МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Дополнительное образование «Основы промышленного программирования»

PyGame проект по теме

«Википедия»

Ученики Полухин Сергей., Прядиев Роман, Половинкин Андрей

Методист Копытина А. Е.

Воронеж 2025

Содержание

[1 Название проекта 3](#_1t3h5sf)

[2 Авторы проекта 3](#_4d34og8)

[3 Описание идеи 3](#_2s8eyo1)

[4 Описание реализации 4](#_17dp8vu)

[5 Описание технологий 8](#_3rdcrjn)

[6 Интерфейс сайта «Википедия» 10](#_26in1rg)

## Название проекта

Требуется выполнить проект на Flask под названием «Википедия», который обладает следующими функциональными возможностями:

* Разработка сайта и его функционала.
* Создание html форм.
* Разметка с помощью bootstrap.
* Создание и работа с базой данных.
* Использование чужого API и создание своего.

## Авторы проекта

Авторами проекта является Полухин Сергей, Половинкин Андрей, Прядиев Роман которые самостоятельно выполняли вышеописанный проект.

## Описание идеи

Платформа для публикации, перевода и обсуждения научных статей, ориентированная на популяризацию науки и вовлечение широкой аудитории в академическое сообщество.

**Актуальность проекта:** В условиях стремительного роста научной информации и труднодоступности оригинальных публикаций для широкой публики, особенно на английском языке, платформа решает сразу несколько актуальных задач:

**1. Доступ к науке:** Сервис автоматически импортирует и переводит научные статьи, облегчая понимание сложных тем русскоязычным пользователям. Это снижает языковой барьер и делает науку доступной для студентов, преподавателей и просто заинтересованных читателей.

**2. Пользовательское участие и комьюнити:** Каждый пользователь может регистрироваться, публиковать свои статьи, комментировать и обсуждать материалы других. Это формирует активное научное сообщество, мотивирует авторов делиться знаниями и получать обратную связь.

**3. Современные цифровые решения:** Использование технологий вроде автоматического перевода, интеллектуального поиска (на базе rapidfuzz), генерации обложек статей с помощью API Unsplash — все это делает интерфейс современным и удобным для пользователя. Платформа легко расширяется за счет API и гибкой архитектуры.

**4. Простота и минимализм:** Простой дизайн, наличие только необходимых функций (регистрация, просмотр, публикация, поиск и комментарии) делают сайт интуитивно простым. Это важно для быстрого вовлечения новых пользователей и повышения времени их взаимодействия с сервисом.

**5. Образовательная ценность:** Проект может стать основой для образовательных инициатив: платформой для школьников и студентов, где они могут учиться анализировать научные тексты, задавать вопросы и делиться мнениями в безопасной и мотивирующей среде.

## Описание реализации

Сайт «Википедия» был разработан с помощью следующих функций:

def create\_app():

*Создает и настраивает Flask-приложение с нужными расширениями*

def get\_img(query):

*'''Запрашивает случайное изображение с Unsplash по ключевому слову (заголовок статьи)'''*

class Category(db.Model):

*'''Модель для категорий, к которым могут принадлежать страницы'''*

class User(UserMixin, db.Model):

*'''Модель пользователя с базовыми полями: логин, email, пароль и связанные страницы'''*

class Page(db.Model):

*'''Модель страницы научной статьи'''*

class Tag(db.Model):

*'''Модель тегов страниц'''*

class Comment(db.Model):

*'''Модель для создания обсуждений под научными статьями'''*

def api\_pages():

*'''Возвращает список страниц в формате JSON'''*

* def load\_user(user\_id):
* *'''Загружает пользователя по ID для Flask-Login'''*
* def show\_category(category\_id):
* *'''Отображает все страницы, принадлежащие выбранной категории'''*

* def home():
* *'''Домашняя страница'''*

* def profile(username):
* *'''Для показа профиля пользователя'''*

def register():

*'''Регистрирует нового пользователя'''*

def random\_page():

*'''Перенаправляет пользователя на рандомно выбранную статью'''*

def api\_search():

*'''API-поиск страниц'''*

def login():

*'''Регистрация пользователя: ник, почта и пароль'''*

def logout():

*'''Выход пользователя из системы'''*

def create\_page():

*'''Для создания страниц авторизованным пользователем'''*

def search():

