Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

**ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ СЕРВЕРНОЙ ЧАСТИ**

**ОБЛАЧНОГО СЕРВИСА ОБУЧЕНИЯ**

Отчёт по производственной практике: преддипломной

Студент гр. 434-1

Ю.А. Богомолов

\_\_\_\_\_\_\_\_\_\_\_ *«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2018 г.*

*Подпись*

Руководитель:

Преподаватель каф. АСУ

Доктор технических наук

\_\_\_\_\_\_\_\_\_\_\_ Профессор

*Оценка* М.Ю. Катаев

\_\_\_\_\_\_\_\_\_\_\_ *«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2018 г.*

*Подпись*

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ: ПРЕДДИПЛОМНУЮ ПРАКТИКУ

студенту \_\_\_\_\_\_\_\_\_\_\_Богомолову Юрию Алексеевичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

группа \_\_\_\_\_\_434-1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ факультет \_\_\_\_Систем Управления\_\_\_\_\_\_

срок практики с \_\_\_\_\_30.04.2018\_\_\_\_\_ по \_\_\_\_\_27.05.2018\_\_\_\_\_

1. Тема индивидуального задания \_\_\_\_\_\_Программное обеспечение серверной части облачного сервиса обучения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Перечень вопросов, подлежащих разработке \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от университета \_\_\_\_\_\_\_\_\_\_ М.Ю. Катаев

Преподаватель каф. АСУ,

д.т.н., профессов

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_ Ю.А. Богомолов

*«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.*

Содержание

[Введение 3](#_Toc494236523)

[1 Краткое описание системы 4](#_Toc494236524)

[2 Процесс разработки 5](#_Toc494236525)

[2.1 Подготовка к разработке 5](#_Toc494236526)

[2.2 Реализация базы данных 6](#_Toc494236527)

[2.3 Создание представлений страниц 8](#_Toc494236528)

[3 Результаты работы 11](#_Toc494236529)

[Заключение 13](#_Toc494236530)

# Введение

Производственная практика проходила на кафедре. В качестве темы задания была выдана тема «Программное обеспечение серверной части облачного сервиса обучения». Данная работа является продолжением того, что было проделано в рамках ГПО в проекте АСУ-1101 «Облачная информационная система обучения студентов». В связи с этим на начало работы уже имелась модель веб-сайта. Поэтому на время практики была поставлена цель разработать концепт серверной части системы. Исходя из этого была поставлена следующая задача: проработать структуру серверного програмнного обеспечения.

# 1 Описание концепта

## 1.1 Назначение системы

В настоящее время современные информационные технологии в образовании применяются не в полной мере. Даже в пределах одной страны использование информационных технологий может сильно разниться: где-то активно используют интерактивные доски, а где-то – обычные доски для мела; в некоторых вузах применяются комплексные системы управления обучением, а в других учебных заведениях занимаются только по бумажным книгам.

Поэтому возникла идея разработки системы, целью которой была бы модернизация процесса обучения, будучи при этом простой и удобной в использовании. Удобной как для студентов и преподавателей, так и для целых организаций.

Система, способная достичь указанной цели, должна предоставлять следующие базовые возможности:

1. Для преподавателя:
   1. создавать и размещать учебные материалы;
   2. выдавать учащимся индивидуальные и коллективные задания;
   3. просматривать решения учащихся.
2. Для студента:
   1. просматривать учебные материалы;
   2. решать выданные задания;
   3. отслеживать свои достижения для составления собственного портфолио.
3. Для организации:
   1. отслеживать активность студентов;
   2. предоставлять свои учебные материалы другим организациям.

Однако отличительной особенностью предлагаемой системы, качественно улучшающей вышеописанные функции, является невероятная гибкость, позволяющая автоматизировать многие процессы. Подобная гибкость достигается за счёт возможности разработать собственные приложения, способные функционировать как на серверной, так и на клиентской стороне. Эти приложения могут быть использованы в разработке учебных материалов и других приложений, выдаче и решении заданий, а также проверке решений.

## 1.2 Структура системы

Перечисленное в прошлом подразделе многообразие функций требует довольно сложной структуры системы, множество компонентов: веб-сайт, серверное программное обеспечение, приложения, учебные материалы.

