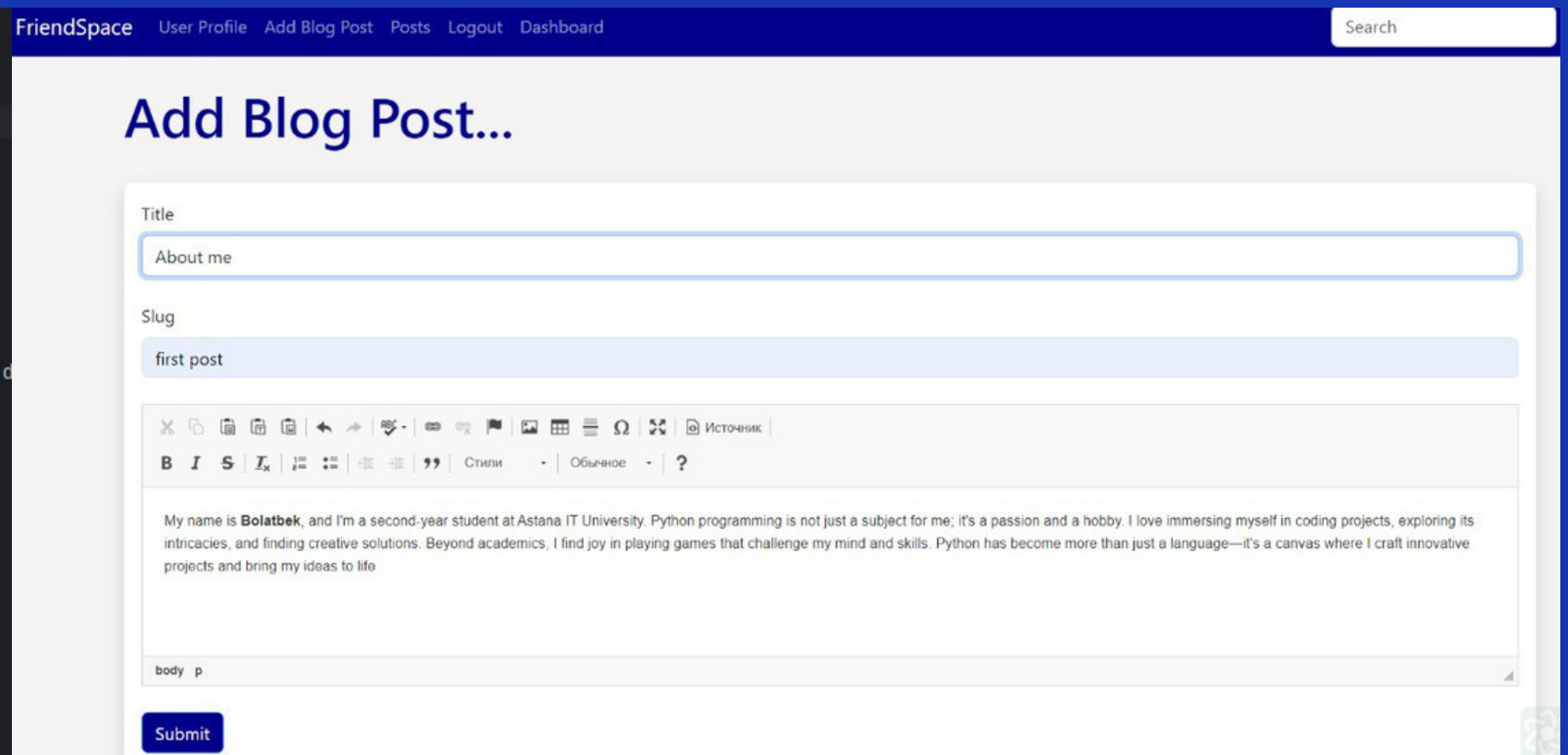
# Adding a Blog

```
#Add Post page
# Bolatbekermekov
@app.route(rule: '/add-post', methods=['GET', 'POST'])
def add_post():
    form = PostForm()
    post_users = Posts.query.order_by(Posts.date_posted)

    if form.validate_on_submit():
        poster = current_user.id
        post = Posts(title=form.title.data, content=form.content.data, poster_id=poster, slug=form.slug.data)
        # Clear the Form
        form.title.data = ''
        form.content.data = ''
        form.slug.data = ''

        db.session.add(post)
        db.session.commit()

        flash(message: "Blog Post Submitted Successfully!", category="success")
    return render_template(template_name_or_list: "add_post.html", form=form, post_users=post_users)
```