*'''Поиск страниц по приблизительному совпадению с помощью библиотеки rapidfuzz'''*

def page\_detail(page\_id):

*'''Отображает содержимое страницы и обсуждения под статьями'''*

def edit\_page(page\_id):

*'''Позволяет автору отредактировать свою страницу'''*

def create\_first\_user():

*'''Создает первого пользователя'''*

def import\_articles\_from\_arxiv(query, max\_results):

*'''Импортирует статьи из архив.орг, переводит и сохраняет как страницы'''*

if \_\_name\_\_ == '\_\_main\_\_':

'''Запуск приложения, создание первого пользователя, импорт статей'''

## Описание технологий

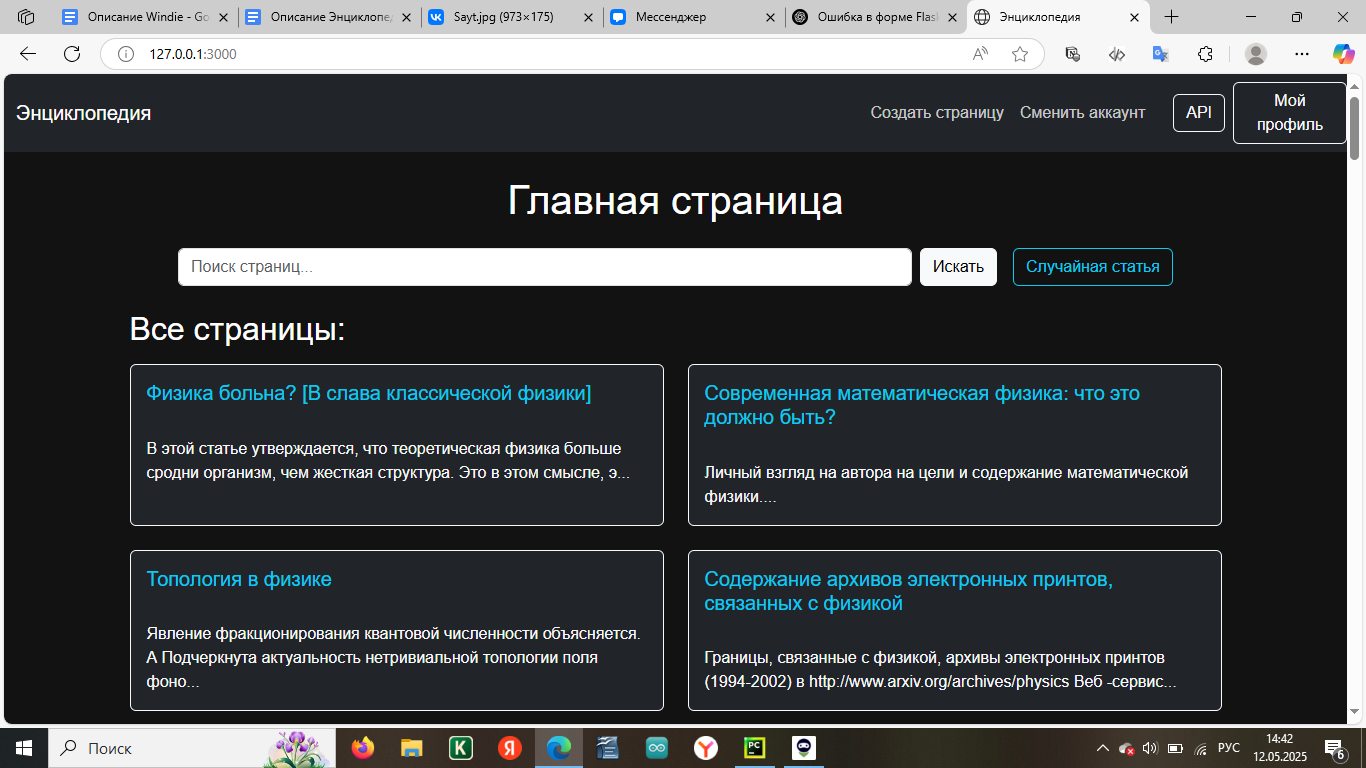
В разработанной системе «Windie» реализованы все рассмотренные в ходе изучения блока PyGame технологии:

