

**Московский государственный технический
университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по домашнему заданию

Выполнил:

студент группы ИУ5-54Б

Киреев А.А.

Подпись и дата:

Москва, 2021 г.

Цель домашнего задания: изучение возможностей создания прототипа веб-приложения на основе базы данных с использованием фреймворка Django.

Задание:

В качестве домашнего задания предлагается выполнить проект «Вопросы и Ответы». Этот сервис позволит пользователям Интернета задавать вопросы и получать на них ответы. Возможности комментирования и голосования формируют сообщество и позволяет пользователям активно помогать другим. В качестве образца реализации рекомендуется использовать Stack Overflow.

Основные сущности:

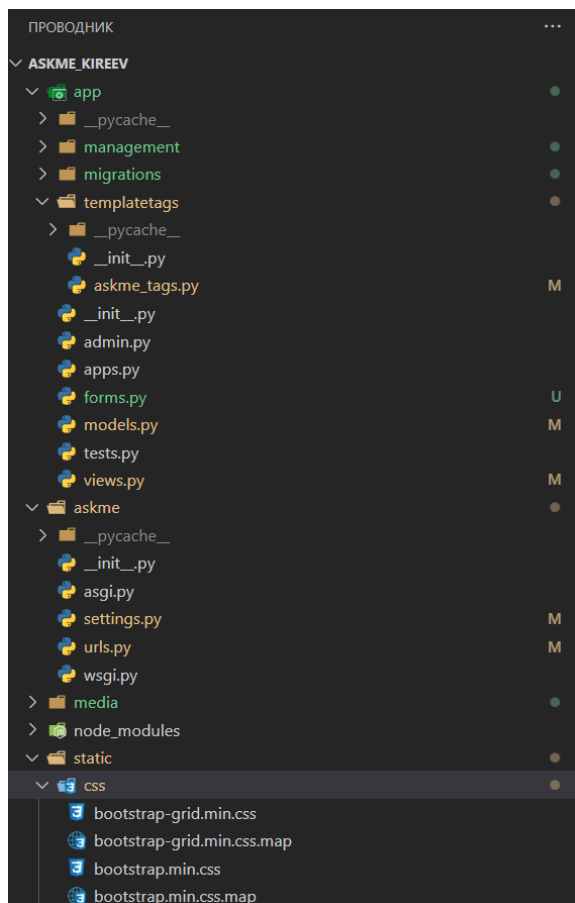
Пользователь – электронная почта, никнейм, пароль, аватарка, дата регистрации, рейтинг.

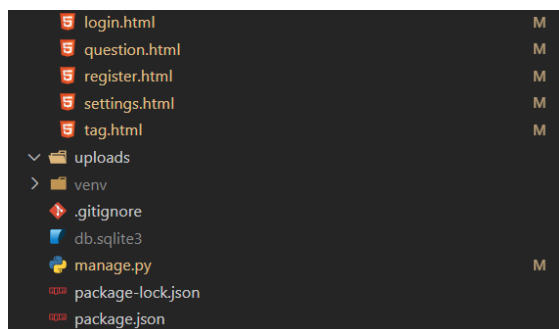
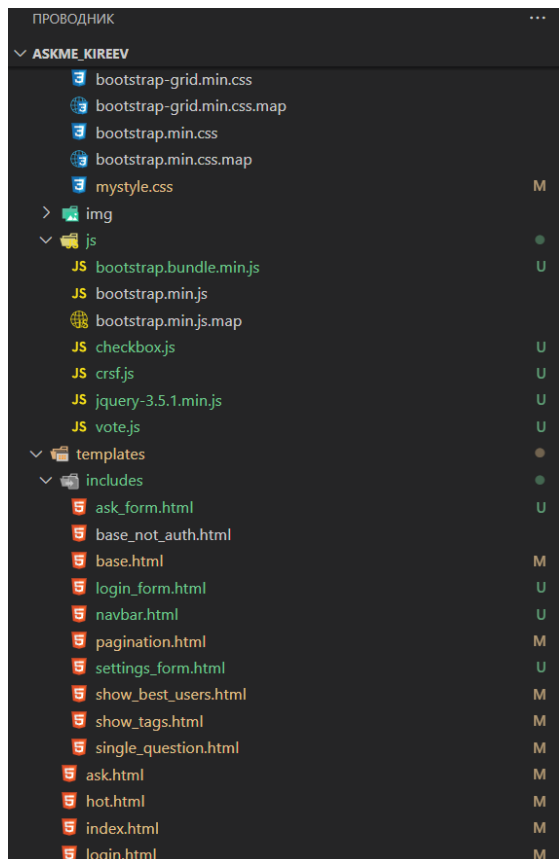
Вопрос – заголовок, содержание, автор, дата создания, теги, рейтинг.

Ответ – содержание, автор, дата написания, флаг правильного ответа, рейтинг.

Тег – слово тега.

Общая структура веб-приложения:





Текст программы:

settings.py:

```
"""
Django settings for askme project.

Generated by 'django-admin startproject' using Django 3.2.9.

For more information on this file, see
https://docs.djangoproject.com/en/3.2/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.2/ref/settings/
"""

from pathlib import Path
import os
```

```
# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-%!t8#41!aq7^^p7@e3g4$_cb4i9pv&^55qpc8k6*=(-!yr023w'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'app.apps.AppConfig',
    'bootstrap4'
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'askme.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': ['templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
            ]
        }
    }
]
```

```

        'django.contrib.messages.context_processors.messages',
    ],
},
],

WSGI_APPLICATION = 'askme.wsgi.application'

# Database
# https://docs.djangoproject.com/en/3.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

# Internationalization
# https://docs.djangoproject.com/en/3.2/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

```

```

USE_L10N = True

USE_TZ = True

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC_URL = '/static/'

STATICFILES_DIRS = [
    BASE_DIR / "static",
]

# Default primary key field type
# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'
LOGIN_URL = "/login/"

```

urls.py:

```

"""askme URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path
from . import settings
from django.contrib.staticfiles.urls import static
from django.contrib.staticfiles.urls import staticfiles_urlpatterns
from app import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.index, name="index"),

```

```

path('ask/', views.ask, name="ask"),
path('settings/', views.settings, name="settings"),
path('register/', views.register, name="register"),
path('hot/', views.hot, name="hot"),
path('question/<int:id>/', views.question, name="question"),
path('tag/<name>/', views.tag, name="tag"),
path('login/', views.login, name='login'),
path('logout/', views.logout, name='logout'),
path('answer/', views.answer, name='answer'),
path('find/', views.find, name='find'),
path('question_vote/', views.question_vote, name='question_vote'),
path('correct/', views.correct, name='correct'),
]
urlpatterns += staticfiles_urlpatterns()
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

models.py:

```

from django.db import models
from django.db.models import Sum, Count
from django.contrib.auth.models import User

class QuestionManager(models.Manager):
    def popular(self):
        return self.prefetch_related('likes', 'author').order_by('-rating')

    def new(self):
        return self.prefetch_related('likes', 'author').order_by('-date')

    def with_tag(self, str):
        return self.prefetch_related('likes', 'author').filter(tags__name=str)

    def find_str(self, str):
        return self.prefetch_related('likes', 'author').filter(title=str)

class AnswerManager(models.Manager):
    def all(self, pk):
        return self.prefetch_related('likes',
        'author').filter(question__id=pk).annotate(like_sum=Sum('likes__like'))

class TagManager(models.Manager):
    def popular(self):
        return self.annotate(count=Count('questions')).order_by('-count')

class ProfileManager(models.Manager):

```

```

    def best_answers(self):
        return self.annotate(count=Count('answers')).order_by('-count')

class Profile(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    avatar = models.ImageField(blank=True)

    objects = ProfileManager()

    def __str__(self):
        return self.user.__str__()

class Question(models.Model):
    title = models.CharField(max_length=30)
    author = models.ForeignKey('Profile', on_delete=models.CASCADE,
related_name='questions')
    text = models.CharField(max_length=5000)
    rating = models.IntegerField(default=0, db_index=True)
    date = models.DateField(auto_now_add=True)
    tags = models.ManyToManyField('Tag', related_name='questions', blank=True)

    objects = QuestionManager()

    def __str__(self):
        return self.title

class Answer(models.Model):
    text = models.CharField(max_length=5000)
    date = models.DateField(auto_now_add=True)
    correct = models.BooleanField(default=False)
    author = models.ForeignKey('Profile', on_delete=models.CASCADE,
related_name='answers')
    question = models.ForeignKey('Question', on_delete=models.CASCADE,
related_name='answers')

    objects = AnswerManager()

    def __str__(self):
        return self.text

class Tag(models.Model):
    name = models.CharField(max_length=50, unique=True)

    objects = TagManager()

    def __str__(self):
        return self.name

```



```

class QuestionLikes(models.Model):
    LIKE_CHOICES = (('like', '1'), ('dis', '-1'))
    like = models.IntegerField(choices=LIKE_CHOICES, default=0)
    question = models.ForeignKey('Question', on_delete=models.CASCADE,
related_name='likes')
    author = models.ForeignKey('Profile', on_delete=models.CASCADE)

    def __str__(self):
        return str(self.like) + ' ' + self.question.__str__()

class AnswerLikes(models.Model):
    LIKE_CHOICES = (('like', '1'), ('dis', '-1'))
    like = models.IntegerField(choices=LIKE_CHOICES, default=0)
    answer = models.ForeignKey('Answer', on_delete=models.CASCADE,
related_name='likes')
    author = models.ForeignKey('Profile', on_delete=models.CASCADE)

    def __str__(self):
        return str(self.like) + ' ' + self.answer.__str__()

```

views.py

