

Blog

Гаврилов Валерий
Цепилова Лена

Введение

Идея проекта заключается в создании социальной сети, в которой можно обмениваться информацией путём публикации новостей и комментирования их.

Описание реализации

В проекте использовались библиотеки flask, flask-login, flask-wtf, sqlalchemy, wtforms, datetime и os (для Heroku)

```
1 from flask import Flask, render_template, redirect, request, abort
2 from flask_login import LoginManager
3 from flask_login import login_user, logout_user, login_required, current_user
4
5 from forms.users_form import RegisterForm, LoginForm, RefactorForm
6 from forms.news_form import NewsForm
7 from forms.comment_form import CommentForm
8 from data.news_class import News
9 from data.users_class import User
10 from data.comment_class import Comment
11 from data import db_session
12
13 import os
14
15 from datetime import datetime
16
```

```
1 from flask_wtf import FlaskForm
2 from wtforms import PasswordField, StringField
3 from wtforms.fields.html5 import EmailField
4 from wtforms.validators import DataRequired
5
6
```

Blog entries

Add an item

Свеча горела

Стихотворение

Author - Юрий, Date - 2021-04-25 16:34:55.025367

Comments

Make a comment

Edit

Delete

Daily record

Today is 22nd of April, Thursday

Author - NewUser, Date - 2021-04-22 17:33:10.001857

Comments

User, 2021-04-23 21:49:58.702292

It was yesterday.

Make a comment

My first blog`s record

Hello, world!

Author - User, Date - 2021-04-22 16:44:04.334698

Comments

NewUser, 2021-04-23 17:47:24.868403

За главную страницу отвечают метод файла `main.py init()` и файлы `base.html` и `index.html`. В них определяется то, что зашёл ли пользователь в свой аккаунт, кнопки `register` и `sign up`, если нет и кнопка для перехода на профиль, если да, а также кнопки для создания новости или комментария и редактирования или удаления своих прошлых новостей (если пользователь зашел в аккаунт) и кнопка `MyBlog` для перехода на главную страницу. Также эти файлы отвечают за то, чтобы показывать все комментарии и новости, кроме приватных (если пользователь не автор публикации). Если новости меньше 2 дней и 2 часов, то она окрашена в зелёный цвет, другие же публикации в серый.

```

35 @app.route("/")
36 def index():
37     db_sess = db_session.create_session()
38     if current_user.is_authenticated:
39         news = db_sess.query(News).filter(
40             (News.user == current_user) | (News.is_private != True)).order_by(-News.id)
41     else:
42         news = db_sess.query(News).filter(News.is_private != True).order_by(-News.id)
43     comments = db_sess.query(Comment)
44     time_now = datetime.now().hour * 60 + datetime.now().minute
45     return render_template("index.html", title="Blog", news=news, comments=comments, time=time_now, datetime=datetime)
46

```

```

1 <!doctype html>
2 <html lang="en">
3 <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
6     <link rel="stylesheet"
7         href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
8         integrity="sha384-Vkoop8X4CGs03+Hhnxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
9         crossorigin="anonymous">
10     <title>{{title}}</title>
11 </head>
12 <body>
13 <header>
14     <nav class="navbar navbar-light bg-light">
15         <a class="navbar-brand" href="/">MyBlog</a>
16         {% if current_user.is_authenticated %}
17         <a class="navbar-brand" href="/profile">{{ current_user.name }}</a>
18         {% else %}
19         <p>
20             <a class="btn btn-primary" href="/register">Register</a>
21             <a class="btn btn-success" href="/login">Sign in</a>
22         </p>
23         {% endif %}
24     </nav>
25 </header>
26 <!-- Begin page content -->
27 <main role="main" class="container">
28     {% block content %}{% endblock %}
29 </main>
30 </body>
31 </html>