* Создание и использование ORM модели.
* Создание html шаблонов (Основная страница сайта, страницы пользователя, статей и тд.).
* bootstrap разметка (Применена во всех html шаблонах).
* Регистрация и авторизация пользователей (Вход, регистрация и смена аккаунта на сайте).
* Создание своего API
* Использование стороннего API (API архив.орг для выгрузки статей и API unsplash для создания к ним картинок)
* Работа с бд (Хранение статей, их авторов, времени написания и тд. в созданной базе данных)
* Хостинг (Развернуто с помощью tuna)

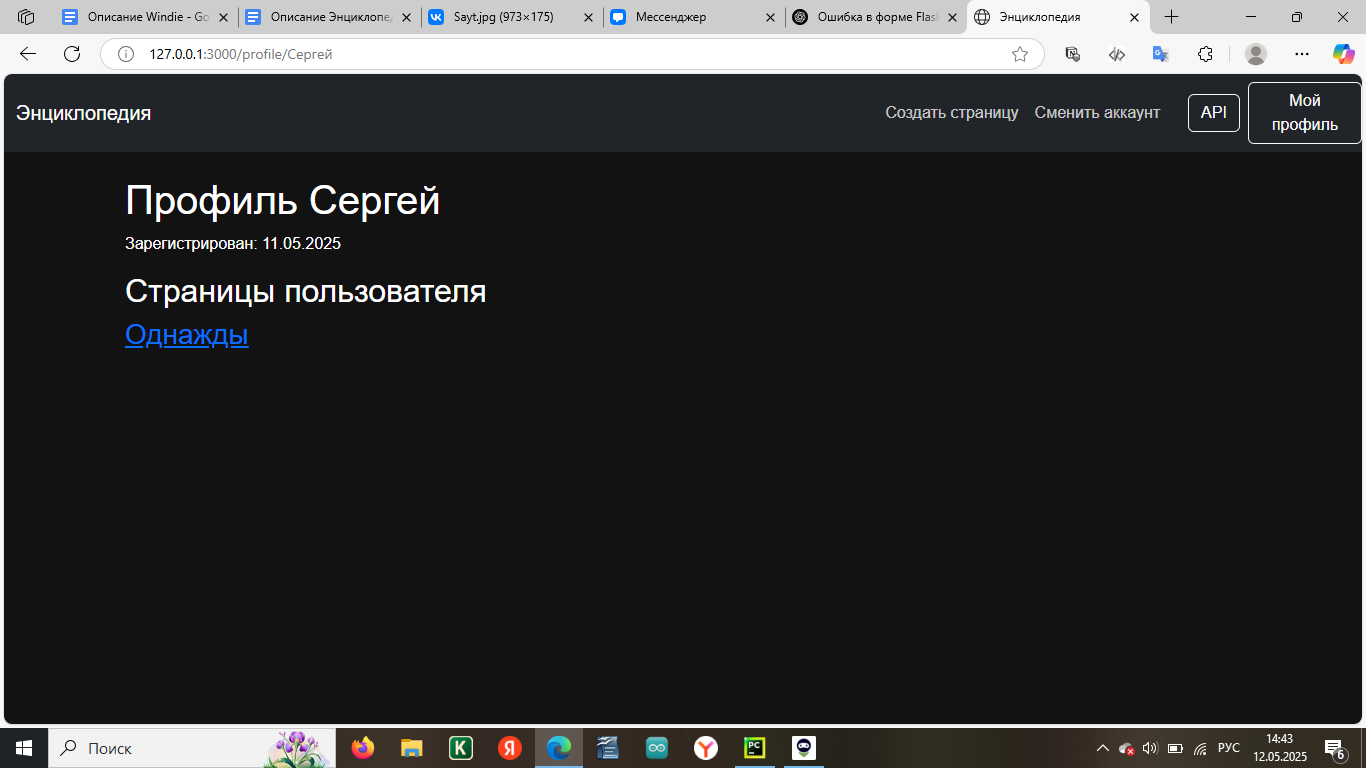
Необходимые для запуска библиотеки:

указаны в requirements.txt

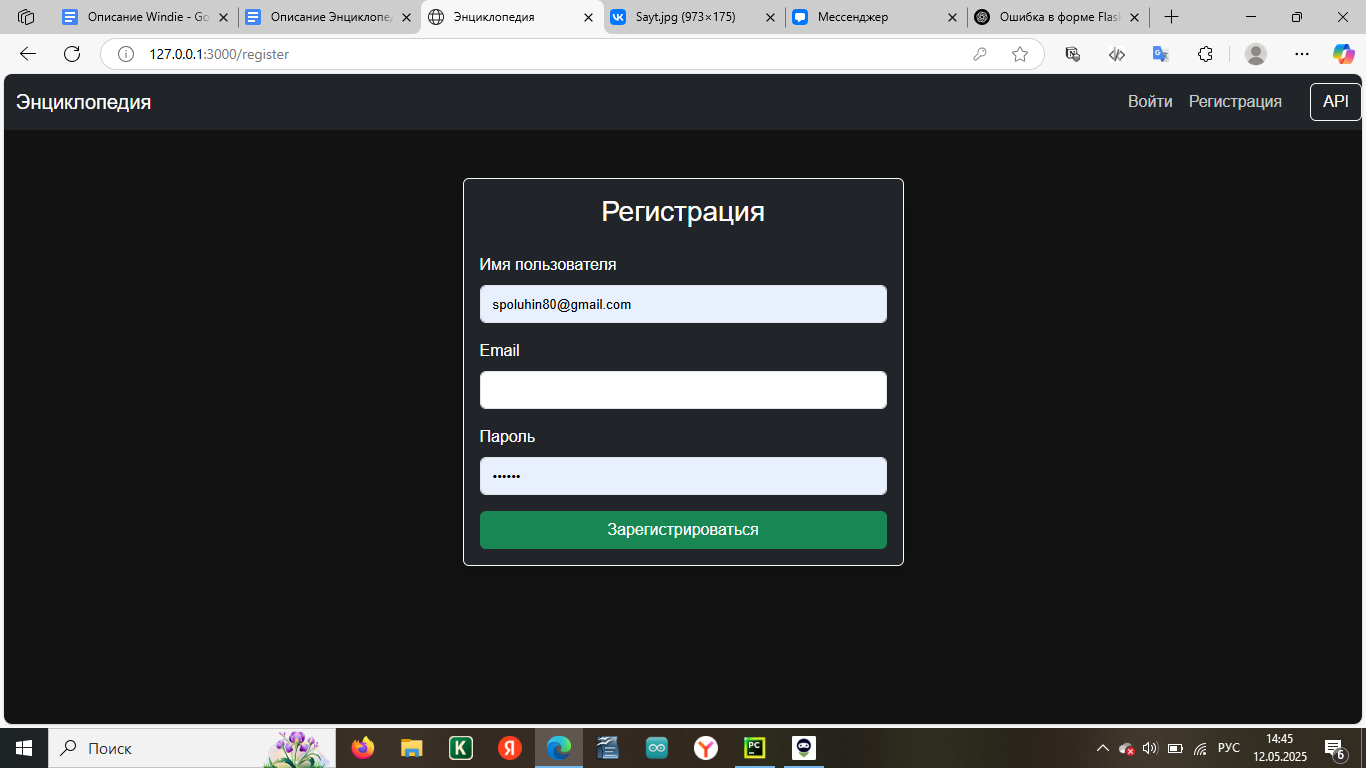