# 2 Процесс разработки

## 2.1 Подготовка к разработке

Для начала работы необходимо было определиться с языками и инструментами, с помощью которых бы велась разработка. Для начала были выбраны фреймворк Django и, соответственно, язык программирования Python 3 (далее просто Python). В Django по умолчанию предусмотрены три варианта СУБД: SQLite, MySQL и PostgreSQL. Из этих трёх вариантов самым оптимальным показался последний, из-за чего и был выбран.

Следующим этапом стало изучение Django и Python. Это заняло некоторое время, так как мои познания Python были достаточно посредственные, а к Django я прежде вообще не прикасался. Кроме того, пару дней пришлось также изучать CSS, чтобы потом верстать front end сайта. Впрочем, обучение совмещалось с разработкой системы, что не сильно снизило темп работы.

## 2.2 Реализация базы данных

Django позволяет упростить процесс реализации структуры базы данных следующим образом: предлагается описывать каждую сущность в качестве класса Python. Затем он преобразует созданные классы в команды выбранной СУБД. Пример такой сущности представлен в листинге 1.

Листинг 1. Код класса сущности ПОЛЬЗОВАТЕЛЬ

|  |
| --- |
| class User(models.Model):  """Сущность ПОЛЬЗОВАТЕЛЬ базы данных."""  name = models.CharField(max\_length=200)  job = models.CharField(max\_length=200)  description = models.TextField()  path = models.CharField(max\_length=256) |

У данного способа есть недостаток: нельзя создавать композитные первичные ключи. Если требуется сделать такое, то вместо первичного ключа необходимо задавать соответствующие атрибуты как «Уникальные в связке» (“Unique Together”). Пример такого решения представлен листингом 2.

Листинг 2. Решение проблемы отсутствия композитного ключа.

|  |
| --- |
| class Teacher(models.Model):  """  Сущность ПРЕПОДАВАТЕЛЬ базы данных.  ПОЛЬЗОВАТЕЛЬ, который может управлять проектами и выдавать задания.  """  user = models.ForeignKey(User, on\_delete=models.CASCADE)  project = models.ForeignKey(Project, on\_delete=models.CASCADE)  class Meta:  unique\_together = ('user', 'project') |

Кроме того, у сущности, где не указан явно первичный ключ, всегда будет уникальное автоинкрементируемое поле *id*, которое будет являться таковым. Впрочем, это скорее даже является плюсом в данном проекте, т.к. в подготовленной схеме базы данных большинство сущностей в качестве первичного ключа имели именно такое поле. Это можно увидеть на рисунке 1, демонстрирующем схему FA базы данных.