```

from django.core.paginator import Paginator
from django.shortcuts import render
from django.http import *
from .models import *
from app.forms import *
from django.contrib.auth.decorators import login_required
from django.contrib import auth
from django.shortcuts import redirect
from django.db.models import F
from django.views.decorators.http import require_POST
from django.urls import reverse
from datetime import date
from urllib.parse import urlencode

'''questions = [
    {
        'title': f"Заголовок лучшего вопроса {i}",
        'text': f"Текст лучшего вопроса {i}",
        'id': i
    } for i in range(100)
]'''

def get_option(str):
    if str == 'like':

```

```

        return 1
    elif str == 'dislike':
        return -1

def get_paginator(request, obj_list, size):
    paginator = Paginator(obj_list, size)
    page = request.GET.get('page')
    return paginator.get_page(page)

def index(request):
    questions_list = Question.objects.new()
    questions = get_paginator(request, questions_list, 10)
    return render(request, "index.html", {'lists': questions})

@login_required
def settings(request):
    if request.method == 'GET':
        form = SettingsForm(initial={'username': request.user.username,
                                     'email': request.user.email,
                                     'avatar': request.user.profile.avatar})
    else:
        form = SettingsForm(data=request.POST, files=request.FILES)
        if form.is_valid():
            user = request.user
            user.username = form.cleaned_data['username']
            user.email = form.cleaned_data['email']
            user.profile.avatar = form.cleaned_data['avatar']
            user.profile.save()
            user.save()
            form = SettingsForm(initial={'username': request.user.username,
                                     'email': request.user.email, 'avatar': request.user.profile.avatar})
        return render(request, 'settings.html', {'form': form})

def login(request):
    alert = ""
    if request.method == 'GET':
        form = LoginForm()
    else:
        form = LoginForm(data=request.POST)
        if form.is_valid():
            user = auth.authenticate(request, **form.cleaned_data)
            if user is not None:
                auth.login(request, user)
                next = request.POST.get('next')
                if not next:
                    next = 'index'
                return redirect(next)
            else:
                form.add_error(None, "Нет данного пользователя")

```

```

        return render(request, 'login.html', {'alert': alert, 'form': form})

@login_required
def logout(request):
    auth.logout(request)
    return redirect(reverse('index'))

def register(request):
    alert = ""
    if request.method == 'GET':
        form = RegisterForm()
    else:
        form = RegisterForm(data=request.POST, files=request.FILES)
        if form.is_valid():
            if User.objects.filter(email=form.cleaned_data['email']):
                alert = "Email already registered."
            else:
                user = User.objects.create_user(form.cleaned_data['username'],
                                                form.cleaned_data['email'],
                                                form.cleaned_data['password1'])
                profile = Profile(avatar=form.cleaned_data['avatar'],
                                user_id=user.id)
                profile.save()
                user.save()
                user = auth.authenticate(username=form.cleaned_data['username'],
                                        password=form.cleaned_data['password1'])
                auth.login(request, user)
                return redirect('index')
        return render(request, 'register.html', {'alert': alert, 'form': form})

@login_required
def ask(request):
    if request.method == 'GET':
        form = QuestionForm()
    else:
        form = QuestionForm(data=request.POST)
        if form.is_valid():
            question = form.save(commit=False)
            question.author = request.user.profile
            question.date = date.today()
            question.save()

            for elem in form.cleaned_data['tags'].split(', '):
                tag = Tag.objects.filter(name=elem)
                if elem and not tag:
                    tag = Tag(name=elem)
                    tag.save()
                else:
                    tag = tag[0]
            question.tags.add(tag.id)

```

```

        question.save()
        return redirect(reverse('question', kwargs={'id': question.id}))
    return render(request, 'ask.html', {'form': form })

@login_required
def answer(request):
    id = request.POST.get('id')
    answer = Answer.objects.create(author_id=request.user.profile.id,
question_id=id,
                                date=date.today(),
text=request.POST.get('text'))
    answer.save()
    return redirect(reverse('question', kwargs={'id': id}))

def hot(request):
    questions = Question.objects.popular()
    questions = get_paginator(request, questions, 20)
    return render(request, 'hot.html', {'lists': questions, 'type': 'hot' })

'''def question(request, id):
    return render(request, "question.html")'''

def question(request, id):
    question = Question.objects.get(id=id)
    answers = Answer.objects.all(id)
    answers = get_paginator(request, answers, 5)
    return render(request, 'question.html', {'lists': answers, 'question':
question})

'''tag_names = ['perl', 'Python', 'TechnoPark', 'MySQL', 'django', 'Mail.Ru',
'Voloshin', 'Firefox']'''

'''def tag(request, name):
    i = 0
    paginator = Paginator(questions, 5)
    page = request.GET.get('page')
    content = paginator.get_page(page)
    while(tag_names):
        if tag_names[i] == name:
            context = {'title': name, 'questions': content}
            break
        i += 1
    return render(request, "tag.html", context)'''

def tag(request, name):
    questions = Question.objects.with_tag(name)
    questions = get_paginator(request, questions, 20)
    return render(request, 'tag.html', {'lists': questions, 'tag': name})

def find(request):

```

```

str = request.GET
print(str.__getitem__("find_str"))
questions = Question.objects.find_str(str.__getitem__("find_str"))
questions = get_paginator(request, questions, 20)
return render(request, 'index.html', {'lists': questions})

@login_required
@require_POST
def question_vote(request):
    data = request.POST
    inc = get_option(data['action'])
    rating = 0
    error = ""

    question = Question.objects.get(id=data['qid'])
    like = QuestionLikes.objects.filter(question_id=data['qid'],
author_id=request.user.id)
    if question and not like:
        rating = question.rating + inc
        question.rating = F('rating') + inc
        question.save()
        QuestionLikes.objects.create(question_id=question.id,
author_id=request.user.profile.id,
                                like=str(inc))

    else:
        error = "Error"

    return JsonResponse({'rating': rating, 'error': error})

@login_required
@require_POST
def correct(request):
    data = request.POST
    error = ""
    answer = Answer.objects.get(id=data['aid'])
    value = True
    if answer and answer.question.author.id == request.user.profile.id:
        if answer.correct:
            answer.correct = False
            answer.save()
            value = False
        else:
            answer.correct = True
            answer.save()
    else:
        error = "Error"
    return JsonResponse({'value': value, 'error': error})

```

forms.py

```

from django import forms
from django.forms.widgets import Textarea
from .models import *
from django.contrib.auth.forms import UserCreationForm

class SettingsForm(forms.Form):
    username = forms.CharField()
    email = forms.EmailField()
    avatar = forms.ImageField()

class LoginForm(forms.Form):
    username = forms.CharField()
    password = forms.CharField(widget=forms.PasswordInput())

class QuestionForm(forms.ModelForm):
    tags = forms.CharField(widget=forms.TextInput(attrs={'placeholder':
'tags...'}))
    text = forms.CharField(widget=forms.Textarea)
    class Meta:
        model = Question
        fields = ['title', 'text']

class RegisterForm(UserCreationForm):
    email = forms.EmailField()
    avatar = forms.ImageField()

```

apps.py

```

from django.apps import AppConfig

class AppConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'app'

```

askme_tags.py

```

from django import template
from django.db.models.lookups import LessThanOrEqual
from app.models import *
import random
register = template.Library()

questions_list = User.objects.filter(id__gte = 10010)
questions_list = list(questions_list)
questions_list2 = []

```

```

n = 0
for i in reversed(questions_list):
    if n < 5:
        questions_list2.append(i)
        n = n + 1

for i in questions_list2:
    print(i)

tags_list = Tag.objects.popular()
tags_list1 = tags_list[0:3]
tags_list2 = tags_list[4:6]
tags_list3 = tags_list[7:10]

@register.simple_tag()
def show_tags():
    tags = {"str1":tags_list1, "str2": tags_list2, "str3": tags_list3}
    return tags

@register.simple_tag()
def show_best_users():

    usernames = questions_list2

    return usernames

```

mystyle.css

```

.wpx-25{
    width: 25px;
}
.wpx-50{
    width: 50px;
}

.hpx-25{
    height: 25px;
}
.hpx-50{
    height: 50px;
}
.hpx-75{
    height: 75px;
}
.hpx-100{
    height: 100px;
}
.hpx-125{
    height: 125px;
}

```

```
}  
.hpx-150{  
    height: 150px;  
}  
.hpx-175{  
    height: 175px;  
}  
.hpx-200{  
    height: 200px;  
}  
.hpx-250{  
    height: 250px;  
}  
.hpx-300{  
    height: 300px;  
}  
.hpx-350{  
    height: 350px;  
}  
.hpx-400{  
    height: 400px;  
}  
.hpx-450{  
    height: 450px;  
}  
.hpx-500{  
    height: 500px;  
}  
.hpx-550{  
    height: 550px;  
}  
.hpx-600{  
    height: 600px;  
}  
.hpx-650{  
    height: 650px;  
}  
.hpx-700{  
    height: 700px;  
}  
.hpx-750{  
    height: 750px;  
}  
.hpx-800{  
    height: 800px;  
}  
.hpx-850{  
    height: 850px;  
}
```



```
.h-5{
  height: 5%;
}
.h-10{
  height: 10%;
}
.h-15{
  height: 15%;
}
.h-20{
  height: 20%;
}
.h-25{
  height: 25%;
}
.h-30{
  height: 30%;
}
.h-35{
  height: 35%;
}
.h-40{
  height: 40%;
}
.h-45{
  height: 45%;
}
.h-50{
  height: 50%;
}
.h-55{
  height: 55%;
}
.h-60{
  height: 60%;
}
.h-65{
  height: 65%;
}
.h-70{
  height: 70%;
}
.h-75{
  height: 75%;
}
.h-80{
  height: 80%;
}
.h-85{
  height: 85%;
}
}
```