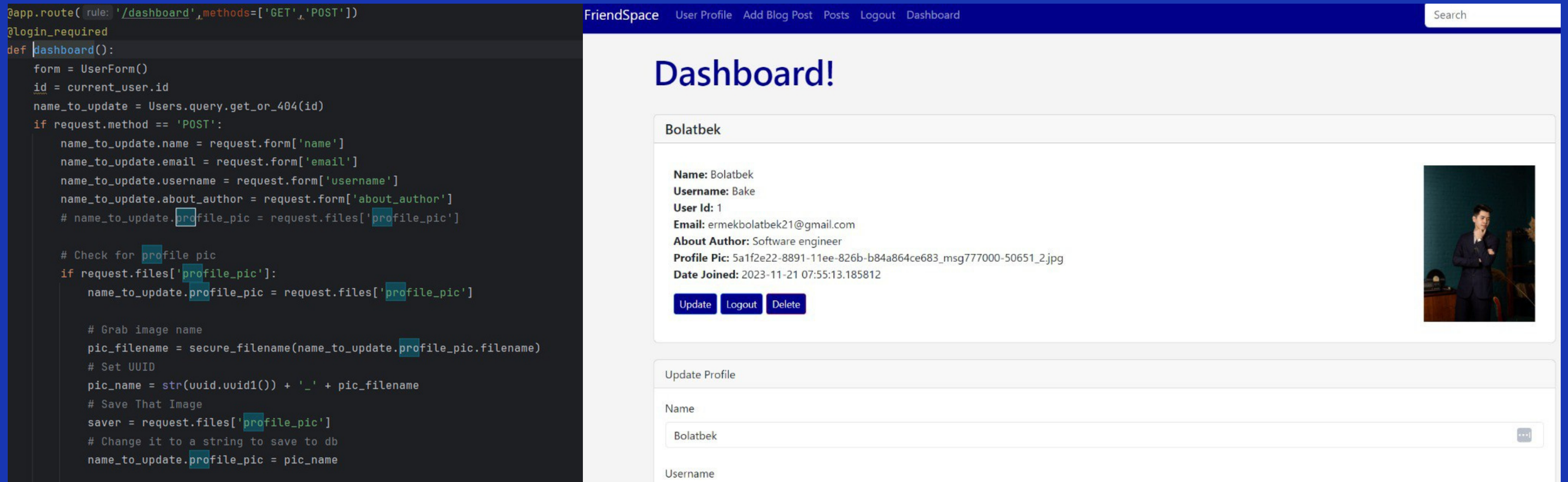


The screenshot shows a web application interface for adding a blog post. The browser's address bar shows the URL 'http://127.0.0.1:5000/add-post'. The page title is 'Add Blog Post...'. The form includes a 'Title' field with the text 'About me', a 'Slug' field with the text 'first post', and a rich text editor. The rich text editor contains the text: 'My name is **Bolatbek**, and I'm a second-year student at Astana IT University. Python programming is not just a subject for me; it's a passion and a hobby. I love immersing myself in coding projects, exploring its intricacies, and finding creative solutions. Beyond academics, I find joy in playing games that challenge my mind and skills. Python has become more than just a language—it's a canvas where I craft innovative projects and bring my ideas to life'. A 'Submit' button is located at the bottom of the form.

The `add_post()` function processes requests to the page for adding a post. It gets data from the post addition form provided on the page and uses it to create a new post in the database.



# Dashboard



The dashboard() function processes requests to the personal account page. It displays user data, including name, email address, username and bio. It also allows the user to edit their data.

# User Login

```
app.route(rule: '/Login', methods=['GET', 'POST'])
def login():
    form = LoginForm()
    if form.validate_on_submit():
        user = Users.query.filter_by(username=form.username.data).first()
        if user:
            #Check the hash
            if check_password_hash(user.password_hash, form.password.data):
                login_user(user)
                flash(message="Login Successfull!!!", category='success')
                return redirect(url_for('dashboard'))
            else:
                flash(message="Wrong Password - Try Again!", category='error')
        else:
            flash(message="That User Doesn't Exist! Try Again...", category='error')
    return render_template(template_name_or_list='login.html', form=form)
```