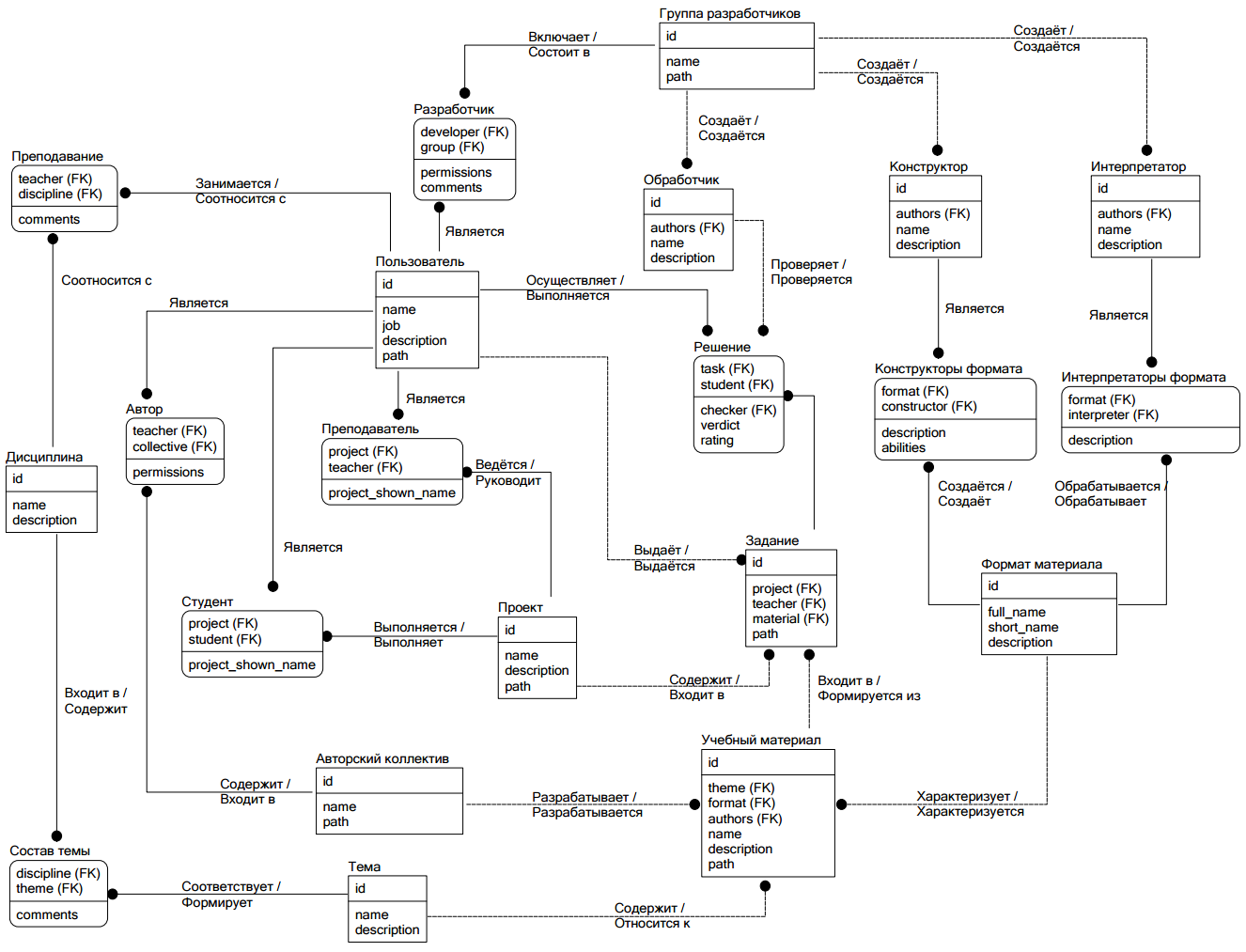


Рисунок 1. FA-модель базы данных

## 2.3 Создание представлений страниц

Django предлагает следующий способ обработки запросов и генерации страниц. Создаётся класс, который содержит методы обработки различных типов запросов (GET, POST и другие). Кроме того, он может извлекать из базы данных необходимые для этого данные. Так как сущностей в БД много, и для каждой было необходимо сделать страницу, а код получался однотипным, было решено написать генератор таких представлений, генератор для HTML-страниц, а также вспомогательные классы, оптимизирующие выборку данных. Код представления (View) страницы пользователя представлен листингом 3, а часть шаблона соответствующей HTML-страницы – листингом 4. Код, представленный в этих листингах большей частью был именно сгенерирован.

Листинг 3. Код представления страницы пользователя

|  |
| --- |
| class UserView(MyView):  """Представление страницы пользователя."""  template = 'core/user/user.html'  get\_context = cgg.make\_context\_getter(context={  'details': cgg.Get(User, pk='id'),  'teacher': {  'projects': cgg.Count(Teacher, user='id'),  'tasks': cgg.Count(Task, user='id'),  },  'student': {  'projects': cgg.Count(Student, user='id'),  'solutions': cgg.Count(Solution, user='id'),  },  'author': {  'collectives': cgg.Count(Author, user='id'),  },  'developer': {  'groups': cgg.Count(Developer, user='id'),  }  }, const\_getters={  'id': lambda \*\*kwargs: kwargs.get('pk'),  }) |