```
.h-90{
  height: 90%;
}
.h-95{
  height: 95%;
}
.h-100{
  height: 100%;
}

.w-5{
  width: 5%;
}
.w-10{
  width: 10%;
}
.w-15{
  width: 15%;
}
.w-20{
  width: 20%;
}
.w-25{
  width: 25%;
}
.w-30{
  width: 30%;
}
.w-35{
  width: 35%;
}
.w-40{
  width: 40%;
}
.w-45{
  width: 45%;
}
.w-50{
  width: 50%;
}
.w-55{
  width: 55%;
}
.w-60{
  width: 60%;
}
.w-65{
  width: 65%;
}
.w-70{
  width: 70%;
}
```

```
}  
.w-75{  
  width: 75%;  
}  
.w-80{  
  width: 80%;  
}  
.w-85{  
  width: 85%;  
}  
.w-90{  
  width: 90%;  
}  
.w-95{  
  width: 95%;  
}  
.w-100{  
  width: 100%;  
}  
  
.h-img-50{  
  height: 50%;  
  width: 50%;  
}  
.h-img-25{  
  height: 25%;  
  width: 25%;  
}  
.h-img-90{  
  width: 5%;  
}  
  
.between{  
  display: flex;  
  justify-content: space-between !important;  
}  
.text-vert-center{  
  display: flex;  
  align-items: center;  
}  
.no-margin{  
  margin: 0;  
}  
  
.link{  
  color: white;  
}  
.link:hover{  
  transition: 0.2s linear;  
  color: #2d7448;
```

```
}

/*Добавлен свой класс контейнер (не используется для вывода основного контента)*/
.my-container{
    display: flex;
    width: 1320px;
    justify-content: space-between;
    align-items: center;
}

.green{
    background: green;
}
.yellow{
    background: yellow;
}
.pink{
    background: pink;
}
.blue{
    background: blue;
}
.orange{
    background: orange;
}
.purple{
    background: purple;
}
.aqua{
    background: aqua;
}
.brown{
    background: brown;
}
.sandybrown{
    background: sandybrown;
}
.darkcyan{
    background: darkcyan;
}
.olive{
    background: olive;
}
.violet{
    background: violet;
}

.dislike{
    transition: 0.1s linear;
    fill: #212529;
}
```

```

.like{
  transition: 0.1s linear;
  fill: #212529;
}

.dislike:hover{
  fill: #dc3b45;
}

.like:hover{
  fill: #2d7448;
}

/*чекбокс выбора корректного ответа*/
.label-cbx-unchecked {
  user-select: none;
  cursor: pointer;
  margin-bottom: 0; }
.label-cbx-unchecked input:checked + .checkbox {
  border-color: #2d7448; }
.label-cbx-unchecked input:checked + .checkbox svg path {
  fill: #2d7448; }
.label-cbx-unchecked input:checked + .checkbox svg polyline {
  stroke-dashoffset: 0; }
.label-cbx-unchecked:hover .checkbox svg path {
  stroke-dashoffset: 0; }
.label-cbx-unchecked .checkbox {
  position: relative;
  top: 2px;
  float: left;
  margin-right: 8px;
  width: 20px;
  height: 20px;
  border: 2px solid #C8CCD4;
  border-radius: 3px; }
.label-cbx-unchecked .checkbox svg {
  position: absolute;
  top: -2px;
  left: -2px; }
.label-cbx-unchecked .checkbox svg path {
  fill: none;
  stroke: #2d7448;
  stroke-width: 2;
  stroke-linecap: round;
  stroke-linejoin: round;
  stroke-dasharray: 71px;
  stroke-dashoffset: 71px;
  transition: all .6s ease; }
.label-cbx-unchecked .checkbox svg polyline {
  fill: none;

```

```

        stroke: #FFF;
        stroke-width: 2;
        stroke-linecap: round;
        stroke-linejoin: round;
        stroke-dasharray: 18px;
        stroke-dashoffset: 18px;
        transition: all .3s ease; }
.label-cbx-unchecked > span {
    pointer-events: none;
    vertical-align: middle; }

.label-cbx-checked {
    user-select: none;
    cursor: pointer;
    margin-bottom: 0; }
    .label-cbx-checked input + .checkbox {
        border-color: #2d7448; }
        .label-cbx-checked input + .checkbox svg path {
            fill: #2d7448; }
        .label-cbx-checked input + .checkbox svg polyline {
            stroke-dashoffset: 0; }
        .label-cbx-checked input:checked + .checkbox {
            border-color: #C8CCD4; }
            .label-cbx-checked input:checked + .checkbox svg path {
                fill: none; }
            .label-cbx-checked input:checked + .checkbox svg polyline {
                stroke-dashoffset: 18px; }
    .label-cbx-checked: hover .checkbox svg path {
        stroke-dashoffset: 0; }
    .label-cbx-checked .checkbox {
        position: relative;
        top: 2px;
        float: left;
        margin-right: 8px;
        width: 20px;
        height: 20px;
        border: 2px solid #C8CCD4;
        border-radius: 3px; }
        .label-cbx-checked .checkbox svg {
            position: absolute;
            top: -2px;
            left: -2px; }
        .label-cbx-checked .checkbox svg path {
            fill: none;
            stroke: #2d7448;
            stroke-width: 2;
            stroke-linecap: round;
            stroke-linejoin: round;
            stroke-dasharray: 71px;
            stroke-dashoffset: 71px;
            transition: all .6s ease; }

```

```

        .label-cbx-checked .checkbox svg polyline {
            fill: none;
            stroke: #FFF;
            stroke-width: 2;
            stroke-linecap: round;
            stroke-linejoin: round;
            stroke-dasharray: 18px;
            stroke-dashoffset: 18px;
            transition: all .3s ease; }
        .label-cbx-checked > span {
            pointer-events: none;
            vertical-align: middle; }

.invisible {
    position: absolute;
    z-index: -1;
    width: 0;
    height: 0;
    opacity: 0; }

```

index.html

```

{% extends "includes/base.html"%}
{% load static %}

{% block content %}
<span class="d-flex flex-row justify-content-start align-items-end pt-4">
    <h1 class="w-40">Новые вопросы</h1>
    <a href="{% url 'hot' %}"><h5>Лучшие вопросы</h5></a>
</span>
<div class="d-flex flex-column">
    {% for question in lists %}
        {% include "includes/single_question.html" %}
    {% endfor %}
</div>
{% include "includes/pagination.html" %}
{% endblock content %}

```

login.html

```

{% extends "includes/base.html"%}
{% load static %}
{% load bootstrap4 %}

{% block content %}

{%include "includes/login_form.html"%}

```

```
{% endblock content %}
```

ask.html

```
{% extends "includes/base.html"%}
{% load static %}
{% load bootstrap4 %}
{% block question-header%}
<div class="question-header">
    <span class="big-text ">New Question<span>
</div>
{% endblock question-header%}

{% block content%}
    {%include "includes/ask_form.html"%}
{% endblock content%}
```

hot.html

```
{% extends "includes/base.html"%}
{% load static %}

{% block content %}
<span class="d-flex flex-row justify-content-start align-items-end pt-4">
    <h1 class="w-40">Лучшие вопросы</h1>
    <a href="{% url 'index' %}"><h5>Новые вопросы</h5></a>
</span>
{% for question in lists %}
    {% include "includes/single_question.html" %}
{% endfor %}
{% include "includes/pagination.html" %}
{% endblock content %}
```

question.html

```
{% extends "includes/base.html"%}
{% load static %}

{% block content %}
<div class="card mt-4 w-95 border-dark">
    <div class="row">
        <div class="col-md-2 d-flex flex-column justify-content-start align-items-center mt-3 ml-2">
            

```



```

<div class="d-flex flex-row justify-content-evenly align-items-start mt-2">
    <svg class="js-vote w-15 dislike js-{{ question.id }}" data-action="dislike" data-qid="{{ question.id }}" enable-background="new 0 0 32 32" id="Glyph" version="1.1" viewBox="0 0 32 32" xml:space="preserve" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink"><path d="M2.156,14.90112.489-8.725C5.012,4.895,6.197,4,7.528,4h13.473C21.554,4,22,4.448,22,5v14 c0,0.215-0.068,0.425-0.197,0.5971-5.392,7.24C15.813,27.586,14.951,28,14.027,28c-1.669,0-3.026-1.357-3.026-3.026V20H5.999 c-1.265,0-2.427-0.579-3.188-1.589C2.047,17.399,1.809,16.12,2.156,14.901z" id="XMLID_259_"/><path d="M25.001,20h4C29.554,20,30,19.552,30,19V5c0-0.552-0.446-1-0.999-1h-4c-0.553,0-1,0.448-1,1v14 C24.001,19.552,24.448,20,25.001,20z M27.001,6.5c0.828,0,1.5,0.672,1.5,1.5c0,0.828-0.672,1.5-1.5,1.5c-0.828,0-1.5-0.672-1.5-1.5 C25.501,7.172,26.173,6.5,27.001,6.5z" id="XMLID_260_"/></svg>
    <div id="block{{ question.id }}" class="wpx-25 text-center">
        {% if question.rating >= 0 %}
        <p class="text-success">{{ question.rating }}</p>
        {% else %}
        <p class="text-danger">{{ question.rating }}</p>
        {% endif %}
    </div>
    <svg class="js-vote w-15 like js-{{ question.id }}" data-action="like" data-qid="{{ question.id }}" enable-background="new 0 0 32 32" id="Glyph" version="1.1" viewBox="0 0 32 32" xml:space="preserve" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink"><path d="M29.845,17.0991-2.489,8.725C26.989,27.105,25.804,28,24.473,28H11c-0.553,0-1-0.448-1-1V13 c0-0.215,0.069-0.425,0.198-0.59715.392-7.24C16.188,4.414,17.05,4,17.974,4C19.643,4,21,5.357,21,7.026V12h5.002 c1.265,0,2.427,0.579,3.188,1.589C29.954,14.601,30.192,15.88,29.845,17.099z" id="XMLID_254_"/><path d="M7,12H3c-0.553,0-1,0.448-1,1v14c0,0.552,0.447,1,1,1h4c0.553,0,1-0.448,1-1V13C8,12.448,7.553,12,7,12z M5,25.5c-0.828,0-1.5-0.672-1.5-1.5c0-0.828,0.672-1.5,1.5-1.5c0.828,0,1.5,0.672,1.5,1.5C6.5,24.828,5.828,25.5,5,25.5z" id="XMLID_256_"/></svg>
</div>
</div>
<div class="col-md-9">
    <div class="card-body">
        <h3 class="card-title">{{ question.title }}</h3>
        <p class="card-text">{{ question.text }}</p>
        <span class="d-flex flex-row">
            <span class="col-md-7 d-flex flex-row">
                <p class="card-text">Теги:&nbsp;</p>
                {% for tag in question.tags.all %}
                <a href="{% url 'tag' name=tag %}" class="card-link">{{ tag
            }}</a>