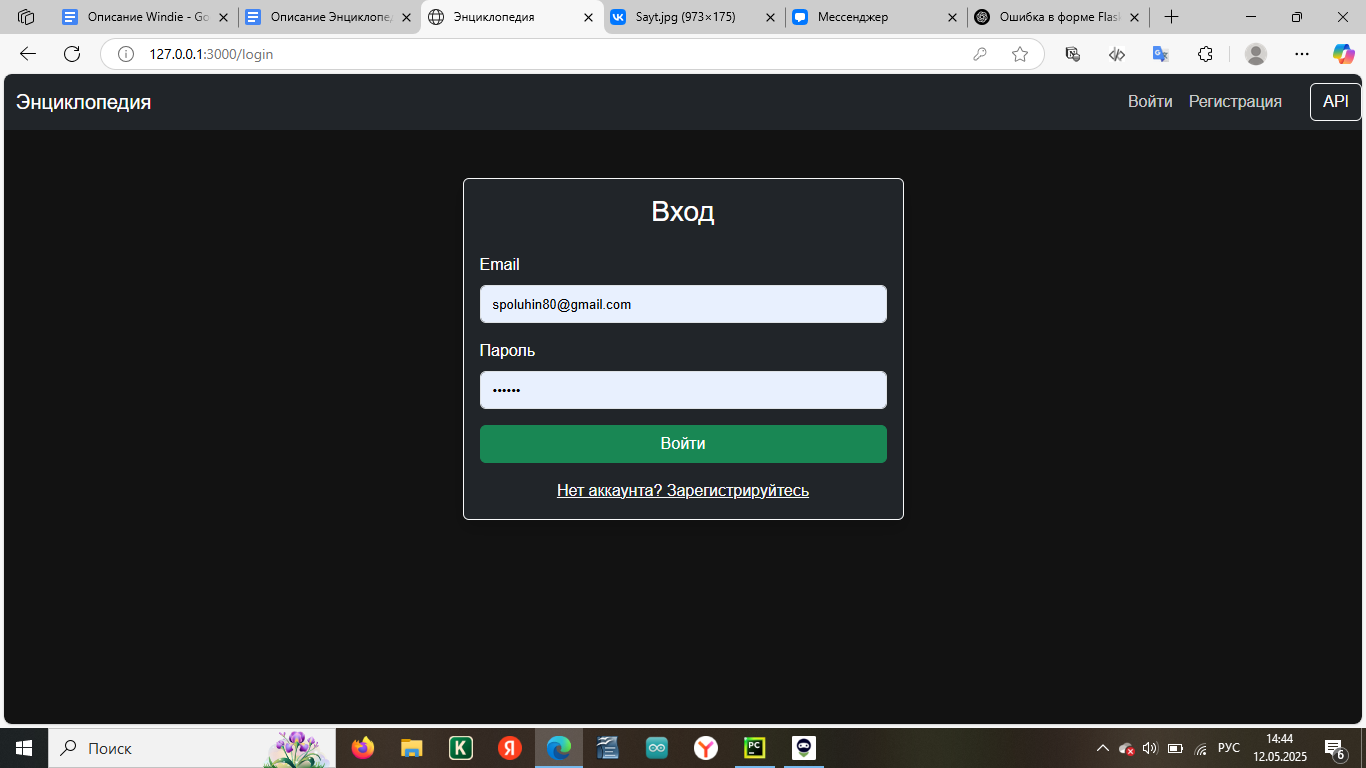
## Интерфейс сайта «Википедия»