Листинг 4. Часть HTML-кода страницы пользователя

|  |
| --- |
| <table class = "info\_panel">  <tr><td>Name:</td><td>{{ details.name }}</td></tr>  <tr><td>Job:</td><td>{{ details.job }}</td></tr>  <tr><td>Description:</td><td>{{ details.description }}</td></tr>  </table>  <div class = "long-preview">  <a href = "{% url 'user/teacher' details.pk %}">  <span>Teacher</span>  {% if teacher %}  <ul>  {% if teacher.projects %}  <li>projects: {{ teacher.projects }}</li>  {% endif %}  {% if teacher.tasks %}  <li>tasks given: {{ teacher.tasks }}</li>  {% endif %}  </ul>  {% else %}  <p>You still have not started being teacher. Wanna try? Just click here!</p>  {% endif %}  </a>  </div> |

Кроме того, для оформления страниц потребовалось написать CSS-код, часть которого представлена листингом 5.

Листинг 5. Часть CSS-кода, используемого для отображения страниц системы

|  |
| --- |
| .info\_panel {  padding-left: 1em;  margin-bottom: 1em;  }  .preview, .long-preview {  display: inline-block;  background-color: #CCC;  width: 200px;  height: 80px;  margin-bottom: 1em;  margin-right: 1em;  padding-left: 0.5em;  padding-top: 0.5em;  padding-right: 1em;  font-family: Arial;  } |

# 3 Результаты работы

В результате прохождения летней практики получился небольшой сайт, отображающий состояние базы данных. Результаты представлены рисунками 2-5. Конечно же, это лишь начальное видение системы, не отражающее её конечное представление. Кроме того, оно не выполняет даже базовых функций системы. Тем не менее, благодаря разработке такого макета, разработчик узнал, как работать с Django, что позволит быстро приступить к разработке полноценного прототипа уже в рамках ГПО.



Рисунок 2. Форма авторизации



Рисунок 3. Главная страница сайта

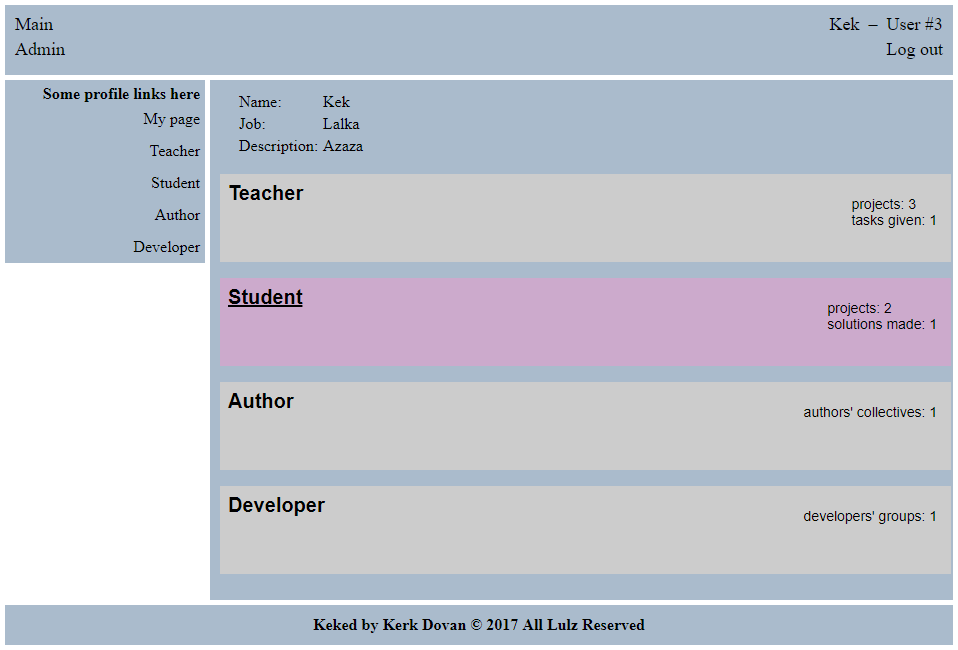


Рисунок 4. Страница пользователя



Рисунок 5. Раздел «Студент» страницы пользователя

# Заключение

В результате прохождения летней практики были выполнены все поставленные задачи: определены и изучены все необходимые для разработки инструменты, а также положено начало разработке системы. Следовательно, цель была достигнута.