*Тема 4.4. Проект «Администрирование веб-сайта»* **(1+5=6час)**

***Лекционное занятие 1 (2 часа)***

Программирование представляют собой современную дисциплину, которая призвана решать актуальные задачи и получать практические навыки в различных сферах человеческой деятельности. При выборе темы проекта по программированию можно придерживаться рекомендаций, но можно также их дополнять и изменять.

Примерный план проекта по программированию имеет вид:

1. Цели и задачи проекта. Язык программирования и платформа проекта.
2. Изучение языка программирования, если необходимо.
3. Построение структурных и логических схем компонентов и всего проекта.
4. Разработка алгоритмов и интерфейса (программирование+дизайн проекта).
5. Тестирование, дебаггинг.
6. Оформление проекта для отчета. Здесь нужно выделить 4 вопроса:
* Что собой представляет этот проект?
* Что собой представляет MVP (минимально жизнеспособный продукт)?
* Какая «посылка» проекта, то есть – зачем он нужен (отличие от аналогов)?
* Когда проект можно считать законченным?

Пример ответов на данные вопросы при разработке проекта на тему «Калькулятор»:

1. Калькулятор с операциями «+», «-», «\*», «:». Реализован на языке Python.
2. MVP – калькулятор с операциями «+», «-», «\*», «:», вводом и выводом данных.
3. Посылка проекта – стильный интерфейс и дополнительный набор операций. Проект завершен по реализации функций калькулятора и оформления документации.

Второй этап ‑ создание рабочего процесса и системы управления проектом. Можно использовать методологию Kanban – визуализации «на доске» продвижения реализаций компонентов проекта, что очень удобно для наблюдения и анализа процессов, но трудно в обеспечении контроля сроков завершения проекта:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Что сделать | В процессе | Уже сделано | Баги, проблемы | Тестирование | … |
|  |  |  |  |  |  |

Третий этап ‑ разбивка проекта на подзадачи (строки на «доске»).

Четвертый этап ‑ реализация подзадач.

Kanban – гибкая технология, где можно усовершенствовать первоначальный план по каждому компоненту в соответствии со следующими базовыми принципами:

* Люди важнее инструментов;
* Качество продукта важнее качества документации;
* Взаимодействие с заказчиком важнее контракта;
* Готовность к изменениям важнее установленного плана.

Здесь хорошо видно, почему технология так популярна в IT, где ысё быстро меняется.

***Практическое занятие 1 (1 час)***

Разработать веб-приложение, используя Django. Примерный список тем:

|  |  |  |
| --- | --- | --- |
| 1. Мир хобби
 | 1. Мой родной город
 | 1. Они изменили мир IT
 |
| 1. Онлайн-библиотеки
 | 1. Доска объявлений
 | 1. Любимая музыка
 |

Требования: 1)Не менее 4 страниц; 2)Использование инструментов Django;

3)Реализация идентификации и регистрации пользователя.

**Отчет по 4.4.1:** *doc-файл с 1)темой проекта 2)назначением и структурой рабочих страниц*

***Пример отчета:*** *1)«Доска объявлений»; 2)Страницы:* ***about.html****-описание+меню,* ***login.html*** *– вход в систему,* ***register.html*** *– регистрация нового пользователя;* ***home.html*** *– интерактивная по пользвателю и объявлению доска объявлений+меню.*

***Практическое занятие 2 (1 час)****Проект: «Доска объявлений» (пример)*

*Предполагается наличие всех предыдущих компонентов, но критично только отсутствие account.*

1. Перейдем в каталог с головным проектом (в примере – djangoProject).
2. Создадим новое приложение mysite: *python manage.py startapp mysite*

|  |  |
| --- | --- |
| Django создаст папку mysite с содержимым, как на Рис.1:1. Откроем djangoProject/settings.py, добавив этот код:

*INSTALLED\_APPS = [ 'django.contrib.admin',* *'django.contrib.auth', 'django.contrib.contenttypes',* *'django.contrib.sessions','django.contrib.messages',* *'django.contrib.staticfiles', 'hello', 'web', 'account', 'mysite']*1. В каталоге mysite создадим папку templates, где будем хранить шаблоны страниц приложения. Создадим в templates файл base.html со следующим кодом:
 |  |

