ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

**Факультет информатики, математики и компьютерных наук**

**Программа подготовки бакалавров по направлению   
(09.03.04) Программная инженерия**

*Дыряев Даниил Александрович*

**C O U R S E W O R K**

Разработка соревновательной образовательной системы с автоматической оценкой решений

|  |  |
| --- | --- |
|  | Научный руководитель  Ст. Преподаватель  Лемайкина Елена Александровна |

Нижний Новгород, 2020

**ОГЛАВЛЕНИЕ**

Страница 2 ([тык](#Content)): Оглавление

Страница 3 ([тык](#Introduction)): Введение, актуальность рассматриваемой темы. Возможно даже введение в актуальность рассматриваемой темы.

Страница 4 ([тык](#TT)): Постановка задачи. Everything begins with choice

Y: Что требуется сделать?

Y: Протокол взаимодействия с пользователем

Y: Пример. (конкретно дерево событий)

X: Делаем обзор на то, как эту задачу решили до нас. Reliable like a Swiss watch

Y: Существующие продукты

Y: Проблемы в существующих решениях

X: Описание моего решения. Sometimes you gotta run before you can walk

Y: Процесс написания продукта

Y: Описание структуры проекта

Z: Приложения: кто за что отвечает

Z: Алгоритмы, выходящие за пределы стандартного web-приложения

Z: Боевой режим

X: Описание тестирования продукта. Houston, we have tests

Y: Сценарии использования

Y: Стандартное тестирование и безопасность продукта

X: Заключение. May the code be with you

Y: Подведение итогов

Y: Что дальше?

*Автор желает вам приятного чтения*

**ВВЕДЕНИЕ**

Путь в тысячу ли начинается с первого шага. А проект в тысячи строк начинается с первого переноса строки, потому что IDE ругается на отсутствие пустого места в конце файла.

Не расписывая название темы работы, перейдем к делу.

Ни для кого не является секретом, что спортивное программирование медленно, но верно завоевывает свои позиции, и не собирается отступать. Мы живем во времена преобладающей рыночной экономики, когда на любой возникающий спрос найдется предложение. Спортивное программирование не стало исключением, и породило невероятный спрос на платформы, имеющие возможность его удовлетворять.

Спортивное программирование — это отрасль, обладающая рядом факторов, которые позволят ей остаться на плаву многие годы, хотя разговор уже сейчас можно вести о десятилетиях.

Несмотря на спорное положение и неоднозначную оценку среди программистов, занимающихся профессиональной деятельностью, спортивное программирование играет важную роль в начале карьеры многих молодых людей.

Взять к примеру Всероссийскую олимпиаду школьников по программированию, которая способствует развитию культуры программирования среди учеников школ.

Самым ярким примером на сегодняшний день является студенческий чемпионат мира по программированию ACM ICPC, в котором участвуют десятки тысяч студентов со всех точек мира. В нем, молодые программисты могут не только показать умение решать сложные логические задачи в ограниченный период времени, но и заинтересовать потенциальных работодателей, что является предусмотрительным и выгодным ходом.

**ПОСТАНОВКА ЗАДАЧИ**

**Что требуется сделать?**

*“Everything begins with choice” (Morpheus)*

Я, как человек, вовлеченный в спортивное программирование, решил привнести свою чеканную монету в этот необъятный мир.

Передо мною встала задача написания платформы, которая служила бы полноценной средой для подготовки людей к чемпионатам и олимпиадам по спортивному программированию.

Давайте попробуем понять, что нужно для подготовки к испытаниям в полностью независимой от других сервисов онлайн-среде.

Во-первых, необходимо дать достаточно теоретической информации, чтобы у обучающихся была интеллектуальная база для дальнейшего применения полученных знаний на настоящих соревнованиях.

Во-вторых, принципы эффективного обучения диктуют нам необходимость отрабатывать свои знания на практике, чтобы обучающийся смог освоиться в полученных навыках и хорошо ориентировался в ситуации, когда эти навыки следует применить.

