**Лабораторная работа № 5**

**Тема:**

1. Веб-программирование;
2. Веб-фреймворк Django

**Цель работы:**

Закрепить знания и навыки по работе со стандартными библиотеками, фреймворком Django.

**Задание.**

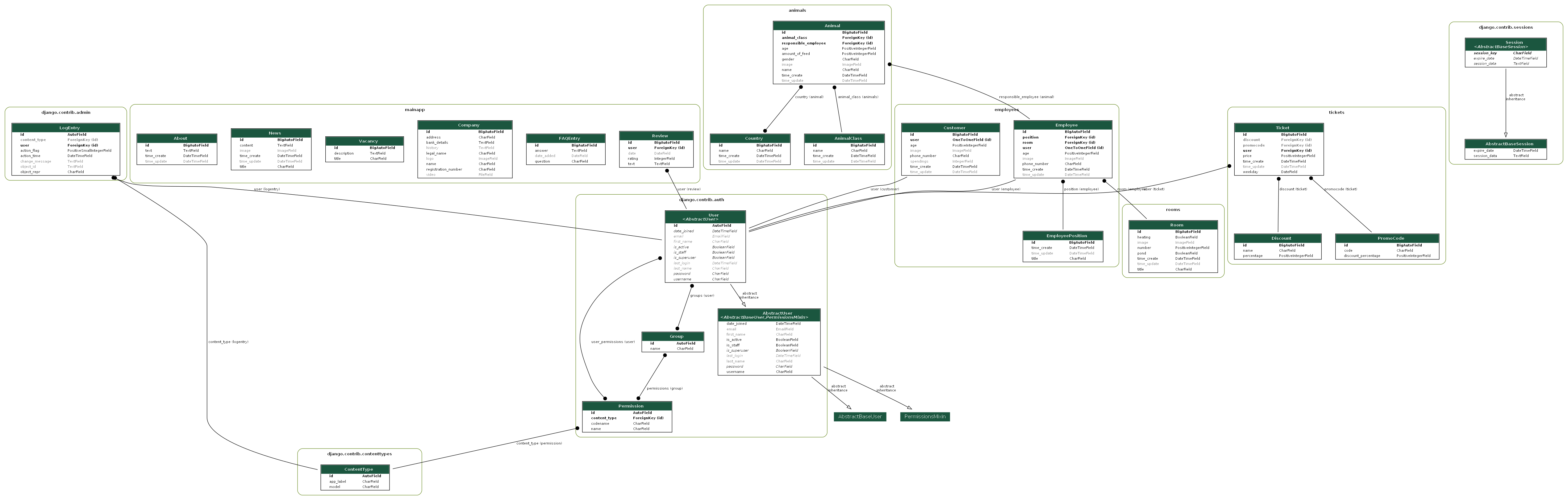
Реализовать веб-сайт с использованием Веб-фреймворк Django по индивидуальному заданию (выбрать вариант по номеру в журнале или реализовать свой проект - соглаёсовать с преподавателем до начала разработки)

**Постановка задачи**:

* Определить необходимые сущности для описания предметной области. Реализовать их в терминах моделей, используя подходящие типы данных и связи объектов <https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Models>. Представить на диаграмме в виде, как на примере ниже. В качестве базы данных рекомендована sqlite, но можно другую по согласованию с преподавателем.

Нужно предусмотреть возможность хранения и изменения данных для обеспечения функционала в соответствии с индивидуальным заданием и для страниц, которые должны быть в каждом приложении независимо от варианта. Эти страницы полностью будут реализованы в следующем семестре, поэтому сейчас можно ограничиться их наполнением через базу данных или админ панель, или выполнить по желанию(отмечено \*\*\*):