****

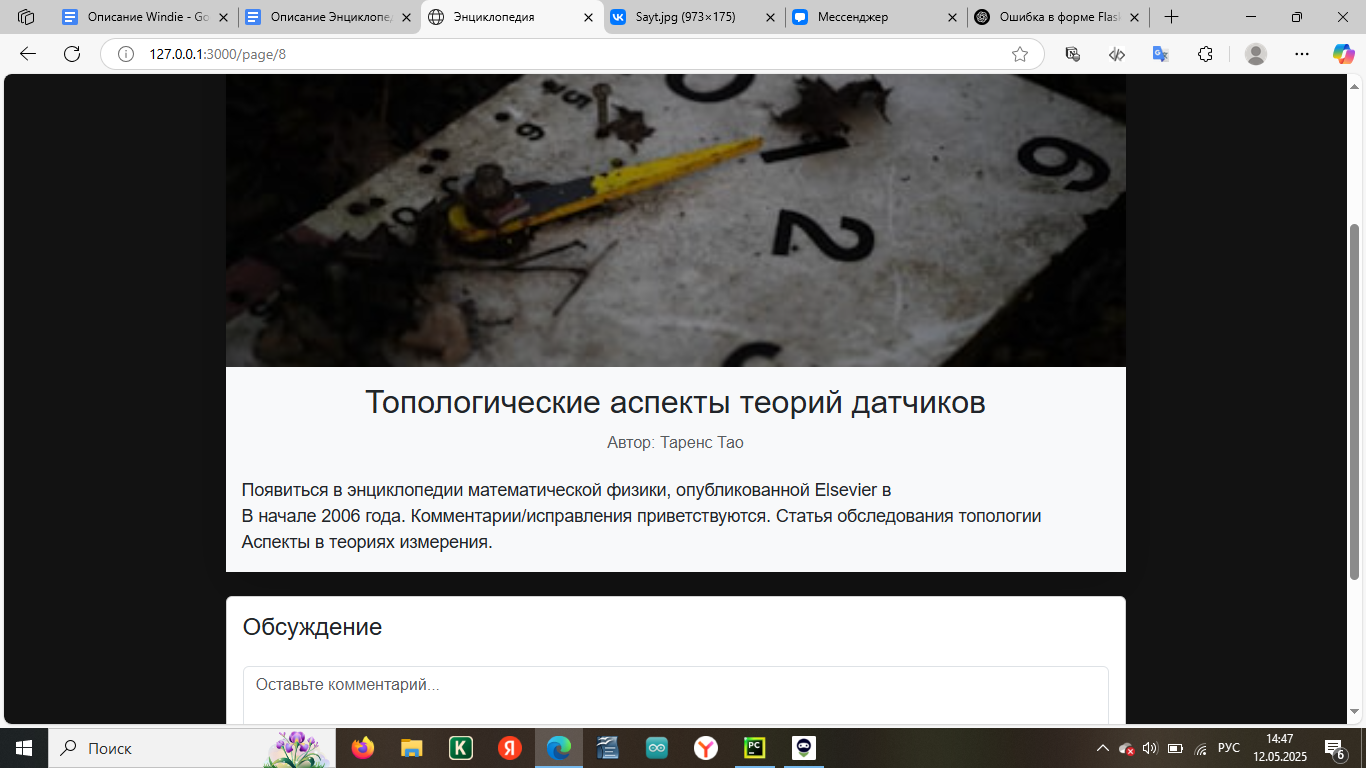
1. Общий вид сайта



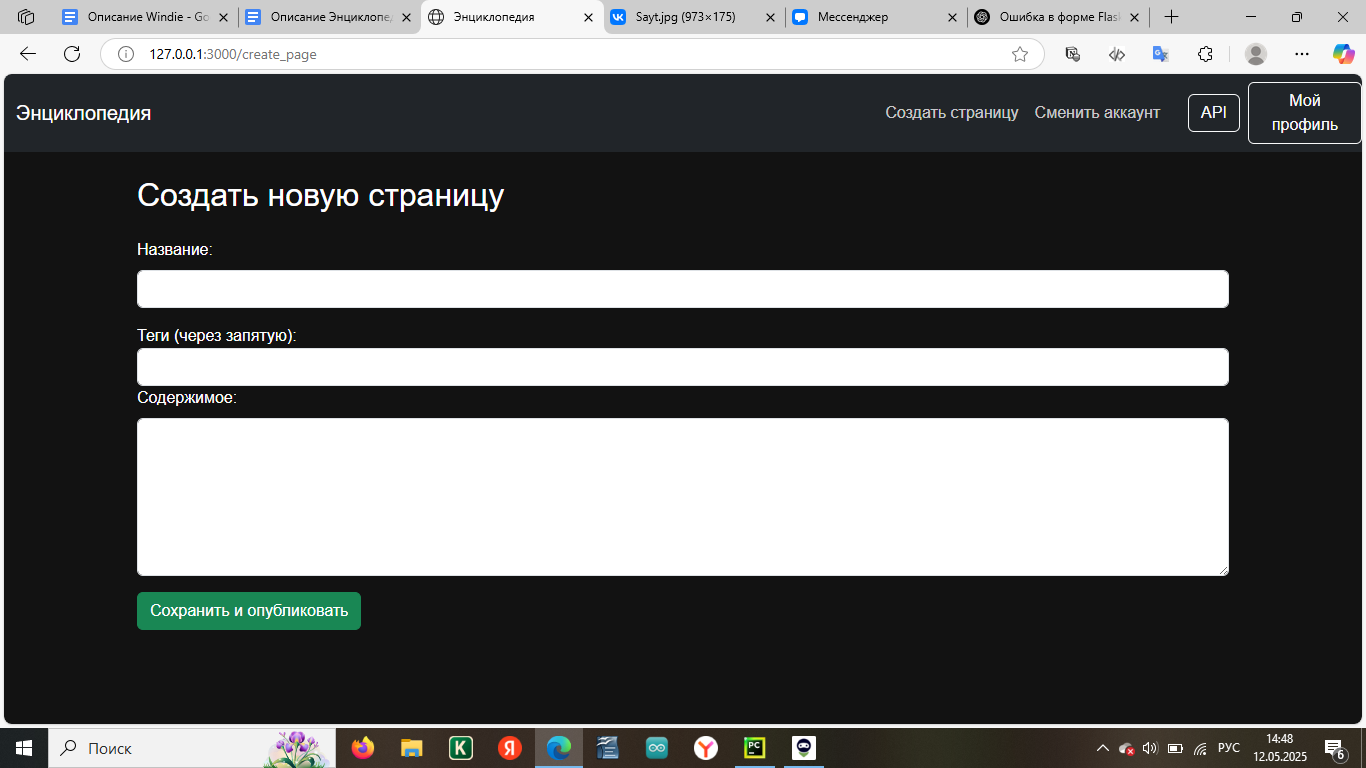
1. Личный аккаунт пользователя