В-третьих, следует дать возможность обучающимся пройти “боевые испытания”, по условиям максимально приближенные к реальным соревнованиям. Будущие спортивные программисты смогут освоится в той среде, которую предлагает им платформа, и на реальных соревнованиях они проявят способность успешно ориентироваться в условиях непосредственного соперничества.

Таким образом, задача формируется непосредственно из трех вышеперечисленных пунктов:

1. Обеспечить удобное и доступное управление обучающими материалами для преподавателей.
2. Обеспечить обучающихся возможностью применять свои навыки в решении задач.
3. Организовывать онлайн-соревнования для проверки успеваемости обучающихся и их подготовке к реальным олимпиадам.

**Протокол взаимодействия с пользователем**

Подавляющая часть продуктов, выпускаемых на рынке программирования нацелены на взаимодействие с пользователями.

Мой продукт – не исключение, поэтому необходимо определить все сценарии взаимодействия пользователя с сайтом.

1. Во-первых, это регистрация и авторизация пользователей, используемая для дальнейшей удобной работы с ними.   
   Здесь стоит обратить внимание на то, что кроме таких обязательных данных, как псевдоним и почта, мы будем собирать имя и фамилию (хоть их можно впоследствии и не указывать), что сразу делает нас оператором персональных данных.   
   Это, в свою очередь, накладывает на нас ответственность, которая звучит как Федеральный закон 27.07.2006. №152-ФЗ.  
     
   В силу этой причины, мы должны определить такую вещь, как “Политика конфиденциальности”. Она подразумевает, что все персональные данные, которые мы получаем от пользователя, мы используем исключительно в целях, улучшающих качество взаимодействия пользователя и сайта.  
     
   Также это подразумевает, что мы не получим штраф из-за какой-то

глупости.

1. Во-вторых, это заполнение различных форм на сайте, таких как подтверждение регистрации на соревнования и обучающие курсы, и многое другое.   
     
   Особое внимание стоит уделить форме с отправкой решений задач на соревнованиях и в архиве. С этими данными проблем не возникает.   
     
   Код не является персональными данными. Также написанное не является интеллектуальной собственностью, потому что решение задачи известно заранее, и патентовать свои алгоритмы никто в данном случае не будет. Поэтому мы можем использовать данные из форм как угодно, и это будет законно.

**Примеры использования в реальности**

Давайте подумаем, в каких условиях система, будучи достаточно развитой, может быть незаменима.

К примеру, онлайн-обучение является актуальным в условиях мировой самоизоляции, поэтому рассмотрим сценарий обучения по предмету “Программирование" непосредственно на платформе.

Традиционно обучение по предмету в университете состоит из теории и практики, а именно из лекций и семинаров. Давайте посмотрим, какую альтернативу предоставляет моя e-learn платформа.

Во-первых, это создание и подготовка преподавателем обучающего курса по предмету и заполнение модулей контентом, относящимся к предмету.

После этого студенты могут изучать рекомендуемые модули без непосредственного взаимодействия с преподавателем.

Во-вторых, для организации практики, или, так называемых семинаров, подойдут приложения “Архив” и “Соревнования”.

Преподаватель, совместно с администратором сайта готовят необходимый материал.

Подготовка материала, по моему опыту, ~~та еще запара~~ процесс нелегкий, поэтому затратный по времени. Исходя из личного опыта, подготовка несложного соревнования на 5 средних по уровню задач заняла у меня около пятнадцати часов чистого времени. Это даже без учета времени, которое мне потребовалось, чтобы эти задачи придумать.

Но игра стоит свеч. Усилия, затраченные на подготовку занятия в такой форме, многократно окупится качеством его проведения.

Почему семинары в виде соревнования — это круто? Начнем с того, как проходили семинары по программированию в Университете перед карантином:

Преподаватель говорит, где лежит файл с текстовым описанием задач.  
После этого студентам дается около 80 минут на их выполнение. В конце занятия преподавателю требуется подойти к каждому студенту, чтобы выборочно проверить несколько задач. После этого студент получает оценку и становится свободен.