* *Главная*: Краткая информация о последней опубликованной статье
* *О компании*: информация о компании – просто текст (\*\*\* видео, логотип, история по годам, реквизиты – пока только сделать таблицу в базе данных) (таблица в базе данных)
* *Новости*: список статей в соответствии с тематикой сайта с заголовком, кратким содержанием (одно предложение), картинкой (таблица в базе данных) (\*\*\* и кнопкой «Читать далее» при нажатии на которую открывается вся статья)
* *Словарь терминов и понятий*: список часто задаваемых вопросов и ответов на них с датой добавления на сайт (таблица в базе данных)
* *Контакты*: Фото сотрудников с описанием выполняемых работ, телефонами, почтой и т.д. (таблица в базе данных)
* *Политика конфиденциальности*: пока добавить пустую страницу
* *Вакансии*: список вакансий с описанием (таблица в базе данных)
* *Отзывы*: список отзывов с указанием имени, оценки, текста, даты (таблица в базе данных). (Кнопка добавить отзыв с переходом к окну регистрации или в личный кабинет залогиненного пользователя. При нажатии кнопки открытие формы с полем для текста отзыва и выбора оценки, кнопкой «Отправить», которая сохраняет отзыв в базе. )
* *Промокоды и купоны*: список промокодов действующих и в архиве. (таблица в базе данных).
* Реализовать связи один к одному (OneToOneField), один ко многим (ForeignKey) и многие ко многим (ManyToManyField).



* Реализовать CRUD (create, read, update, delete) операции (<https://metanit.com/python/django/5.4.php> );
* Добавить все модели в админ панель, создать суперюзера, обеспечить работу с данными, фильтрацию, встроенное редактирование связанных записей? ‘что это значит’ (<https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Admin_site>).
* Реализовать механизмы авторизации/аутентификации;
* Разграничить доступ в зависимости от прав: например, владелец магазина(superuser), User с регистрацией, User без регистрации (см. инд. задание)
* Обеспечить наполнение данными для демонстрации не менее 10 записей в списке товаров/услуг/объектов/клиентов
* Подключить и использовать как минимум 2 сторонние API (базовый список прилагается), какие именно - выбрать на свой вкус.
* Использовать регулярные выражения для связи URL-адреса с функцией отображения (<https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Generic_views> )
* отображать статистические показатели для своего сайта (использовать данные, соответствующие предметной области). Например:
  + список клиентов/товаров в алфавитном порядке и общую сумму продаж;
  + статистические показатели (среднее, мода и медиана) по сумме продаж;
  + статистические показатели (среднее и медиана) по возрасту клиентов;
  + какой тип товаров наиболее популярен?
  + какой тип товаров приносит наибольшую прибыль?
* отображать тайм зону пользователя, текущую дату, дату добавления/изменения данных в таблицы для тайм зоны пользователя и для UTC, календарь в текстовом виде. Дату указывать в формате (DD/MM/YYYY)
* Номер телефона клиента указывается в формате +375 (29) XXX-XX-XX;
* Клиенты и сотрудники должны иметь возрастное ограничение 18+;
* Реализовать визуализацию в виде диаграммы или графика для отображения распределения показателей по группам и/или изменения показателей по датам/группам.
* Реализовать поиск по любому из параметров и сортировку отображаемых данных
* Добавить тесты. Покрытие тестами кода на 80% и выше (<https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Testing> ) (<https://pytest-docs-ru.readthedocs.io/ru/latest/parametrize.html> );
* Добавить logging(<https://habr.com/ru/companies/wunderfund/articles/683880/> );
* Внешний вид сайта не важен, главное отображать нужную информацию. Использование css по желанию. Если будете использовать <https://www.bootstrapcdn.com/>, то сохраните на github в разных ветках версию без стилей и со стилями. В следующем семестре разработаете свои стили и нужен будет проект без bootstrap;
* Валидация форм как на стороне сервера, так и на стороне клиента;
* Поддержка разного уровня логирования (уровень логирования брать из конфигурации приложения);
* Ограничить использование API проекта для неавторизованных запросов.
* \*\*\* дополнительное задание – по желанию тем, кто стремится освоить больше:
* Дополнить проект модулем для работы с параллельным кодом (можно использовать asyncio, multiprocessing или multithreading на выбор);
* API должно подыматься в production режиме;
* Разработать Dockerfile для вашего проекта (или частей проекта);
* docker-compose файл, с помощью которого возможно запустить проект локально (должен включать образы вашего проекта (образы сделать публичными, чтобы преподаватель мог запустить у себя) и базы данных + возможно какие-то сервисы, необходимые для проекта);
* Развернуть проект в облаке. Выбирать можно на вкус студента, но рекомендуется использовать что-то из heroku, AWS, GCP, Azure Cloud. (Аккуратно с платными ресурсами). Заранее посмотрите какую квоту на бесплатное пользование дает каждый облачный провайдер

**Примечание**: если проект будет не только в вашем гит репозитории, то результат лабораторных в этом семестре обнуляется.