```

```

1 {% extends "base.html" %}
2
3 {% block content %}
4 <h1>Blog entries</h1>
5 {% if current_user.is_authenticated %}
6     <a href="/news" class="btn btn-primary">Add an item</a>
7 {% endif %}
8
9 {% for item in news %}
10     {% if (datetime.today() - item.created_date).days < 2 %}
11         <div class="col-md6 border rounded bg-success">
12
13     {% elif ((datetime.today() - item.created_date).days == 2 and
14             (time - item.created_date.hour * 60 + item.created_date.minute) > 120 %}
15         <div class="col-md6 border rounded bg-secondary">
16
17     {% else %}
18         <div class="col-md6 border rounded bg-secondary">
19     {% endif %}
20
21         <h2>{{item.title}}</h2>
22         <div>
23             {{item.content}}
24         </div>
25         <div>
26             Author - {{item.user.name}}, Date - {{item.created_date}}
27             <div class="col-md6 border rounded">
28                 <h5>Comments</h5>
29                 {% for elem in comments %}
30                     {% if item.id == elem.new_id %}
31                         <div class="col-md6 border">
32                             <p><font color="blue">{{ elem.user }}, {{elem.created_date}}</font></p>
33                             <p>{{ elem.content }}</p>
34                         </div>
35                     {% endif %}
36                 {% endfor %}
37             </div>
38             {% if current_user.is_authenticated %}
39             <div>
40                 <a href="/comment/{{ item.id }}" class="btn btn-info">
41                     Make a comment
42                 </a>
43             </div>
44             {% endif %}
45         </div>
46         {% if current_user.is_authenticated and current_user == item.user %}
47         <div>
48             <a href="/news/{{ item.id }}" class="btn btn-warning">
49                 Edit
50             </a>
51             <a href="/news_delete/{{ item.id }}" class="btn btn-danger">
52                 Delete
53             </a>
54         </div>
55         {% endif %}
56     </div>
57 </div>
58 </div>
59 {% endfor %}
60 {% endblock %}

```

За написание, редактирование и удаление новостей отвечают методы главного файла `add_news`, `news_delete` и `edit_news` и файлы `news_form.py`, `news_class.py` и `news.html`. В методы редактирования и удаления передаётся `id` новости, чтобы определить нужную новость

```
125 @app.route('/news', methods=['GET', 'POST'])
126 @login_required
127 def add_news():
128     form = NewsForm()
129     if form.validate_on_submit():
130         db_sess = db_session.create_session()
131         news = News()
132         news.title = form.title.data
133         news.content = form.content.data
134         news.is_private = form.is_private.data
135         current_user.news.append(news)
136         db_sess.merge(current_user)
137         db_sess.commit()
138         return redirect('/')
139     return render_template('news.html', title='Adding news',
140                             form=form)
```

```
194 @app.route('/news_delete/<int:id>', methods=['GET', 'POST'])
195 @login_required
196 def news_delete(id):
197     db_sess = db_session.create_session()
198     news = db_sess.query(News).filter(News.id == id,
199                                     News.user == current_user
200                                     ).first()
201     db_sess.query(Comment).filter(Comment.new_id == id).delete()
202     if news:
203         db_sess.delete(news)
204         db_sess.commit()
205     else:
206         abort(404)
207     return redirect('/')
208
```

```
160 @app.route('/news/<int:id>', methods=['GET', 'POST'])
161 @login_required
162 def edit_news(id):
163     form = NewsForm()
164     if request.method == "GET":
165         db_sess = db_session.create_session()
166         news = db_sess.query(News).filter(News.id == id,
167                                         News.user == current_user
168                                         ).first()
169     if news:
170         form.title.data = news.title
171         form.content.data = news.content
172         form.is_private.data = news.is_private
173     else:
174         abort(404)
175     if form.validate_on_submit():
176         db_sess = db_session.create_session()
177         news = db_sess.query(News).filter(News.id == id,
178                                         News.user == current_user
179                                         ).first()
180     if news:
181         news.title = form.title.data
182         news.content = form.content.data
183         news.is_private = form.is_private.data
184         db_sess.commit()
185         return redirect('/')
186     else:
187         abort(404)
188     return render_template('news.html',
189                             title='Editing news',
190                             form=form
191                             )
192
```

```

main.py x news_form.py x news_class.py x news.html x
1 from flask_wtf import FlaskForm
2 from wtforms import StringField, TextAreaField
3 from wtforms import BooleanField, SubmitField
4 from wtforms.validators import DataRequired
5
6
7 class NewsForm(FlaskForm):
8     title = StringField('Заголовок', validators=[DataRequired()])
9     content = TextAreaField("Содержание")
10    is_private = BooleanField("Личное")
11    submit = SubmitField('Применить')
12

```

```

main.py x news_form.py x news_class.py x news.html x
1 import datetime
2 import sqlalchemy
3 from sqlalchemy import orm
4
5 from .db_session import SqlAlchemyBase
6
7
8 class News(SqlAlchemyBase):
9     __tablename__ = 'news'
10
11    id = sqlalchemy.Column(sqlalchemy.Integer,
12                           primary_key=True, autoincrement=True)
13    title = sqlalchemy.Column(sqlalchemy.String, nullable=True)
14    content = sqlalchemy.Column(sqlalchemy.String, nullable=True)
15    created_date = sqlalchemy.Column(sqlalchemy.DateTime,
16                                     default=datetime.datetime.now)
17    is_private = sqlalchemy.Column(sqlalchemy.Boolean, default=True)
18
19    user_id = sqlalchemy.Column(sqlalchemy.Integer,
20                                sqlalchemy.ForeignKey("users.id"))
21    user = orm.relation('User')
22

