Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

‹‹Уральский федеральный университет имени первого Президента России

Б.Н.Ельцина››

Кафедра ‹‹Интеллектуальных информационных технологий››

дисциплина ‹‹WEB программирование››

Курсовая работа

«Платформа для ведения блога»

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_ Свинцов Д. В.

(подпись преподавателя)

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_ Русанов Е. А.

(подпись студентов)

Группа: ФО-321001

Екатеринбург

2015

# Введение

Все мои блоги, которые я веду в интернете, всегда написаны мной самостоятельно. Периодически, узнавая новые технологии, а зачастую и для того, чтобы изучить на практике новую для меня технологию, я пишу новую платформу для ведения блога. Поэтому разработка этого сайта и стала темой моей курсовой работы по веб программированию.

# Описание проекта

Сайт состоит из нескольких страниц:

Главная – содержит список всех постов;

Раздел – содержат посты определённой категории

Пост – одержит полную версию поста, комментарии к нему и форму для добавления нового комментария;

Страница – содержит информацию, которая не входит в концепцию поста.

# Обоснование проекта

Так как создание веб-приложений на языке программирования Python является новым, я решил изучать эту возможность, как обычно, создавая платформу для ведения блога.

# Инструкция пользователя

Сайт состоит из главной страницы, с которой можно перейти на любой пост или на страницу раздела. Также на главной странице справа расположено меню, с помощью которого можно перейти на страницы рассказывающие об услуга или контактах.