Доработка html, css, js будет выполнена на первых трех ЛР в курсе «Современные технологии разработки web-приложений», поэтому обеспечьте сохранность проекта до следующего семестра.

Для справки cм. также инфо в СЭО:

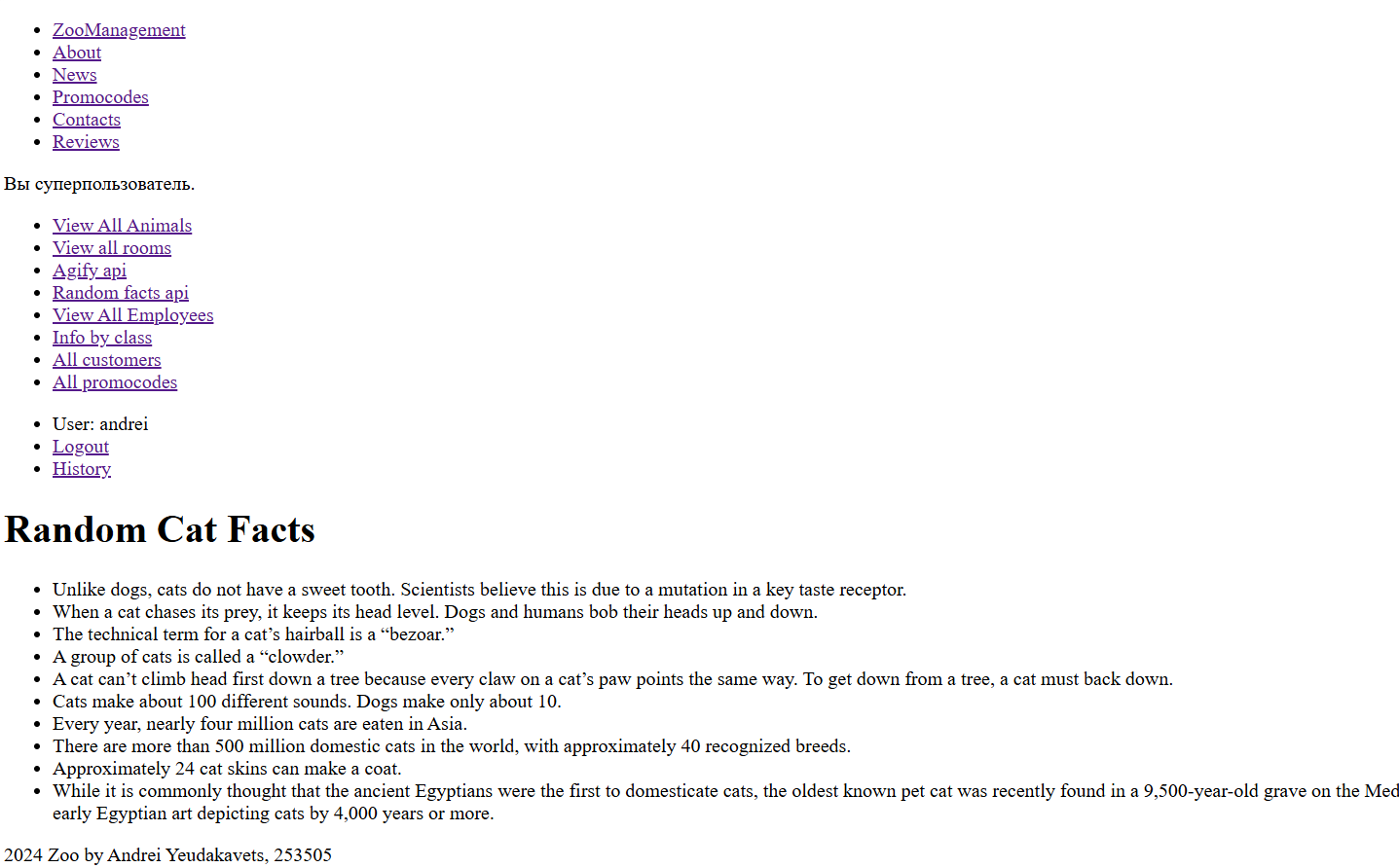
Список бесплатных API: <https://lms.bsuir.by/mod/resource/view.php?id=214284>