```

```

main.py x news_form.py x news_class.py x news.html x
1 {% extends "base.html" %}
2
3 {% block content %}
4     <h1>Adding news</h1>
5     <form action="" method="post">
6         {{ form.hidden_tag() }}
7         <p>
8             {{ form.title.label }}<br>
9             {{ form.title(class="form-control") }}<br>
10            {% for error in form.title.errors %}
11                <p class="alert alert-danger" role="alert">
12                    {{ error }}
13                </p>
14            {% endfor %}
15        </p>
16        <p>
17            {{ form.content.label }}<br>
18            {{ form.content(class="form-control") }}<br>
19            {% for error in form.content.errors %}
20                <p content="alert alert-danger" role="alert">
21                    {{ error }}
22                </p>
23            {% endfor %}
24        </p>
25        <p>{{ form.is_private() }} {{ form.is_private.label }}</p>
26        <p>{{ form.submit(type="submit", class="btn btn-primary") }}</p>
27        {{message}}
28    </form>
29    {% endblock %}

```


Adding news

Заголовок

Свеча горела

Содержание

Стихотворение

☐ Личное

Применить

Добавление/редактирование
новости (одинаково выглядят)

Blog entries

Add an item

Новость

Текст

Author - Юрий, Date - 2021-04-26 17:10:37.884964

Comments

Make a comment

Edit Delete

Свеча горела

Стихотворение

Author - Юрий, Date - 2021-04-26 17:10:25.720983

Comments

Make a comment

Edit Delete

Blog entries

Add an item

Новость

Текст

Author - Юрий, Date - 2021-04-26 17:10:37.884964

Comments

Make a comment

Edit Delete

Daily record

Удаление новости “Свеча горела”

За написание комментариев отвечают метод главного файла `add_comment` и файлы `comment_class.py`, `comment_form.py` и `comment.html`. При вызове метода добавления комментариев передаётся `id` новости, чтобы поставить комментарий под нужной новостью

```
143 @app.route('/comment/<int:id>', methods=['GET', 'POST'])
144 @login_required
145 def add_comment(id):
146     form = CommentForm()
147     if form.validate_on_submit():
148         db_sess = db_session.create_session()
149         comment = Comment()
150         comment.content = form.content.data
151         comment.user = current_user.name
152         comment.new_id = id
153         db_sess.add(comment)
154         db_sess.commit()
155         return redirect('/')
156     return render_template('comment.html', title='Adding comments',
157                             form=form)
```

```
main.py x comment_class.py x comment_form.py x comment.html x
1 from flask_wtf import FlaskForm
2 from wtforms import StringField, TextAreaField
3 from wtforms import BooleanField, SubmitField
4 from wtforms.validators import DataRequired
5
6
7 class CommentForm(FlaskForm):
8     content = TextAreaField("Комментарий")
9     submit = SubmitField("Отправить")
10
```

```
main.py x comment_class.py x comment_form.py x comment.html x
1 import datetime
2 import sqlalchemy
3 from sqlalchemy import orm
4
5 from .db_session import SQLAlchemyBase
6
7
8 class Comment(SQLAlchemyBase):
9     __tablename__ = 'comments'
10
11     id = sqlalchemy.Column(sqlalchemy.Integer,
12                           primary_key=True, autoincrement=True)
13     content = sqlalchemy.Column(sqlalchemy.String, nullable=True)
14     created_date = sqlalchemy.Column(sqlalchemy.DateTime,
15                                     default=datetime.datetime.now)
16     user = sqlalchemy.Column(sqlalchemy.String)
17     new_id = sqlalchemy.Column(sqlalchemy.Integer)
```

```
main.py x comment_class.py x comment_form.py x comment.html x
1 {% extends "base.html" %}
2
3 {% block content %}
4     <form action="" method="post">
5         {{ form.hidden_tag() }}
6         <p>
7             {{ form.content.label }}<br>
8             {{ form.content(class="form-control") }}<br>
9             {% for error in form.content.errors %}
10                 <p content="alert alert-danger" role="alert">
11                     {{ error }}
12                 </p>
13             {% endfor %}
14         </p>
15         <p>{{ form.submit(type="submit", class="btn btn-primary") }}</p>
16         {{ message }}
17     </form>
18 {% endblock %}
```

Комментарий

I like it!