*<!doctype html><head> <meta charset="utf-8">*

*<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">*

*{% load static %}*

*<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">*

*<link href='http://fonts.googleapis.com/css?family=Roboto:400, 100, 100italic, 300, 300italic, 400italic, 500, 500italic, 700, 700italic, 900italic, 900' rel='stylesheet' type='text/css'>*

*<link href='https://unpkg.com/boxicons@2.1.2/css/boxicons.min.css' rel='stylesheet'>*

*<title> {% block title %} {% endblock %} </title></head>*

*<body>{% block content %}{% endblock %}</body></html>*

|  |  |
| --- | --- |
| **Отчет по 4.4.2:** *doc – файл со скриншотом base.html (здесь фрагмент)* |  |

***Практическое занятие 3 (1 час)***

1. Создадим в templates файлы home.html, about.html, login.html, register.html:

**home.html**

*{% extends "base.html" %}*

*{% block title %} Главная страница {% endblock %}*

*{% block content %}*

 *<p align="left"><font size="5" color="#6495ed"><b>Доска объявлений... </b></font></p>*

*{% block menu %}*

 *<header class="site-header">*

 *<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">*

 *<div class="container" align="right"><p font size="4"><i>*

 *<a class="nav-item nav-link" href="{% url 'about' %}">О проекте</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'login' %}">Вход</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'register' %}">Регистрация</a>*

 *</i>*

 *</p>*

 *</div>*

 *</nav>*

*</header>*

 *{% endblock %}*

 *{% for post in posts %} <article class="media content-section">*

 *<div class="media-body"><div class="article-metadata">*

 *<a class="mr-2" href="#">{{ post.author }}</a>*

 *<small class="text-muted">{{ post.date\_posted }}</small> </div>*

 *<h2><a class="article-title" href="#">{{ post.title }}</a></h2>*

 *<p class="article-content">{{ post.content }}</p></div></article>*

 *{% endfor %}*

*{% endblock content %}*

**about.html**

*{% extends "base.html" %}*

*{% block content %}*

 *{% block menu %}*

 *<header class="site-header">*

 *<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">*

 *<div class="container" align="right"><p font size="4"><i>*

 *<a class="nav-item nav-link" href="{% url 'home' %}">Главная</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'login' %}">Вход</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'register' %}">Регистрация</a>*

 *</i></p>*

 *</div> </nav> </header>*

 *{% endblock %}*

 *<h3>Здесь краткий рассказ о проекте…</h3>*

*{% endblock content %}*

**login.html**

*{% extends "base.html" %}{% block title %} Страница входа{% endblock %}*

*{% block content %}<h1>Вход в систему</h1>*

*<p>Используйте следующую форму входа в систему:</p><form action="." method="post">*

*{{form.as\_p}}{%csrf\_token%}<p><input type="submit" value="Вход"></p></form>{% endblock %}*

**register.html**

*{% extends "base.html" %}{% block title %}Регистрация{% endblock %}{% block content %}*

*<h1>Регистрация нового пользователя</h1><form action="." method="post">{{ form.as\_p }}*

*{% csrf\_token %}<p><input type="submit" value="Cоздать аккаунт"></p></form>{% endblock %}*

1. Далее необходимо создать файл mysite/forms.py

*from django import forms*

*class LoginForm(forms.Form):*

 *username = forms.CharField(); password = forms.CharField(widget=forms.PasswordInput)*

1. Зададим представления в mysite/views.py тестовых данных через словари ***posts***:

*from django.http import HttpResponse*

*from django.shortcuts import render*

*from django.contrib.auth import authenticate, login*

*from django.contrib.auth.forms import UserCreationForm*

*from django.urls import reverse\_lazy*

*from django.views import generic*

*from .forms import LoginForm*

*posts=[{ 'author': 'User1', 'title':'Объявление №1', 'content':'Содержание1', 'date\_posted':'10.04.2023'},*