                <p class="card-text">,&nbsp;</p>
                {% endfor %}
            </span>

```

```

    </span>
    <p class="card-text"><small class="text-muted">{{ question.date
}}</small></p>
  </div>
</div>
</div>
<hr class="w-95 no-margin mt-4">
{% for answer in lists %}
{% if answer.correct %}
<div class="card mt-4 w-95" id="block-ans-{{ answer.id }}">
  <div class="row">
    <div class="col-md-2 d-flex flex-column justify-content-start align-
items-center mt-3">
      
      <div class="d-flex flex-row justify-content-evenly align-items-start
mt-2">

        </div>
      </div>
    <div class="col-md-9">
      <div class="card-body">

        <p class="card-text">{{ answer.text }}</p>
        <span class="d-flex flex-row">
          <input class="form-check-input correct-{{ answer.id }}"
type="checkbox" value="" data-aid="{{ answer.id }}" data-qid="{{ question.id }}"
checked>

          </span>
          <h6>ОТВЕТ!</h6>
          <p class="card-text"><small class="text-muted">{{ answer.date
}}</small></p>
        </div>
      </div>
    </div>
  </div>
  </div>
</div>
{% endif %}
{% endfor %}
{% for answer in lists %}
{% if answer.correct == 0 %}
<div class="card mt-4 w-95" id="block-ans-{{ answer.id }}">
  <div class="row">
    <div class="col-md-2 d-flex flex-column justify-content-start align-
items-center mt-3">
      
      <div class="d-flex flex-row justify-content-evenly align-items-start
mt-2">

```

```

        </div>
    </div>
    <div class="col-md-9">
        <div class="card-body">

            <p class="card-text">{{ answer.text }}</p>
            <span class="d-flex flex-row">
                <input class="form-check-input correct-{{ answer.id }}"
type="checkbox" value="" data-aid="{{ answer.id }}" data-qid="{{ question.id }}">
            </span>
            <p class="card-text"><small class="text-muted">{{ answer.date
}}</small></p>
        </div>
    </div>
</div>
{% endif %}
{% endfor %}

<hr class="w-95 no-margin mt-4 mb-4">
<form class="answer" action="/answer/" method="post">
    {% csrf_token %}
    <div class="mb-3">
        <input type="hidden" name="id" value="{{ question.id }}" />
        <textarea name="text" class="form-control" rows="4" placeholder="Enter your
answer here"></textarea>
    </div>
    <div class="form-btn answer-btn">
        <button class="btn btn-primary" type="submit">Ответить</button>
    </div>
<br>
</form>
{% endblock content %}

```

register.html

```

{% extends "includes/base.html"%}
{% load static %}
{% load bootstrap4 %}

{% block content %}
<div class="col-4 q-title center">
    <h1 class="title-content">Регистрация</h1>
</div>

{% if alert %}
    <div class="alert alert-danger center" role="alert">
        {{ alert }}
    </div>

```

```
{% endif %}

<form class="signup center"  enctype="multipart/form-data" action="/register/"
method="post">
    {% csrf_token %}
    <div class="mb-3">
        {% bootstrap_form form %}
    </div>
    {% buttons %}
    <button type="submit" class="btn btn-primary">Зарегистрироваться</button>
    {% endbuttons %}
</form>
{% endblock content %}
```

settings.html

```
{% extends "includes/base.html"%}
{% load static %}
{% load bootstrap4 %}
{% block question-header%}
<div class="question-header">
    <span class="big-text ">Profile: {{ request.user.username }}<span>
</div>
{% endblock question-header%}

{% block content%}
    {%include "includes/settings_form.html"%}
{% endblock content%}
```

tag.html

```
{% extends "includes/base.html"%}
{% load static %}

{% block content %}
<span class="d-flex flex-row justify-content-start align-items-end pt-4">
    <h2 class="w-40">Tag: <a href="{% url 'tag' tag %}">{{tag}}</a></h2>
</span>
{% for question in lists %}
    {% include "includes/single_question.html" %}
{% endfor %}
{% include "includes/pagination.html" %}
{% endblock content %}
```

ask_form.html

```
{% load bootstrap4 %}
```

```

<form class="col-8" action="/ask/" method="post">
    {% csrf_token %}
<div class="col-12">
    {% bootstrap_form form %}

</div>
    {% buttons %}
<div class="form-btn">
    <button class="btn btn-primary margintop40" type="submit">ASK!</button>
</div>
    {% endbuttons %}
</form>

```

base.html

```

{% load static %}
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
        <title>AskME</title>
        <!-- Bootstrap core CSS -->
        <link href="{% static 'css/bootstrap.min.css' %}" rel="stylesheet">
        <link href="{% static 'css/bootstrap.min.css.map' %}" rel="stylesheet">
        <link href="{% static 'css/bootstrap-grid.min.css' %}" rel="stylesheet">
        <link href="{% static 'css/bootstrap-grid.min.css.map' %}" rel="stylesheet">
        <link href="{% static 'css/mystyle.css' %}" rel="stylesheet">
        <!-- Custom styles for this template -->

    </head>
    <body class="bg-light">
        {% include "includes/navbar.html" %}
        <section class="d-flex justify-content-center text-dark">
            <div class="my-container align-items-start">
                <content class="col-9 container d-flex flex-column">
                    {% block content %}
                    {% endblock content %}
                </content>
                <div class="col-3 hpx-600 mt-4">
                    <div class="col h-50">
                        <div class="col d-flex align-items-center h-30">
                            <h3>Популярные теги:</h3>
                        </div>
                        <div class="col h-70">
                            {% include "includes/show_tags.html" %}
                        </div>
                    </div>
                </div>
            </div>
        </section>
    </body>
</html>

```

```

        <div class="col h-50">
            <div class="col d-flex align-items-center h-30">
                <h3>Последние зарегистрированные:</h3>
            </div>
            <span class="col h-70">
                {% include "includes/show_best_users.html" %}
            </span>
        </div>
    </div>
</div>
</section>
<footer class="bg-dark hpx-150 d-flex justify-content-center" >
    <div class="my-container h-100">
        <div class="col-4 h-100">
            <div class="col h-30 d-flex align-items-center text-white pt-
2">
                <h4>AskME</h4>
            </div>
            <div class="d-flex h-70">
                <div class="col h-100">
                    <div class="col pt-1 pb-1">
                        <a class="text-decoration-none link" href="{% url
'index' %}">Главная</a>
                    </div>
                    <div class="col pt-1 pb-1">
                        <a class="text-decoration-none link" href="{% url
'ask' %}">Добавить вопрос</a>
                    </div>
                    <div class="col pt-1 pb-1">
                        <a class="text-decoration-none link" href="{% url
'settings' %}">Настройки</a>
                    </div>
                </div>
                <div class="col h-100">
                    <div class="col h-100">
                    </div>
                </div>
            </div>
        </div>
        <div class="col-3 h-100 d-flex pr-4">
            <div class="col h-100 d-flex justify-content-center align-
items-end">
                <a href="https://t.me/V_R_K" class="d-flex align-items-end
justify-content-center pb-4 h-30 w-30"></a>
            </div>
            <div class="col h-100 d-flex justify-content-center align-
items-end">

```

```

        <a class="d-flex align-items-end justify-content-center pb-
4 text-decoration-none link lead h-20"
href="https://park.vk.company/profile/an.kireev/">Технопарк</a>
    </div>
</div>
</div>
</footer>
    <script src="{% static 'js/jquery-3.5.1.min.js' %}"></script>
    <script src="{% static 'js/bootstrap.bundle.min.js' %}"></script>
    <script src="{% static 'js/crsf.js' %}"></script>
    <script src="{% static 'js/checkbox.js' %}"></script>

    <script src="{% static 'js/vote.js' %}"></script>
</body>
</html>

```

login_form.html

```

{% load bootstrap4 %}
<form class="login col-8" action="/login/" method="post">
    {% csrf_token %}
    <input type="hidden" name="next" value="{{ request.GET.next }}" />
    <div class="mb-3">
        {% bootstrap_form form %}
    </div>
    {% buttons %}
    <button type="submit" class="btn btn-primary">Вход</button>
    <br>
    <a class="marginleft40" href="{% url 'register' %}">Создать новый
аккаунт</a>
    {% endbuttons %}