Отправить

За регистрацию аккаунта, вход в него и редактирование аккаунта отвечают функции главного файла register (регистрация), login (вход), profile (страница профиля), edition (редактирование) и logout (выход) и файлы users_form.py, users_class.py, register.html, login.html, profile.html и edition.html

```
48 @app.route('/register', methods=['GET', 'POST'])
49 def register():
50     form = RegisterForm()
51     if form.validate_on_submit():
52         if form.password.data != form.password_again.data:
53             return render_template('register.html', title='Register',
54                                   form=form,
55                                   message="Пароли не совпадают")
56         db_sess = db_session.create_session()
57         if db_sess.query(User).filter(User.email == form.email.data).first():
58             return render_template('register.html', title='Регистрация',
59                                   form=form,
60                                   message="Такой пользователь уже есть")
61         user = User(
62             name=form.name.data,
63             email=form.email.data,
64             about=form.about.data
65         )
66         user.set_password(form.password.data)
67         db_sess.add(user)
68         db_sess.commit()
69         return redirect('/login')
70     return render_template('register.html', title='Register', form=form)
```

```
73 @app.route('/login', methods=['GET', 'POST'])
74 def login():
75     form = LoginForm()
76     if form.validate_on_submit():
77         db_sess = db_session.create_session()
78         user = db_sess.query(User).filter(User.email == form.email.data).first()
79         if user and user.check_password(form.password.data):
80             login_user(user, remember=form.remember_me.data)
81             return redirect("/")
82         return render_template('login.html',
83                               message="Неправильный логин или пароль",
84                               form=form)
85     return render_template('login.html', title='Authorization', form=form)
86
87
88 @app.route('/profile')
89 @login_required
90 def profile():
91     return render_template('profile.html', title="Profile")
92
```

```

94 @app.route('/edition', methods=['GET', 'POST'])
95 @login_required
96 def edition():
97     form = RefactorForm()
98     if request.method == "GET":
99         db_sess = db_session.create_session()
100         user = db_sess.query(User).filter(User.id == current_user.id).first()
101         if user:
102             form.name.data = user.name
103             form.about.data = user.about
104         else:
105             abort(404)
106
107     if form.validate_on_submit():
108         db_sess = db_session.create_session()
109         user = db_sess.query(User).filter(User.id == current_user.id).first()
110         user.name = form.name.data
111         user.about = form.about.data
112         db_sess.commit()
113         return redirect('/profile')
114
115     return render_template('edition.html', title='Refactoring', form=form)
116
117
118 @app.route('/logout')
119 @login_required
120 def logout():
121     logout_user()
122     return redirect("/")

```

```

1 from flask_wtf import FlaskForm
2 from wtforms import PasswordField, StringField, TextAreaField, SubmitField, BooleanField
3 from wtforms.fields.html5 import EmailField
4 from wtforms.validators import DataRequired
5
6
7 class RegisterForm(FlaskForm):
8     email = EmailField('Почта', validators=[DataRequired()])
9     password = PasswordField('Пароль', validators=[DataRequired()])
10    password_again = PasswordField('Повторите пароль', validators=[DataRequired()])
11    name = StringField('Имя пользователя', validators=[DataRequired()])
12    about = TextAreaField("Немного о себе")
13    submit = SubmitField('Войти')
14
15
16 class LoginForm(FlaskForm):
17     email = EmailField('Почта', validators=[DataRequired()])
18     password = PasswordField('Пароль', validators=[DataRequired()])
19     remember_me = BooleanField('Запомнить меня')
20     submit = SubmitField('Войти')
21
22
23 class RefactorForm(FlaskForm):
24     name = StringField('Имя пользователя', validators=[DataRequired()])
25     about = TextAreaField("Немного о себе")
26     confirm = SubmitField('Confirm')
27

```


Register

Почта

1@yandex.ru

Пароль

.....

Повторите пароль

.....

Имя пользователя

Valery

Немного о себе

Ученик ЯЛ

Войти

Authorization

Почта

1@yandex.ru

Пароль

.....

☐ Запомнить меня

Войти

My profile

Name

Valery

Email

1@yandex.ru

About

Ученик ЯЛ

Created date

2021-04-26 17:13:33.247101

Editing Sign out

Refactor profile

Name

Имя пользователя

Valera

About

Немного о себе

Выпускник ЯЛ

Confirm Cancel

Вывод

В данном проекте мы сделали простую социальную сеть.

В дальнейшем можно реализовать такие вещи, как просмотр профилей других пользователей, добавление аватара в профиле, улучшение комментариев (функции редактирования и удаления), а также личные сообщения между пользователями.

Спасибо за внимание!