*{ 'author': 'User2', 'title':'Объявление №2', 'content':'Содержание2', 'date\_posted': '10.04.2023'},*

*{'author': 'User1', 'title':'Объявление №3','content': 'Содержание3', 'date\_posted': '7.04.2023'}]*

*def home(request): context = {'posts': posts}; return render(request, 'home.html', context)*

*def about(request): return render(request, 'about.html', {'title': 'О проекте'})*

*def user\_login(request):*

 *if request.method == 'POST':*

 *form = LoginForm(request.POST)*

 *if form.is\_valid():*

 *cd = form.cleaned\_data*

 *user = authenticate(request, username=cd['username'], password=cd['password'])*

 *if user is not None:*

 *if user.is\_active: login(request, user); return HttpResponse('Authenticated successfully')*

 *else: return HttpResponse('Disabled account')*

 *else: return HttpResponse('Invalid login')*

 *else: form = LoginForm(); return render(request, 'login.html', {'form': form})*

*class user\_register(generic.CreateView):*

 *form\_class = UserCreationForm; success\_url = reverse\_lazy('login'); template\_name = 'register.html'*

Откроем файл urls.py в djangoProject и добавим следующий код:

*from django.contrib import admin*

*from django.urls import path, include*

*urlpatterns=[path('admin/',admin.site.urls),path('',include('hello.urls')),path('web/',include('web.urls')), path('account/',include('account.urls')),path('mysite/',include('mysite.urls')),]*

Далее создадим файл urls.py в mysite со следующим кодом:

*from django.urls import path*

*from . import views*

*from .views import user\_register*

*urlpatterns = [ path('', views.home, name='home'), path('login/', views.user\_login, name='login'),*

*path('register/', user\_register.as\_view(), name='register'), path('about/', views.about, name='about'),]*

1. Запустив: *python manage.py runserver* и перейдя в 127.0.0.1:8000/mysite, увидим:

|  |  |
| --- | --- |
| Рис. 1. Страница home.html |  Рис. 2. Страница login.html |
|  Рис. 3. Страница register.html |  Рис. 4. Страница about.html |

Эти четыре скриншота – **отчет по практике 4.4.3**

***Практическое занятие 4 (2 часа) – защита проекта***

Это пример оценочного листа проекта: (выберем - по 10-балльной шкале)

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерий оценивания** | **Балл** | **Критерий оценивания** | **Балл** |
| 1.Актуальность темы  |  | 5.Дизайн интерфейса |  |
| 2.Соответствие теме |  | 6.Степень разработанности |  |
| 3.Сложность программы |  | 7.Применение в другихх задачах |  |
| 4.Оригинальность алгоритма |  | 8.Итоговое количество баллов |  |

Web-приложение оценивают по следующим критериям:

|  |  |  |  |
| --- | --- | --- | --- |
| **Диапазон применения** | Балл | **Структура и навигация** | Балл |
| На всех платформах и браузерах  |  | Правильность организации содержания |  |
| Адаптивность к разным разрешениям |  | Наличие форм и их логическая связь |  |
| Работоспособность всех компонентов  |  | Страница для 404 ошибки |  |
| Актуальность для целевой аудитории |  | Понятность назначения каждой страницы |  |
| **Контент (содержание)** |  | Проверка на пустые страницы или разделы |  |
| Соответствие назначению |  | Взаимоподобие навигации всех страниц |  |
| Актуальность и полезность |  | Работоспособность ссылок |  |
| Достоверность информации |  | Легкость использования навигации |  |
| Понятность информации и терминов |  | Подсказки к активным компонентам |  |
| Орфографические ошибки в тексте |  | Наличие ссылок обратной связи |  |
| Поисковик информации (если нужно) |  | Возврат на прежний уровень |  |
| **Внешний вид** |  |
| Единство стилей по цвету/шрифту/графике |  | Читаемость текста |  |
| Сочетаемость графики с другими элементами |  | Понятность назначения сайта |  |

**Отчет по 4.4.4:** - doc-файл с самооценкой по этим оценочным листам.

Для итоговой аттестации необходим архив со следующими компонентами проекта:

1.Учетный лист проекта следующего вида (пример):

|  |
| --- |
| ***Проект по программированию на тему «Электронная доска объявлений»***1. **Автор:** Иванов Иван Иванович
2. **Аннотация:** «Разработана интерактивная интернет-страница для реализации электронной доски объявлений с регистрацией и аутентификацией пользователей»
3. **Содержание:** Продукт содержит 4 рабочих страницы, включающие описание системы, формы для регистрации и входа, и головную страницу с базовой информацией и системой меню. Базовая информация, помимо текстов объявлений и авторов, содержит гиперссылки на авторов и объявления, что обеспечивает получение более полной информации.
4. **Ограничения продукта**:
	1. *Возможности использования*: при наличии предустановленного пакета Django
	2. *Характеристики интерфейса:* html-файлы с меню, формами регистрации и входа
5. **Формализация алгоритма:**

Помимо модулей, сформированных Django, продукт включает добавленные подпрограммы вызова нужных web-страниц и форм, а также разработанные для этого html-страницы.1. **Листинг программы:** см. Приложение 1 (тексты используемых модулей программы).
2. **Скриншоты работы программы:** см. Приложение 2
 |

2. Листинг программы (Приложение 1):

|  |  |
| --- | --- |
| a)Дерево (или список) папок приложения Здесь не включены папки, которые также присутствуют в Django-проекте, но к работе Продукта отношения не имеют. Папка **venv** создана приложением Django и содержит необходимые для работы Продукта скрипты, которые являются библиотечными, а потому в листинге не представлены, но подпапка djangoProject содержит файлы, редактируемые при создании Продукта, а потому вошедшие в листинг. | djangoProject├account│ └───templates├djangoProject├mysite│ └───templates└───venv |
| b)Здесь представлено дерево папок и файлов компонента Account. Листинги файлов будут представлены ниже. | ACCOUNT│ forms.py│ urls.py│ views.py└─templates base.html login.html profile.html register.html |
| c)Здесь представлено дерево файлов головной папки проекта Django, где удалены все файлы, не изменяемые при разработке Продукта.Листинги файлов представлены ниже | DJANGOPROJECT│ settings.py│ urls.py |
| d)Основной продукт – «Электронная доска объявлений».Листинги всех файлов будут представлены ниже | MYSITE│ forms.py│ urls.py│ views.py└─templates about.html base.html home.html login.html register.html |

**Account**

forms.py

*from django import forms*

*class LoginForm(forms.Form):*

 *username = forms.CharField();*

 *password = forms.CharField(widget=forms.PasswordInput)*

urls.py

*from django.urls import path*

*from account import views*

*from account.views import Register*

*from django.contrib.auth import views as auth\_views*

*from . import views as user\_views*

*urlpatterns = [ path('login/', views.user\_login, name='login'),*

 *path('login/', auth\_views.LoginView.as\_view(template\_name='login.html'), name='login'),*