riendSpace Add Blog Post Posts Login Register Dashboard

## Login!

Username

Password

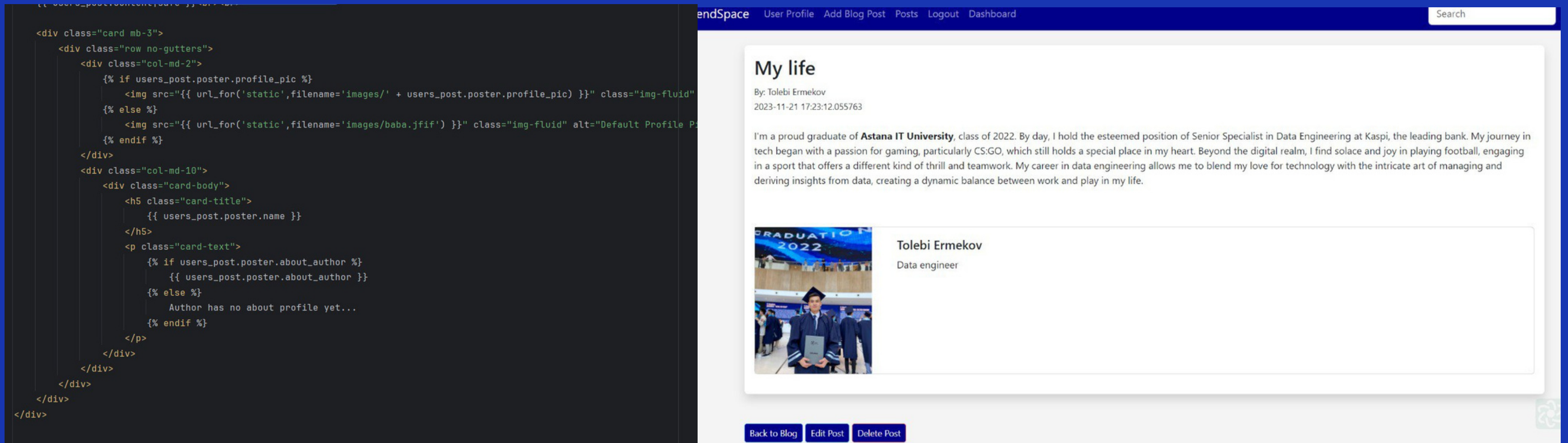
Submit

[Forgot Password?](#)

Need An Account? [Sign Up Now](#)

The login() function processes requests to the login page. It receives data from the login form provided on the page and uses it to search for the user in the database. If the user is found, his credentials are verified. If they are correct, the user logs in and is redirected to the personal account page.

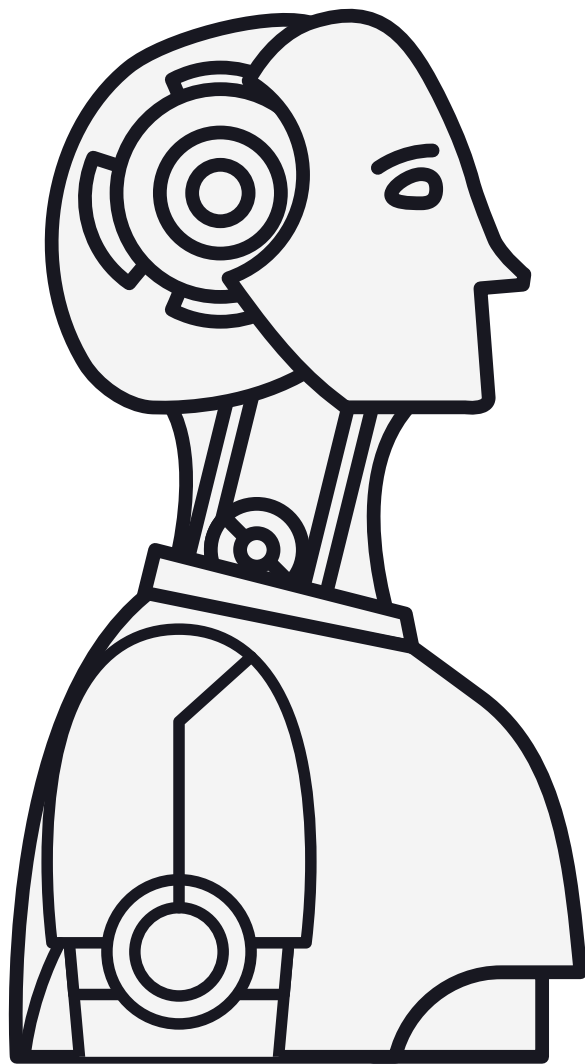
# Posts



This code displays a card with information about the author of the blog post. The `users_post.poster` object is accessed to get the author's name and profile picture. If the author has a profile picture, it is displayed. Otherwise, a default profile picture is displayed. The author's about section is also displayed if it is available.



# Future Possibilities



01

Video call  
conversations

02

Watching &  
Making videos

03

Live Broadcasting &  
Streaming

# Advantages of Project

Our code is a ready-made project for creating a blog with user management functions. It is easy to use and configure, and it can be used to create a blog of any size.

## **Ease to use**

The code is well documented and easy to use. Even beginner developers can easily figure it out and start creating their own blog.

## **Reliability**

The code has been tested and verified to ensure its reliability. It also uses secure methods to store user passwords.

## **Scalability**

The code can be easily scaled to add additional features or increase the number of users

# Usage of Project

If you want to create a blog, our code is a great option. It will help you quickly and easily create a functional and reliable blog.



## Personal blog

Our code can be used to create a personal blog where you can share your thoughts, ideas and experiences.



## Professional blog

Our code can be used to create a professional blog where you can promote your business or services



## Educational blog

Our code can be used to create an educational blog where you can share knowledge and information.

# Hope you enjoyed our presentation

Thank you for your attention!