</form>
{% comment %}
<form class="login" action="/login/" method="post">
{% csrf_token %}
<input type="hidden" name="next" value="{{ request.GET.next }}" />
<div class="row settings-row ">

    <div class="col-3">
        <span class="normal-text ">Вход<span>
    </div>
    <div class="col-9 settings-field ">
        <input type="text" class="form-control" placeholder="Login" aria-
label="Login">
    </div>
</div>
<div class="row settings-row ">

```

```

<div class="col-3">
    <span class="normal-text ">Пароль<span>
</div>
<div class="col-9 settings-field ">
    <div>
        <input type="text" class="form-control" placeholder="Password" aria-label="Password">
    </div>
</div>
</div>
<div class="row settings-row ">

    <div class="col-3">

    </div>
    <div class="col-9 ask-btn-pad">

        <button id="ask-btn" type="submit" class="btn btn-primary">Вход</button><br><br>
        <a class = "margintop40" href = "{% url 'register' %}">Создать новый аккаунт</a>
    </div>
</div>
</form> {% endcomment %}

```

navbar.html

```

{% load static %}

<header class="d-flex justify-content-center text-white bg-dark">
    <div class="my-container hpx-100">
        <a class="navbar-brand col-md-1 text-white" href="{% url 'index' %}"><h3>AskME</h3></a>
        <form class="form col-md-4 offset-md-1" action="/find/" method="get">
            <input class="form-control " name="find_str" type="text" placeholder="Поиск..." aria-label="default input example">
        </form>
        <div class=" text-start col-md-2 ">
            <a href="{% url 'ask' %}"><button type="button" class="btn btn-success center-block w-50">ASK!</button></a>
        </div>

        {% if request.user.is_authenticated %}

        <div class="col-md-1 d-flex justify-content-end align-items-end">
            
        </div>
        <div class="col-md-2 ">
            <div class="row h-50">

```



```

        <a class="text-decoration-none link" href="{% url 'settings'
%}"><p class="lead d-flex align-items-end">{{ request.user.username }}</p></a>
        </div>
        <div class="row h-50">
            <div class="col d-flex align-items-center">
                <a class="text-decoration-none link" href="{% url
'settings' %}">Настройки</a>
            </div>
            <div class="col d-flex align-items-center">
                <a class="text-decoration-none link" href="{% url
'logout' %}">Выход</a>
            </div>
        </div>
    </div>
    {% else %}

    <div class="col-md-2 d-flex flex-row">
        <div class="col d-flex align-items-center justify-content-end">
            <a class="text-decoration-none link" href="{% url 'register'
%}">Зарегистрироваться</a>
        </div>
        <div class="col d-flex align-items-center justify-content-end">
            <a class="text-decoration-none link" href="{% url 'login'
%}">Войти</a>
        </div>
    </div>
    {% endif %}
</div>
</header>

```

pagination.html

```
{% load static %}
<div class="d-flex align-items-center justify-content-end w-95 mb-2">
  <nav class="d-flex mt-4">
    <ul class="pagination">
      {% if lists.has_previous %}
        <li class="page-item enabled">
          <a class="page-link" href="?page=1">&laquo;</a>
        </li>
        <li class="page-item enabled">
          <a class="page-link" href="?page={{ lists.previous_page_number }}">{{
lists.previous_page_number }}</a>
        </li>
      {% else %}
        <li class="page-item disabled">
          <a class="page-link">&laquo;</a>
        </li>
      {% endif %}
    </ul>
  </nav>
</div>
```

```

        {% if lists.has_next != false and lists.has_next != false %}
        <li class="page-item current active"><a class="page-link">{{ lists.number
}}</a></li>
        {% endif %}

        {% if lists.has_next %}
        <li class="page-item">
            <a class="page-link" href="?page={{ lists.next_page_number }}">{{
lists.next_page_number }}</a>
        </li>
        <li class="page-item">
            <a class="page-link" href="?page={{ lists.paginator.num_pages
}}">&raquo;</a>
        </li>
        {% endif %}
    </ul>
</nav>
</div>

```

settings_form.html

```

{% load bootstrap4 %}
<form class="signup margintop40 col-8" enctype="multipart/form-data"
action="/settings/" method="post">
    {% csrf_token %}
    <div class="mb-5 ">
        {% bootstrap_form form %}
    </div>
    {% buttons %}
    <button type="submit" class="btn btn-primary">Save</button>
    {% endbuttons %}
</form>

```

show_best_users.html

```

{% load static %}
{% load askme_tags %}

{% show_best_users as users %}
{% for user in users %}
<a href="#"><p>{{ user.username }}</p></a>
{% endfor %}

```

show_tags.html

```

{% load static %}
{% load askme_tags %}

{% show_tags as tags %}
{% for key,value in tags.items %}
<span class="d-flex flex-row justify-content-around align-items-center pt-2">

    {% for t in value %}
    <a class="text-decoration-none" href="{% url 'tag' t %}">
        <h5 class="text-dark">{{ t }}</h5>
    </a>

    {% endfor %}
    <br>

</span>
{% endfor %}

```

single_question.html

```

{% load static %}
<div class="card mt-4 w-95">
    <div class="row">
        <div class="col-md-2 d-flex flex-column justify-content-start align-items-center mt-3">
            
            <div class="d-flex flex-row justify-content-evenly align-items-start mt-2">
                <svg class="js-vote w-15 dislike js-{{ question.id }}" data-
action="dislike" data-qid="{{ question.id }}" enable-background="new 0 0 32 32"
id="Glyph" version="1.1" viewBox="0 0 32 32" xml:space="preserve"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"><path d="M2.156,14.90112.489-
8.725C5.012,4.895,6.197,4,7.528,4h13.473C21.554,4,22,4.448,22,5v14 c0,0.215-
0.068,0.425-0.197,0.5971-5.392,7.24C15.813,27.586,14.951,28,14.027,28c-1.669,0-
3.026-1.357-3.026-3.026V20H5.999 c-1.265,0-2.427-0.579-3.188-
1.589C2.047,17.399,1.809,16.12,2.156,14.901z" id="XMLID_259_"/><path
d="M25.001,20h4C29.554,20,30,19.552,30,19V5c0-0.552-0.446-1-0.999-1h-4c-0.553,0-
1,0.448-1,1v14 C24.001,19.552,24.448,20,25.001,20z
M27.001,6.5c0.828,0,1.5,0.672,1.5,1.5c0,0.828-0.672,1.5-1.5,1.5c-0.828,0-1.5-
0.672-1.5-1.5 C25.501,7.172,26.173,6.5,27.001,6.5z" id="XMLID_260_"/></svg>

                <span id="block{{ question.id }}" class="wpx-25 text-center">
                    {% if question.rating >= 0 %}
                    <p class="text-success">{{ question.rating }}</p>
                    {% else %}

```

```

        <p class="text-danger">{{ question.rating }}</p>
        {% endif %}
    </span>
    <svg class="js-vote w-15 like js-{{ question.id }}" data-action="like"
data-qid="{{ question.id }}" enable-background="new 0 0 32 32" id="Glyph"
version="1.1" viewBox="0 0 32 32" xml:space="preserve"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"><path d="M29.845,17.0991-
2.489,8.725C26.989,27.105,25.804,28,24.473,28H11c-0.553,0-1-0.448-1-1V13 c0-
0.215,0.069-0.425,0.198-0.59715.392-
7.24C16.188,4.414,17.05,4,17.974,4C19.643,4,21,5.357,21,7.026V12h5.002 c1.265,0,
2.427,0.579,3.188,1.589C29.954,14.601,30.192,15.88,29.845,17.099z"
id="XMLID_254_"/><path d="M7,12H3c-0.553,0-1,0.448-
1,1v14c0,0.552,0.447,1,1,1h4c0.553,0,1-0.448,1-
1V13C8,12.448,7.553,12,7,12z M5,25.5c-0.828,0-1.5-0.672-1.5-1.5c0-0.828,0.672-
1.5,1.5-1.5c0.828,0,1.5,0.672,1.5,1.5C6.5,24.828,5.828,25.5,5,25.5z"
id="XMLID_256_"/></svg>