 *path('register/', Register.as\_view(), name='register'),*

 *path('profile/', user\_views.profile, name='profile'),]*

views.py

*from django.http import HttpResponse*

*from django.shortcuts import render*

*from django.contrib.auth import authenticate, login*

*from django.contrib.auth.forms import UserCreationForm*

*from django.urls import reverse\_lazy*

*from django.views import generic*

*from .forms import LoginForm*

*from django.contrib.auth.decorators import login\_required*

*class Register(generic.CreateView):*

 *form\_class = UserCreationForm*

 *success\_url = reverse\_lazy('login')*

 *template\_name = 'register.html'*

*def user\_login(request):*

 *if request.method == 'POST':*

 *form = LoginForm(request.POST)*

 *if form.is\_valid():*

 *cd = form.cleaned\_data*

 *user = authenticate(request, username=cd['username'], password=cd['password'])*

 *if user is not None:*

 *if user.is\_active:*

 *login(request, user)*

 *return HttpResponse('Authenticated successfully')*

 *else: return HttpResponse('Disabled account')*

 *else: return HttpResponse('Invalid login')*

 *else:*

 *form = LoginForm(); return render(request, 'login.html', {'form': form})*

*@login\_required*

*def profile(request): return render(request, 'profile.html')*

templates/register.html

*{% extends "base.html" %}{% block title %}Регистрация{% endblock %}*

*{% block content %} <h1>Регистрация нового пользователя</h1>*

*<form action="." method="post"> {{ form.as\_p }} {% csrf\_token %}*

*<p><input type="submit" value="Cоздать аккаунт"></p>*

*</form>*

*{% endblock %}*

templates/profile.html

*{% extends "base.html" %}*

*{% block content %}*

 *<div class="content-section">*

 *<div class="media-body"><h2 class="account-heading">{{ user.username }}</h2>*

 *<p class="text-secondary">{{ user.email }}</p> </div>*

 *</div>*

*{% endblock content %}*

templates/base.html

*<!doctype html>*

*<html lang="en">*

*<head>*

 *<meta charset="utf-8"> <title>{% block title %}{% endblock %}</title>*

 *{% load static %}*

 *<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD" crossorigin="anonymous">*

*</head>*

*<body>*

 *<div id="header"> <span class="logo"></span> </div>*

 *<div id="content"> {% block content %}{% endblock %} </div>*

*</body>*

*</html>*

**djangoProject**

urls.py

*from django.contrib import admin*

*from django.urls import path,include*

*urlpatterns = [ path('admin/', admin.site.urls),*

*path('account/', include('account.urls')), path('mysite/', include('mysite.urls')),]*

settings.py

*from pathlib import Path*

*LOGIN\_REDIRECT\_URL = 'profile'; LOGIN\_URL = 'login'*

*BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent*

*SECRET\_KEY = 'django-insecure-r!c!5mec)a3se0i4xk\_5o#666-k3-$372i2l49%gan+nt$zy78'*

*DEBUG = True; ALLOWED\_HOSTS = []*

*INSTALLED\_APPS = [*

 *'django.contrib.admin', 'django.contrib.auth', 'django.contrib.contenttypes', 'django.contrib.sessions',*

 *'django.contrib.messages', 'django.contrib.staticfiles', 'account','mysite']*

*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 = 'djangoProject.urls'*

*TEMPLATES = [*

 *{ 'BACKEND': 'django.template.backends.django.DjangoTemplates',*

 *'DIRS': [BASE\_DIR / '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 = 'djangoProject.wsgi.application'*

*DATABASES = {'default': {'ENGINE': 'django.db.backends.sqlite3', 'NAME': BASE\_DIR / 'db.sqlite3', }}*

*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', },]*

*LANGUAGE\_CODE = 'en-us'; TIME\_ZONE = 'UTC'; USE\_I18N = True; USE\_TZ = True*

*STATIC\_URL = 'static/'; DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'*

**mysite**

urls.py

*from django.urls import path*

*from . import views*

*from .views import user\_register*

*urlpatterns = [ path('', views.home, name='home'), path('login/', views.user\_login, name='login'),*

*path('register/', user\_register.as\_view(), name='register'), path('about/', views.about, name='about'),]*

forms.py

*from django import forms*

*class LoginForm(forms.Form):*

 *username = forms.CharField();*

 *password = forms.CharField(widget=forms.PasswordInput)*

views.py

*from django.http import HttpResponse*

*from django.shortcuts import render*

*from django.contrib.auth import authenticate, login*

*from django.contrib.auth.forms import UserCreationForm*

*from django.urls import reverse\_lazy*

*from django.views import generic*

*from .forms import LoginForm*

*posts=[{ 'author': 'User1', 'title':'Объявление №1', 'content':'Содержание1', 'date\_posted':'10.04.2023'},*

*{ 'author': 'User2', 'title':'Объявление №2', 'content':'Содержание2', 'date\_posted': '10.042023'},*

*{'author': 'User1', 'title':'Объявление №3','content': 'Содержание3', 'date\_posted': '07.042023'}]*

*def home(request): context = {'posts': posts}; return render(request, 'home.html', context)*

*def about(request): return render(request, 'about.html', {'title': 'О проекте'})*