Как создать проект в Django и использовать стороннее API <https://lms.bsuir.by/mod/resource/view.php?id=214285>

**Варианты заданий:**

| **№** | **Web сайт** | **Ориентировочные модели** | **Описание**  **(что должно быть обеспечено)** | **Отображается для Админа (Superuser)** | **Отображается для User с регистрацией – личный кабинет + все что видит User без регистрации** | **Отображается для User без регистрации** |
| --- | --- | --- | --- | --- | --- | --- |
| 8 | Зоопарк | Животное, Помещения, Сотрудники, Должности сотрудников, Вид животного, Класс(Семейство животного), Страны обитания, Вид корма | хранение информации о деятельности зоопарка, т.е. о помещениях, животных, сотрудниках;  доступ к информации, какое животное в каком месте находится и какой сотрудник за ним закреплен;  учет расхода кормов разных видов. | Примеры вывода инфо:  суточное потребление корма обитателями комплекса ‘приматы’. • Определить общую численность представителей семейства ‘псовые’ в зоопарке. • Вывести все пары видов, которые содержатся в одном помещении.  информация о помещениях (номер, название, наличие водоема, площадь),  всех животных(название, вид, класс, дата поступления, год рождения, фото, факты, параметры, смотрящий сотрудник), и возможность отобрать те, которые поступили в зоопарк совсем недавно (на протяжении последнего полугодия); • какие сотрудники работают в каком комплексе (задается пользователем при поиске), приглядывая за животными,  всю информацию об этих сотрудниках(фио, зал, телефон, должность); | User с регистрацией – сотрудник – видит инфо о том, за какими животными закреплен, их детальное описание(включая дату поступления в зоопарк и год рождения), расписание кормления  Минимальный список характеристик: • Название вида животного, суточное потребление корма, семейство, континент обитания. • Название комплекса, номер помещения, наличие водоема, отопления, количество животных в помещении. Один и тот же вид животного может в зоопарке находиться в разных помещениях и в одном помещении может находиться несколько видов животных.  User с регистрацией – посетитель – имеет возможность купить билет, видит инфо о покупках и промокодах | Инфо о животных зоопарка с фото и описанием, помещениях и условиях обитания согласно виду животного,  Стоимость посещения в зависимости от дня недели, доп. услуг(кормить животных, активные часы, т.д.), промокоды, купоны |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |





Контрольные вопросы:

1. Что такое Django
2. Установка и настройка Django
3. Создание первого проекта
4. Создание первого приложения
5. Представления и маршрутизация в Django
6. Обработка запроса
7. Определение маршрутов и функции path и re\_path
8. Получение данных запроса. HttpRequest
9. HttpResponse и отправка ответа
10. Параметры представлений
11. Вложенные маршруты и функция include
12. Параметры строки запроса
13. Переадресация и отправка статусных кодов
14. Отправка json
15. Шаблоны
16. Создание и использование шаблонов
17. Передача данных в шаблоны
18. Встроенные теги шаблонов
19. Фильтры шаблонов
20. Статические файлы
21. TemplateView
22. Конфигурация шаблонов
23. Расширение шаблонов и фильтр extends
24. Вложенные шаблоны и фильтр include
25. Работа с формами в Django
26. Отправка форм
27. Определение форм Django
28. Типы полей формы
29. Настройка формы и ее полей
30. Валидация данных
31. Детальная настройка полей формы
32. Стилизация полей форм
33. Модели в Django
34. Подключение к базе данных
35. Создание моделей
36. Типы полей моделей
37. QuerySet API
38. Создание и получение объектов модели
39. Редактирование и удаление объектов модели
40. Фильтрация
41. values и values\_list и сортировка
42. Операции с множествами
43. Получение отдельных объектов и проверка их наличия
44. Агрегатные операции
45. Выполнение SQL-выражений
46. CRUD. Все базовые операции с моделями в веб-приложении
47. Отношение один ко многим (One to Many)
48. Практический пример связи один ко многим
49. Отношение многие ко многим (Many to Many)
50. Отношение многие ко многим (Many to Many)
51. Отношение один к одному (One to one)