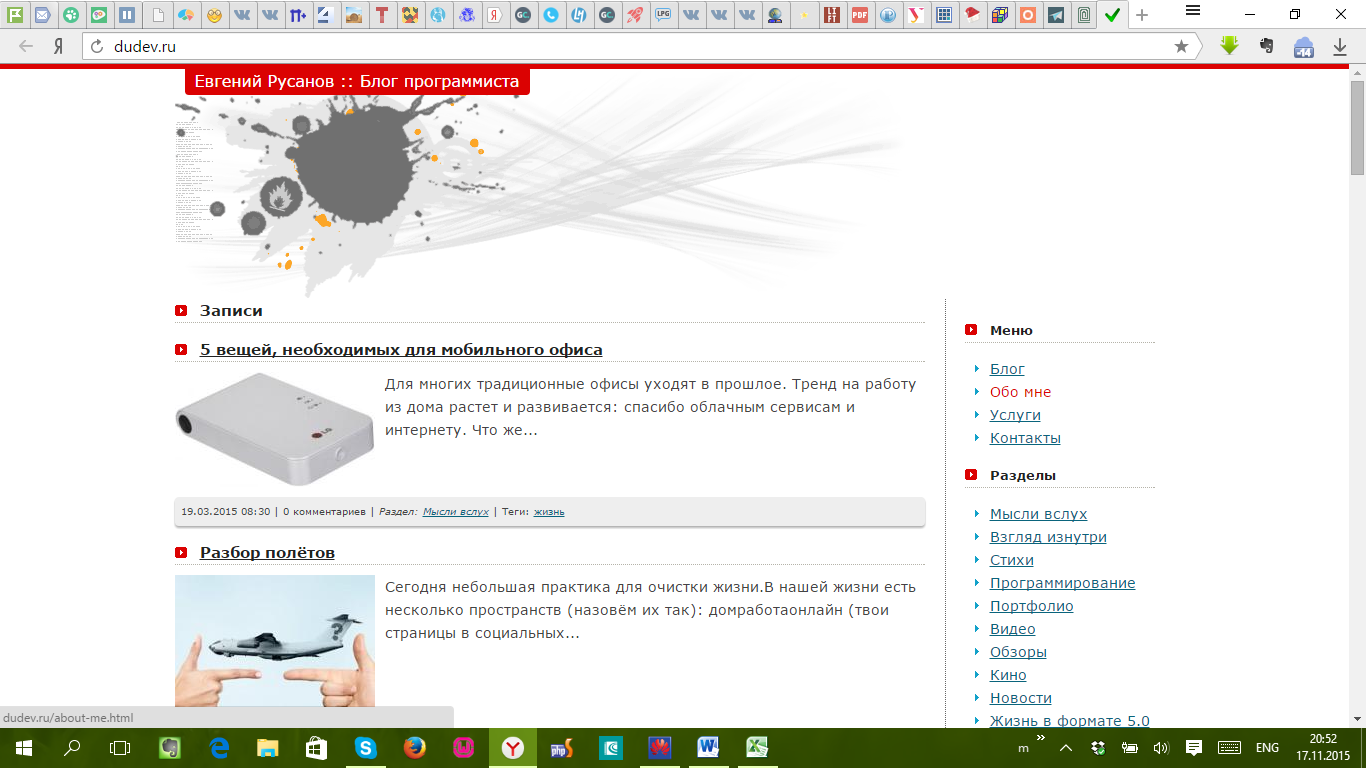


Рис 1. Вид сайта

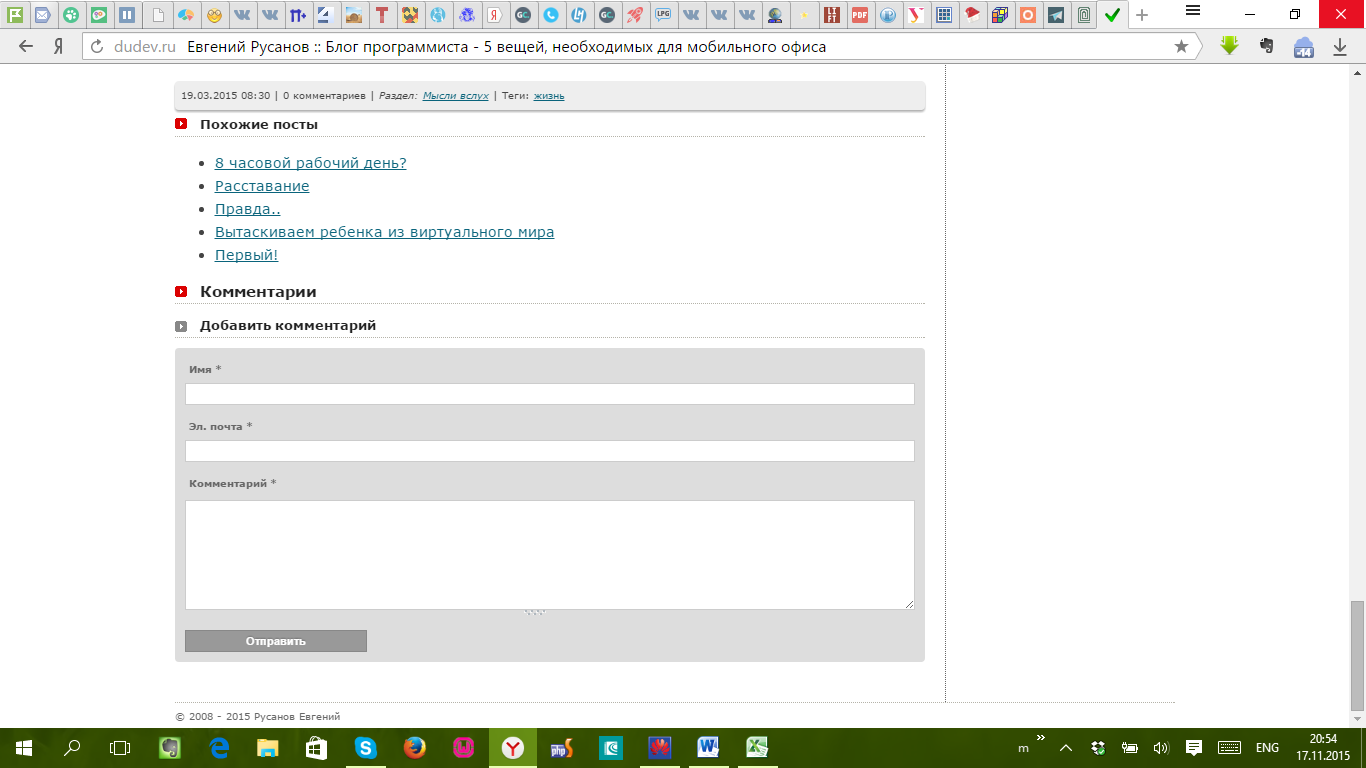


Рис 2. Форма добавления комментария

# Инструкция программиста

Установка

Для установки требуется версия Python 2.7, наличие установленных пакетов Jinja2 и PyramidFramework.

Открываем командную строку и переходим в корневую папку проекта. Затем прописываем команду:

python setup.py develop

Инициализация баз данных

Для инициализации баз данных прописываем в командной строке следующую команду:

initializedb.py development.ini

Запуск проекта

Запуск проекта производится следующей командой:

pserve development.ini

Описание ключевых функций, классов, блоков кода

В файл \_init\_.py были добавлены роуты для страниц сайта, а также настройки sacrud для работы с базой данных.

# coding: utf8

from pyramid.config import Configurator

from sqlalchemy import engine\_from\_config

from .models import (

DBSession,

Post,

Category,

Page,

Comment,

Base

)

def main(global\_config, \*\*settings):

""" This function returns a Pyramid WSGI application.

"""

engine = engine\_from\_config(settings, 'sqlalchemy.')