*def user\_login(request):*

 *if request.method == 'POST':*

 *form = LoginForm(request.POST)*

 *if form.is\_valid():*

 *cd = form.cleaned\_data*

 *user = authenticate(request, username=cd['username'], password=cd['password'])*

 *if user is not None:*

 *if user.is\_active: login(request, user); return HttpResponse('Authenticated successfully')*

 *else: return HttpResponse('Disabled account')*

 *else: return HttpResponse('Invalid login')*

 *else: form = LoginForm(); return render(request, 'login.html', {'form': form})*

*class user\_register(generic.CreateView):*

 *form\_class = UserCreationForm; success\_url = reverse\_lazy('login'); template\_name = 'register.html'*

**templates**/login.html

*{% extends "base.html" %}{% block title %} Страница входа{% endblock %}*

*{% block content %} <h1>Вход в систему</h1>*

 *<p>Используйте следующую форму входа в систему:</p>*

 *<form action="." method="post"> {{ form.as\_p }} {% csrf\_token %}*

 *<p><input type="submit" value="Вход"></p> </form>*

*{% endblock content %}*

**templates**/register.html

*{% extends "base.html" %}*

*{% block title %} Регистрация {% endblock %}*

*{% block content %}*

 *<h1>Регистрация нового пользователя</h1>*

 *<form action="." method="post">*

 *{{ form.as\_p }} {% csrf\_token %}*

 *<p><input type="submit" value="Cоздать аккаунт"></p>*

 *</form>*

*{% endblock %}*

**templates**/base.html

*<!doctype html><head> <meta charset="utf-8">*

*<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">*

*{% load static %}*

*<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">*

*<link href='http://fonts.googleapis.com/css?family=Roboto:400,100,100italic,300,300italic,400italic,500,500italic,700,700italic,900italic,900' rel='stylesheet' type='text/css'>*

*<link href='https://unpkg.com/boxicons@2.1.2/css/boxicons.min.css' rel='stylesheet'>*

 *<title> {% block title %} {% endblock %} </title>*

*</head><body>{% block content %}{% endblock %}</body>*

**templates**/home.html

*{% extends "base.html" %}*

*{% block title %} Главная страница {% endblock %}*

*{% block content %}*

 *<p align="left"><font size="5" color="#6495ed"><b>Доска объявлений... </b></font></p>*

*{% block menu %}*

 *<header class="site-header">*

 *<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">*

 *<div class="container" align="right"><p font size="4"><i>*

 *<a class="nav-item nav-link" href="{% url 'about' %}">О проекте</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'login' %}">Вход</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'register' %}">Регистрация</a>*

 *</i> </p>*

 *</div>*

 *</nav>*

 *</header>*

*{% endblock %}*

 *{% for post in posts %} <article class="media content-section">*

 *<div class="media-body"><div class="article-metadata">*

 *<a class="mr-2" href="#">{{ post.author }}</a>*

 *<small class="text-muted">{{ post.date\_posted }}</small> </div>*

 *<h2><a class="article-title" href="#">{{ post.title }}</a></h2>*

 *<p class="article-content">{{ post.content }}</p></div></article>*

 *{% endfor %}*

*{% endblock content %}*

**templates**/about.html

*{% extends "base.html" %}*

*{% block content %}*

 *{% block menu %}*

 *<header class="site-header">*

 *<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">*

 *<div class="container" align="right"><p font size="4"><i>*

 *<a class="nav-item nav-link" href="{% url 'home' %}">Главная</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'login' %}">Вход</a>&nbsp &nbsp*

 *<a class="nav-item nav-link" href="{% url 'register' %}">Регистрация</a>*

 *</i></p>*

 *</div>*

 *</nav>*

 *</header> {% endblock %}*

 *<h3>Здесь краткий рассказ о проекте…</h3>*

*{% endblock content %}*

3. Скриншоты работы Продукта (Приложение 2):

|  |  |
| --- | --- |
| home.html |  login.html |
|  register.html |  about.html |

4. Презентация Продукта (здесь не показана, т.к. это отдельный файл, добавляемым в архив).