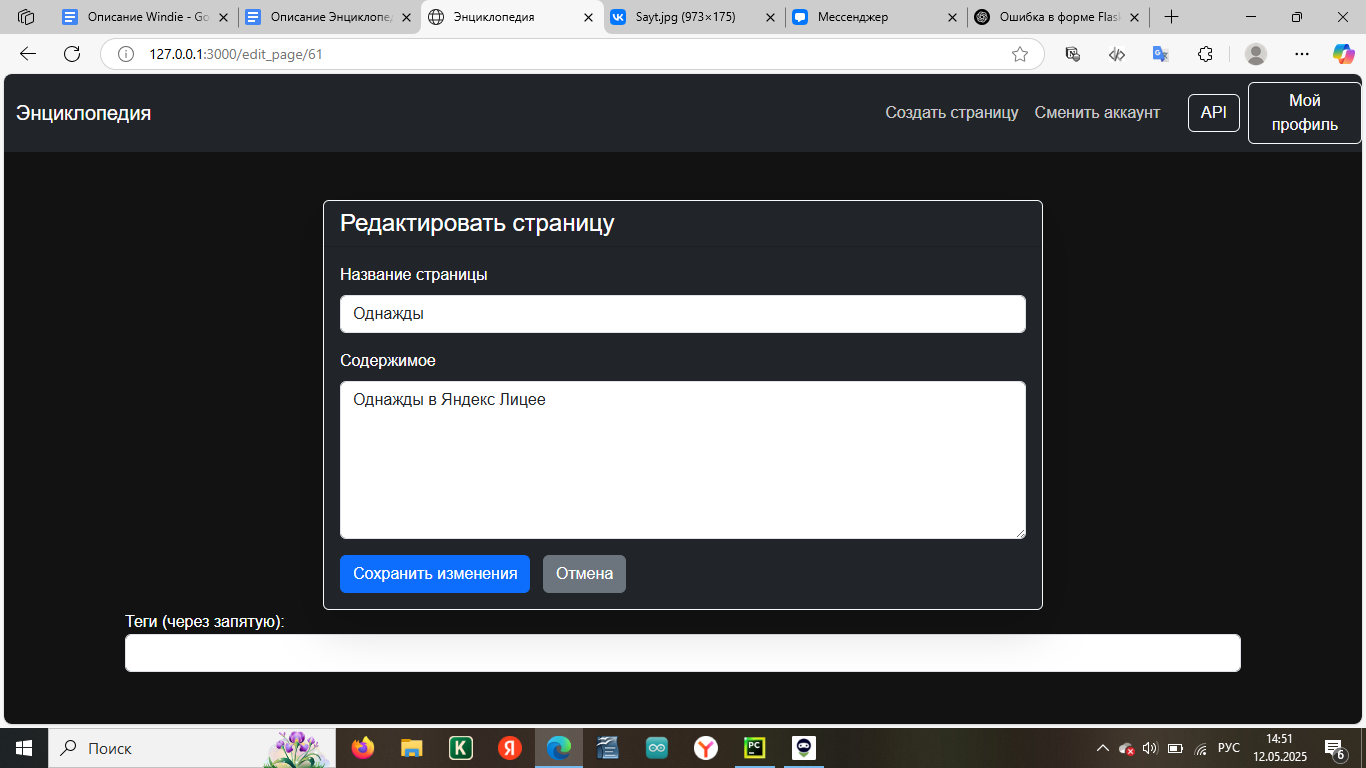
1. Формы входа и регистрации



1. Пример научной статьи



1. Написание новой статьи



1. Редактирование написанной тобой статьи