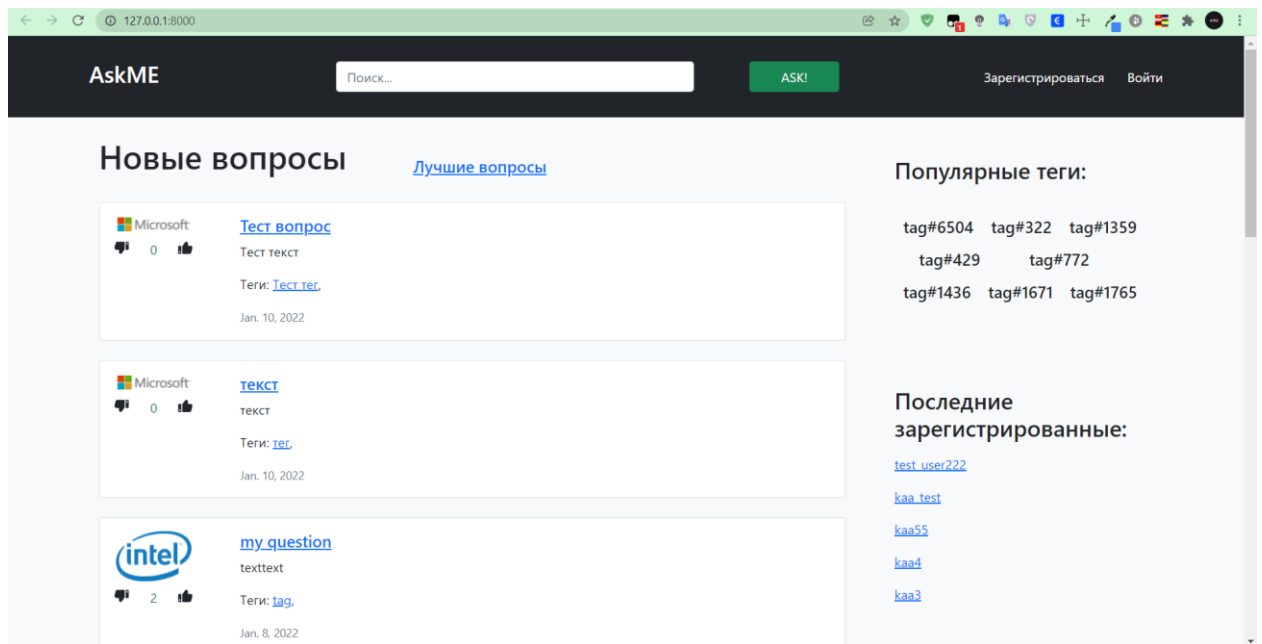
    </div>
</div>
<div class="col-md-9">
    <div class="card-body">
        <a href="{% url 'question' id=question.id %}"><h5 class="card-title">{{
question.title }}</h5></a>
        <p class="card-text">{{ question.text }}</p>
        <span class="d-flex flex-row">

            <span class="col-md-7 d-flex flex-row">
                <p class="card-text">Теги:&nbsp;</p>
                {% for tag in question.tags.all %}
                <a href="{% url 'tag' name=tag %}" class="card-link">{{ tag }}</a>
                <p class="card-text">,&nbsp;</p>
                {% endfor %}
            </span>
        </span>
        <p class="card-text"><small class="text-muted">{{ question.date
}}</small></p>
    </div>
</div>
</div>
</div>