Это бывает неэффективно. К примеру, какие-то задачи могли оказаться неправильно решены, на что преподаватель может не обратить внимания. Также возможно, что сразу несколько человек дорешают задачи досрочно, либо несколько человек решат задачи только к концу занятия. В этом случае преподаватель очень сильно нагружен, что не очень хорошо.

Проведение контеста в качестве семинара решает эти проблемы.

Перед занятием семинаристу лишь требуется дать студентам ссылку на соревнование, а самому лишь заварить чай, и наслаждаться тем, как студенты тренируют свои навыки программирования.

Не возникает проблем с проверкой задач при их выполнении – система даст объективный вердикт в любом случае. После соревнования система предоставляет таблицу результатов, в соответствии с которой можно выставлять оценки/баллы за семинар.

**ДЕЛАЕМ ОБЗОР НА ТО, КАК ЭТУ ЗАДАЧУ РЕШИЛИ ДО НАС**

“That is a great plan, Walter. If I understand it correctly, it’s a Swiss watch” (*Jeffrey «The Dude» Lebowski*)

**Существующие решения**

В силу популярности спортивного программирования, есть масса сайтов, предлагающие похожий функционал.

Давайте посмотрим на 3 из них, по возрастанию. Возрастание определяется по ~~субъективному никому не нужному мнению автора~~ степени крутости:

1. <https://acm.timus.ru/>
2. <https://welcome.stepik.org/>
3. <http://codeforces.com/>

**Timus Online Judge**

Самый старый представитель из списка мною выбранных.

По словам некоторых программистов, существовал еще при коммунизме и проработает еще пару тысяч лет.

Сайт выполнен в максимально лаконичном стиле, что заметно сразу. Функционал у платформы небольшой, тем не менее все еще удовлетворяет предъявляемым требованиям.

Сайт предоставляет две возможности – это решение задач в архиве и участие в соревнованиях.

Я, как человек, решивший на этом сайте более ста задач, могу выделить пару особенностей.  
Во-первых, для запуска кода используется не самое мощное железо, благодаря чему ломаются ожидания сдать задачу на python 3.6.0 с ассимптотикой , хоть и с немалой константой. Что интересно, решение на языке C++ (Компилятор gnu g++, стандарт 14) отработало за 31 миллисекунду, что было очень странным случаем.   
([Задача 2071. Фруктовые коктейли](https://acm.timus.ru/problem.aspx?space=1&num=2071))

Во-вторых, есть онлайн-соревнования в духе ACM ICPC, но:

- На них приходят мало людей.

- Контесты проходят очень редко.

Из-за этого становится понятно, что,   
как сказал один мой знакомый – Неактуально.

Сайт функционирует, и удовлетворяет запросы пользователей на наличие автоматической проверки задач, но с точки зрения новых пользователей платформа может показаться непривлекательной, потому что в проекте нет того, что могло бы заинтересовать новых клиентов и заставить их остаться на сайте на долгое время.

**Stepik**

Самая молодая платформа из списка, успевшая стать за время существования довольно популярной.

Платформа была выбрана в качестве примера, потому что она довольно разносторонняя и гибкая, поэтому удобна как для преподавателей, так и для студентов.

Каждый имеет возможность создавать курсы, и это, как я считаю, правильный подход, потому что данное решение позволяет привлечь на платформу новых ценных преподавателей.

Создание и настройка курсов гибкая и имеет достаточный функционал, чтобы быть конкурентоспособной на e-learning рынке.

Также стоит отметить, что имеется возможность автоматической проверки задач. Реализована она отвратно (Как и у меня сейчас, ха-ха).

Все работает, и к этому вопросов нет. Но как работает?

1. Файл загрузить нельзя.
2. Нужно писать в мерзкой online IDE.
3. Отправил задачу на проверку? – Будь добр, подожди. Прогресс нельзя отследить непосредственно.
4. Когда задача упала на тесте – тебе предоставляется код ошибки. Это, скажем, не очень хорошая практика, хотя для учебных целей подойдёт.

В связи с перечисленными недостатками можно сделать для себя некоторые выводы и не повторять их у себя в проекте.