DBSession.configure(bind=engine)

Base.metadata.bind = engine

config = Configurator(settings=settings)

config.include('pyramid\_jinja2')

config.add\_static\_view('static', 'static', cache\_max\_age=3600)

config.add\_route('blog', '/')

config.add\_route('post', '/blog/{id}')

config.add\_route('page', '/page/{nick}')

config.include('pyramid\_sacrud', route\_prefix='admin')

settings = config.registry.settings

settings['pyramid\_sacrud.models'] = (

('Tables', [Post,Category,Page,Comment]),)

config.scan()

return config.make\_wsgi\_app()

В файле models.py описаны таблицы базы данных: Post, Category, Page, Comment.

from sqlalchemy import (

Column,

Integer,

Text,

ForeignKey,

String,

DateTime,

func

)

from sqlalchemy.ext.declarative import declarative\_base

from sqlalchemy.orm import (

scoped\_session,

sessionmaker,

relationship

)

from zope.sqlalchemy import ZopeTransactionExtension

DBSession = scoped\_session(sessionmaker(extension=ZopeTransactionExtension()))

Base = declarative\_base()

class Category(Base):

\_\_tablename\_\_ = 'category'

id = Column(Integer, primary\_key=True)

name = Column(String(255), nullable=False)

post = relationship("Post")

class Comment(Base):

\_\_tablename\_\_ = 'comment'

id = Column(Integer, primary\_key=True)

content = Column(Text, nullable=False)

email = Column(String(255))

author = Column(String(255), nullable=False)

create\_time = Column(DateTime, default=func.now(), nullable=False)

post\_id = Column(Integer, ForeignKey('post.id'), nullable=False)

class Page(Base):

\_\_tablename\_\_ = 'page'

id = Column(Integer, primary\_key=True)

title = Column(String(255), nullable=False)

nick = Column(String(255), unique=True, nullable=False)

content = Column(Text, nullable=False)

class Post(Base):

\_\_tablename\_\_ = 'post'

id = Column(Integer, primary\_key=True)

title = Column(String(255), nullable=False)

content = Column(Text, nullable=False)

public\_time = Column(DateTime, default=func.now(), nullable=False)

category\_id = Column(Integer, ForeignKey('category.id'), nullable=False)

comment = relationship("Comment", backref="post")

category = relationship("Category")

Файл views.py содержит представления страниц сайта.

# coding: utf8

from pyramid.response import Response

from pyramid.view import view\_config

from sqlalchemy.exc import DBAPIError

from .models import (

DBSession,

Post,

Category,

Page,

Comment

)

@view\_config(route\_name='post', request\_method='GET', renderer='templates/post.jinja2')

def post\_view(request):

post\_id = request.matchdict['id']

post = DBSession.query(Post).filter\_by(id=post\_id).first()

comments = DBSession.query(Comment).filter\_by(post\_id=post\_id).all()

categories = DBSession.query(Category).all()

if not post:

return Response('Not found')

return {'post' : post, 'comments' = comments, 'categories' = categories, 'report': 0}

@view\_config(route\_name='post', request\_method='POST', renderer='templates/post.jinja2')

def post\_comment(request):

post\_id = request.matchdict['id']

post = DBSession.query(Post).filter\_by(id=post\_id).first()

try:

author = request.POST['author']

email = request.POST['email']

content = request.POST['content']

if not (subject and email and body):

return {'report': 2}

comment = Comment(

content = content

email = email

author = author

post\_id = post\_id

)

except Exception as ex:

return {'report': 2}

comments = DBSession.query(Comment).filter\_by(post\_id=post\_id).all()

categories = DBSession.query(Category).all()

return {'post' : post, 'comments' = comments, 'categories' = categories, 'report': 1}

@view\_config(route\_name='blog', renderer='templates/blog.jinja2')

def blog\_view(request):

categories = DBSession.query(Category)

category = None

if 'cat' in request.GET:

category\_id = request.GET['cat']

category = categories.filter\_by(id=category\_id).first()

if category:

posts = category.post

else:

posts = DBSession.query(Post).all()

return {'posts': posts,

'categories': categories}

@view\_config(route\_name='page', renderer='templates/page.jinja2')

def page\_view(request):

page\_nick = request.matchdict['nick']

page = DBSession.query(Page).filter\_by(id=page\_id).first()

categories = DBSession.query(Category).all()

if not post:

return Response('Not found')

return {'page' : page, 'categories' = categories}

# Выводы

Во время разработки сайта в проекте были использованы следующие технологии:

* Фреймворк Pyramid
* Шаблонизатор Jinja2
* ОРМ SQLAlchemy
* MySQL

В процессе работы я научился создавать сайты при помощи фреймворка Pyramid с использованием шаблонов, а также подключать к проекту базу данных MySQL и работать с ней.