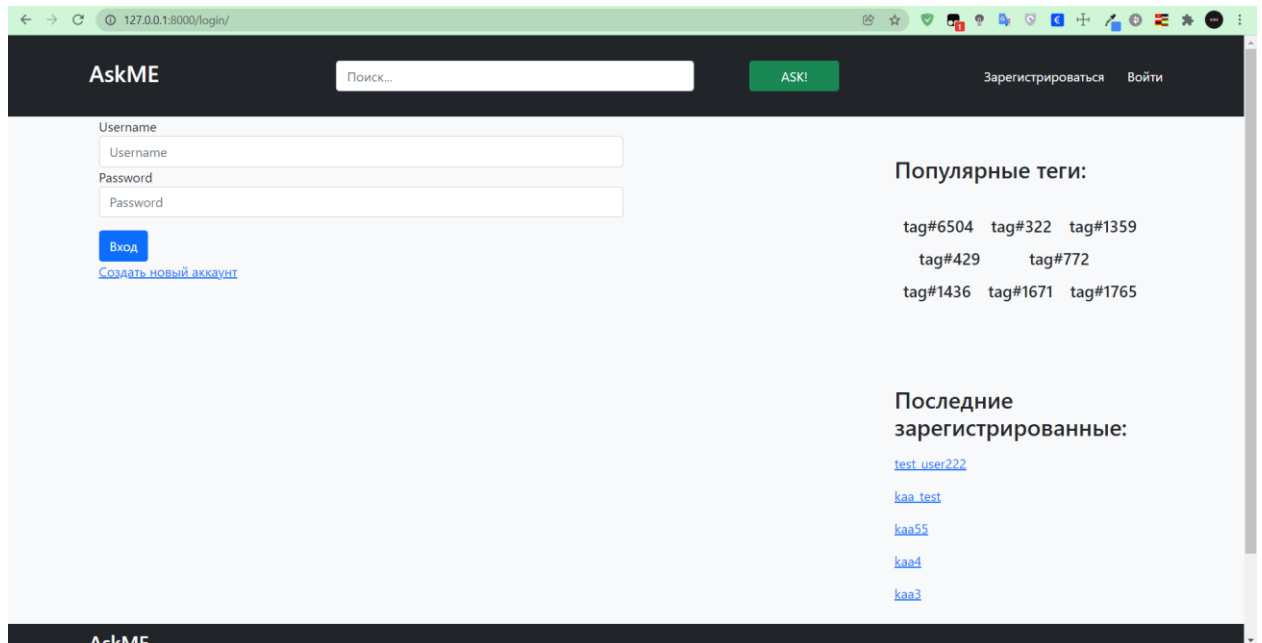
```

Результат работы программы:

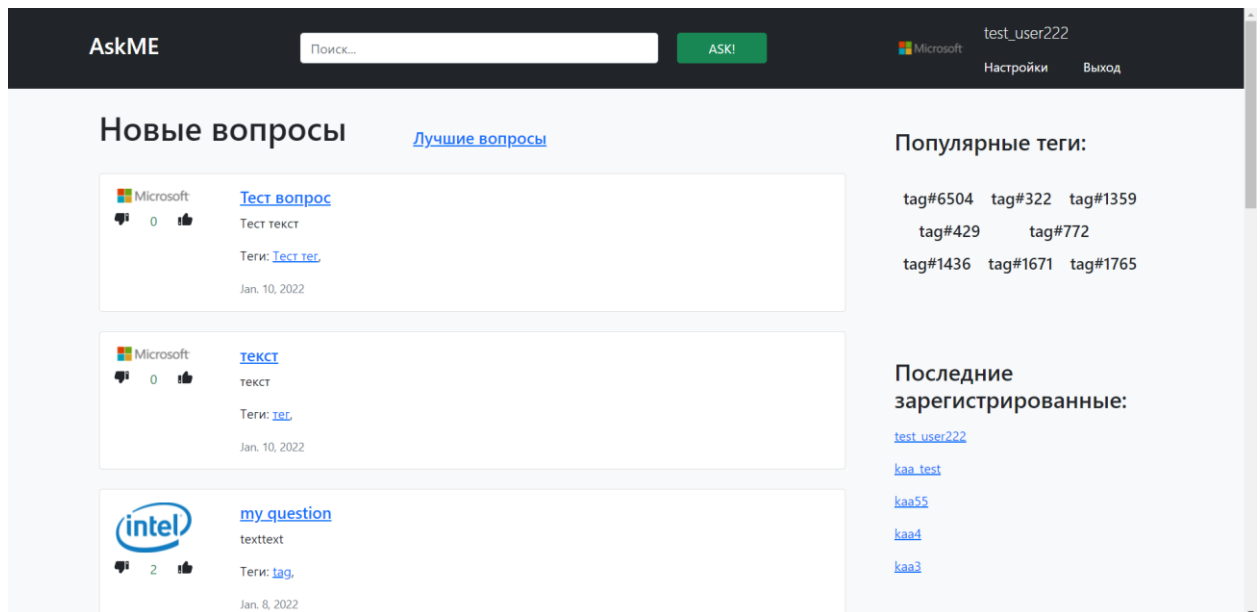
Начальная страница:



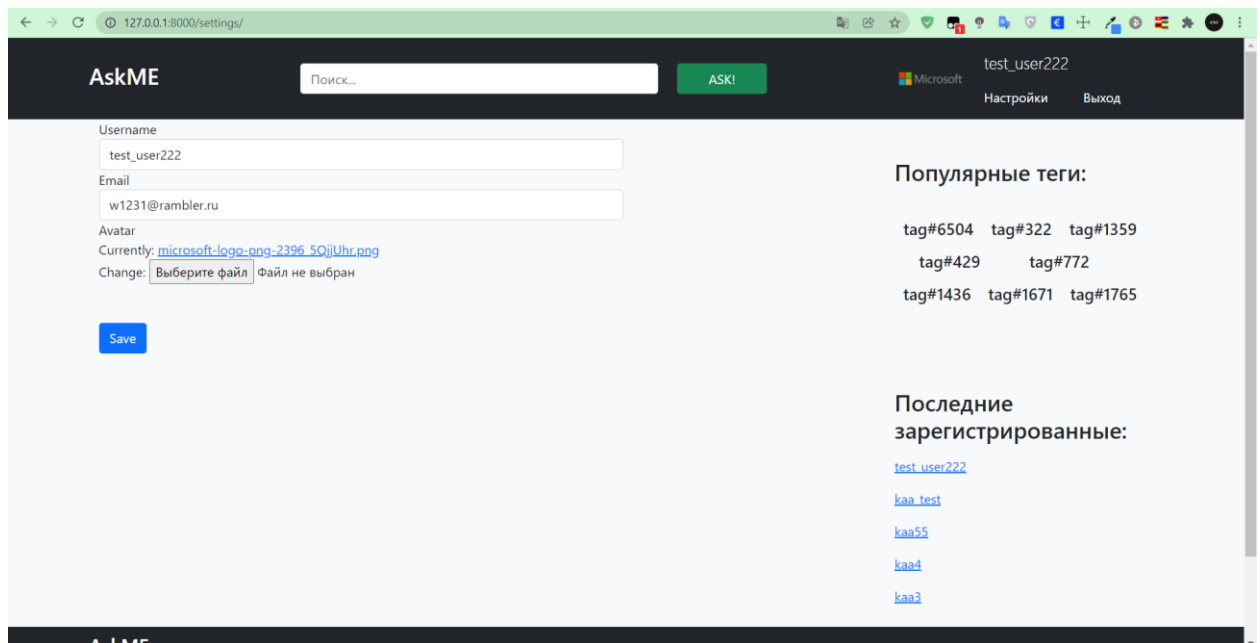
Вход:



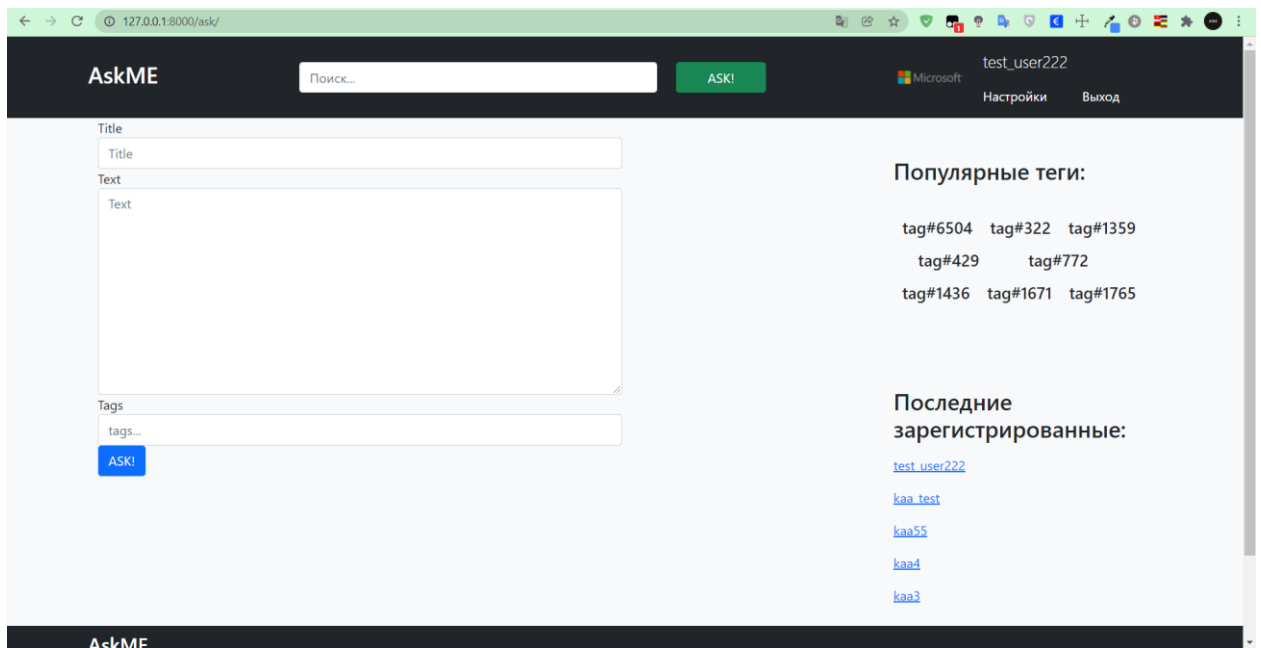
Пройденная авторизация:



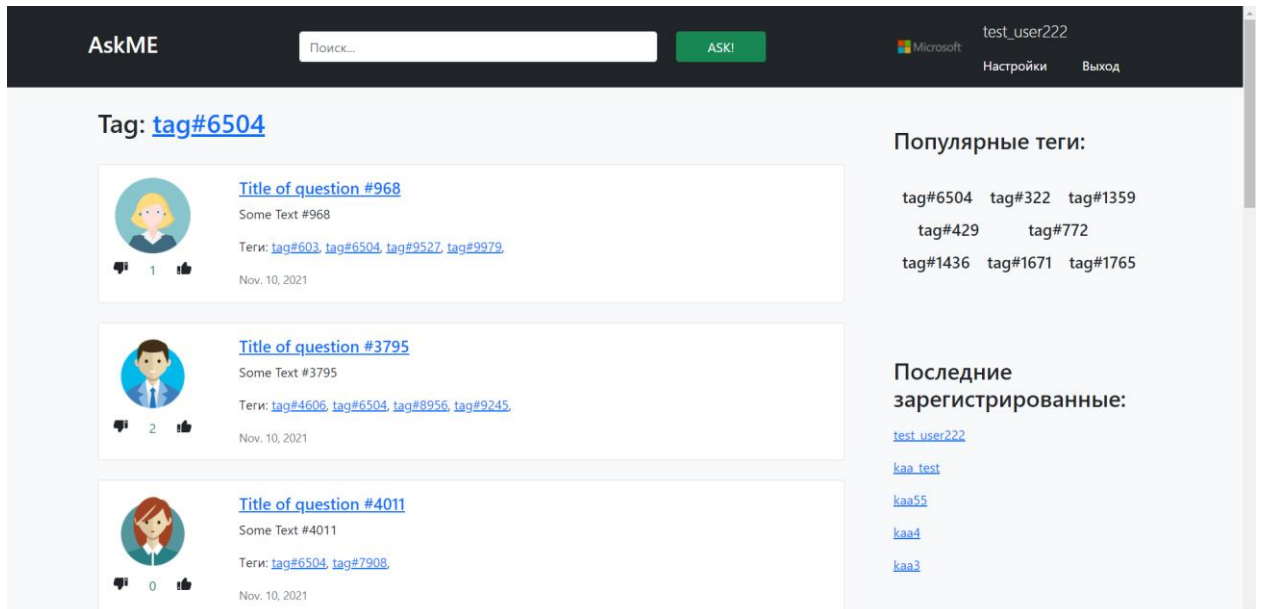
Настройки:



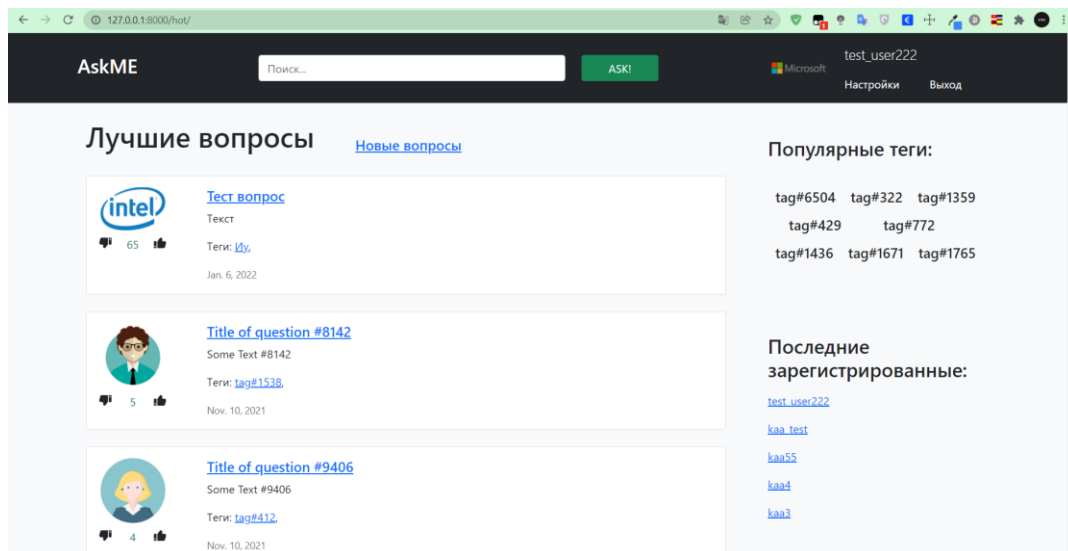
Создание вопроса:



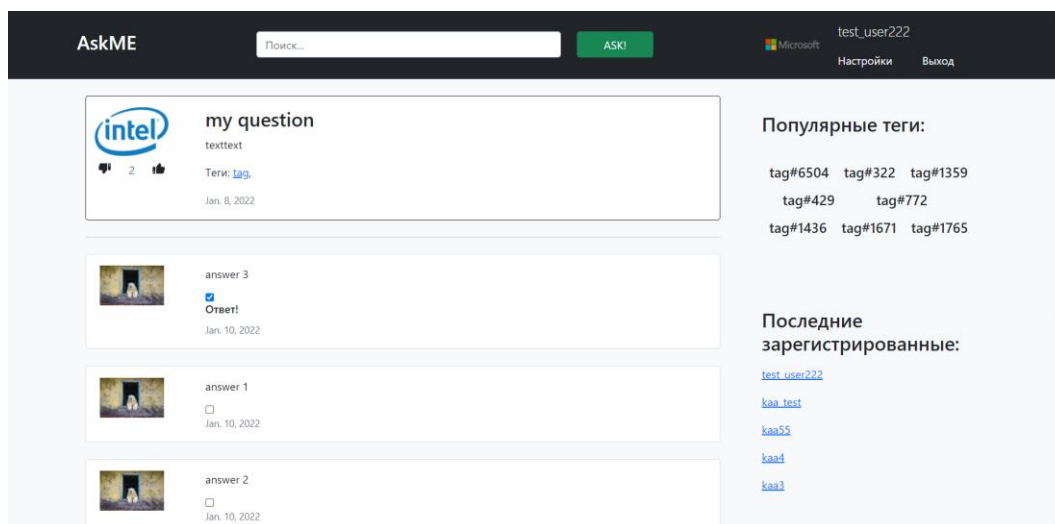
Переход по тегу:



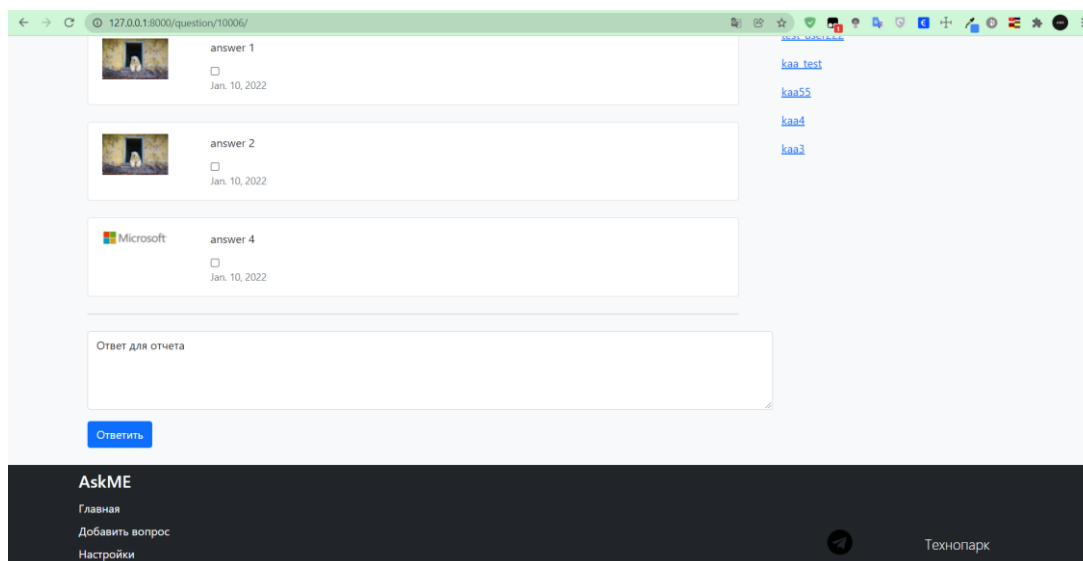
Лучшие вопросы:

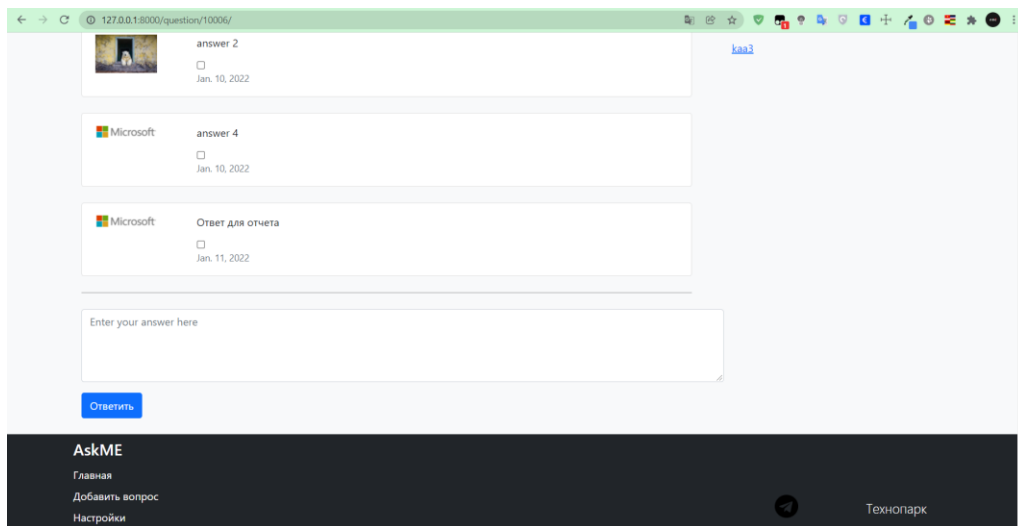


Страница вопроса:

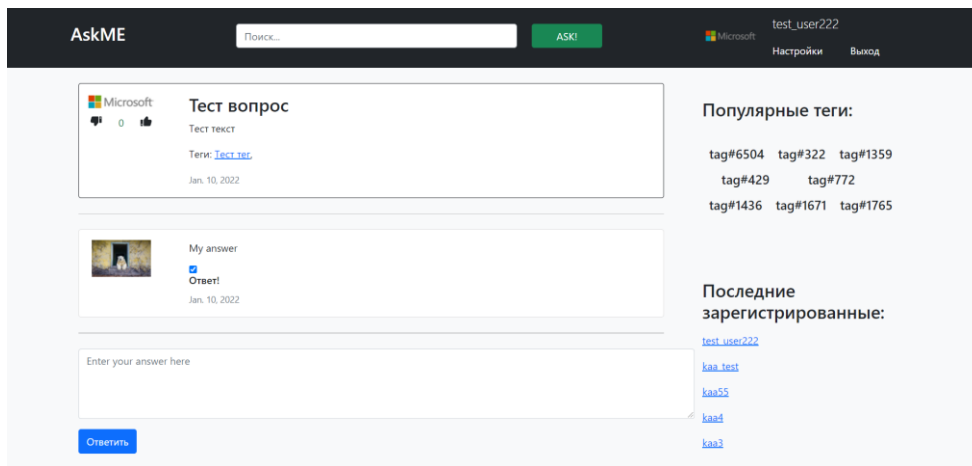
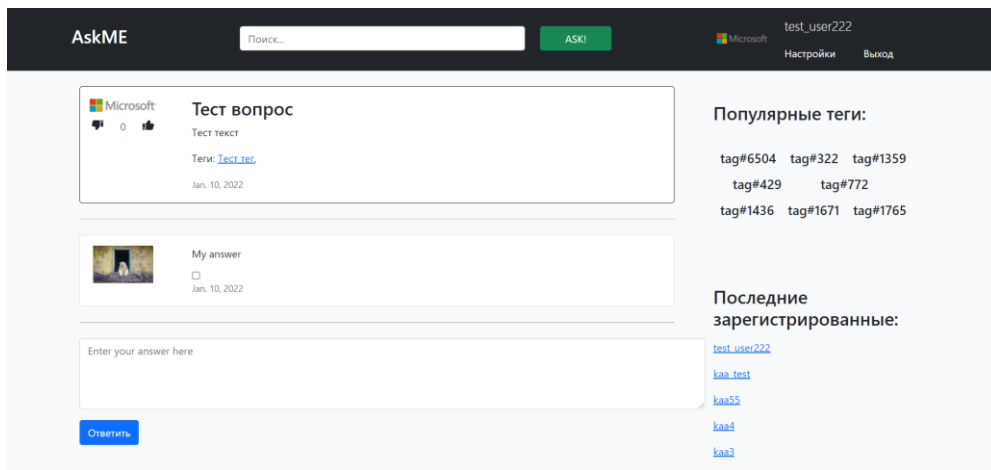


Ввод ответа:





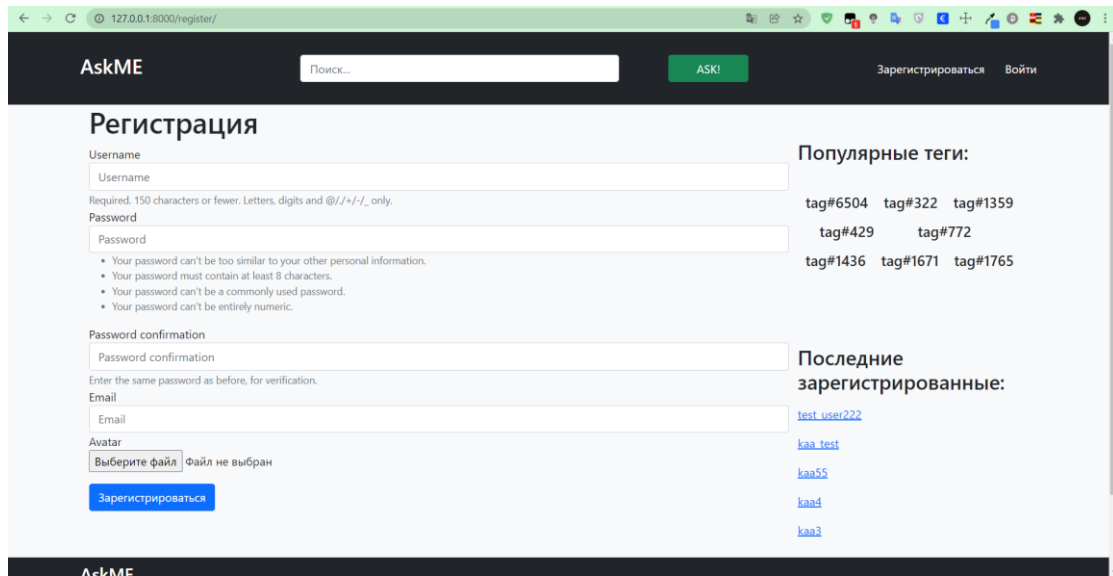
Пометка корректного ответа:



«Лайк» вопросу:



Регистрация:



The screenshot shows a web browser window with the URL 127.0.0.1:8000/register/. The page has a dark header with the 'AskME' logo, a search bar, and links for 'Зарегистрироваться' and 'Войти'. The main content area is titled 'Регистрация' and contains several form fields: 'Username', 'Password', 'Password confirmation', 'Email', and 'Avatar'. The 'Password' field has a list of requirements. The 'Avatar' field has a file selection button. A blue 'Зарегистрироваться' button is at the bottom left. On the right, there are two sections: 'Популярные теги:' with a grid of tags and 'Последние зарегистрированные:' with a list of user links.

AskME

Поиск...

ASK!

Зарегистрироваться Войти

Регистрация

Username

Username

Required: 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password

Password

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation

Password confirmation

Enter the same password as before, for verification.

Email

Email

Avatar

Выберите файл | Файл не выбран

Зарегистрироваться

Популярные теги:

tag#6504 tag#322 tag#1359
tag#429 tag#772
tag#1436 tag#1671 tag#1765

Последние зарегистрированные:

[test_user222](#)
[kaa_test](#)
[kaa55](#)
[kaa4](#)
[kaa3](#)

Вывод: выполнено домашнее задание, разработан сайт «Вопрос-Ответ» и база данных для него.