*Здесь стоит отметить, что похожие в чём-то косяки есть и у меня сейчас, но уже при дальнейшей разработке я буду знать об их существовании и, соответственно, принимать меры по их устранению, и это хорошо.*

Автор ©

**CodeForces**

Пожалуй, пока что это одна из самых качественных и проработанных платформ для проведения онлайн-соревнований по программированию. И это говорю не я. Сама за себя скажет популярность сайта на сегодняшний момент.

На каждом контесте собирается около 10-15 тысяч человек, в то время как на acm.timus.ru это количество редко переваливает за сотню участников.

С CodeForces я брал пример в сфере проведения онлайн-соревнований. Когда у меня зародилась идея сделать свою платформу, объединяющую концепции e-learn и acm, на сайте еще не было обучающих курсов. Именно это обстоятельство подтолкнуло меня к мысли о том, что моя разработка станет в какой-то степени уникальной.

Но неожиданно, примерно весной, на codeforces появляется первый онлайн-курс. Реакция у меня тогда была примерно следующая:



От своей идеи, я, тем не менее, не отказался, и продолжил делать.

Почему же codeforces является для меня примером для подражания и неким эталоном в мире спортивного программирования?

Во-первых, реализация системы проведения соревнований сделана на высшем уровне. Проблемы возникают разве что из-за внешних угроз.

Во-вторых, при долгом пользовании сайтом, возникает впечатление, что платформа возникла не на пустом месте, и пустышкой не является.

Все эти моменты не оставят равнодушным пользователя, который опробует все идеи и функции сайта собственноручно.

**Проблемы в существующих решениях**

По большому счёту, я перечислил всё, что мне не нравится в платформах, которые ненароком попались под мою руку.

Давайте резюмируем те моменты, которые ухудшили мою оценку этих сайтов, и на которые я обращу внимание при выполнении своей работы:

Timus:

1. Сайт имеет недостаточно функционала, чтобы привлекать новых людей. Из-за этого платформа непопулярна.
2. Платформа медленно развивается или не развивается вообще, поэтому интерес пользователей бистро охладевает, и нет того, что могло бы его подогреть.

Stepik:

1. Система проверки кода пользователей недостаточно проработана и не имеет гибкости.

CodeForces:

1. Если посмотреть на сайт со стороны концепции e-learning, он не привлекателен для новых преподавателей, потому что доступ к созданию курсов пока что сильно ограничен.

\*\*\*

Что ж, если вы дочитали до этих строк, значит вы заинтересованы в происходящем, и хотите посмотреть на результаты моей работы.

Как сказал один известный персонаж – “Не можем ходить, будем бегать”.

Поэтому мы поехали.

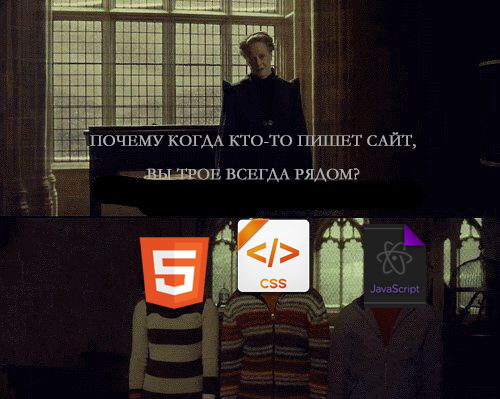
**Процесс написания продукта**

Я человек серьезный, поэтому выбрал язык python в качестве основы для своего проекта.

Есть такая шутка про этот язык – “Нажал на пробел – создал стартап”.

Смешно, и отчасти является правдой. Python славится быстрой разработкой и развертыванием проектов, написанных на нём. Это стало решающим фактором в дальнейшем выборе инструментов для разработки.

Сам по себе проект – это web-application со сложной логикой и множеством модулей, из которых и собран сайт. При разработке пользовательского интерфейса был использовал стандартный набор, в который входят HTML5, CSS3, JavaScript. Хоть последнего в проекте пара десятков строк, но свою роль он отыграл неплохо.



На чистом питоне далеко не… Нет, на чистом питоне можно далеко уехать, и даже улететь, но писать сайт без использования специально созданных для этого инструментов – просто нерациональный подход к работе.

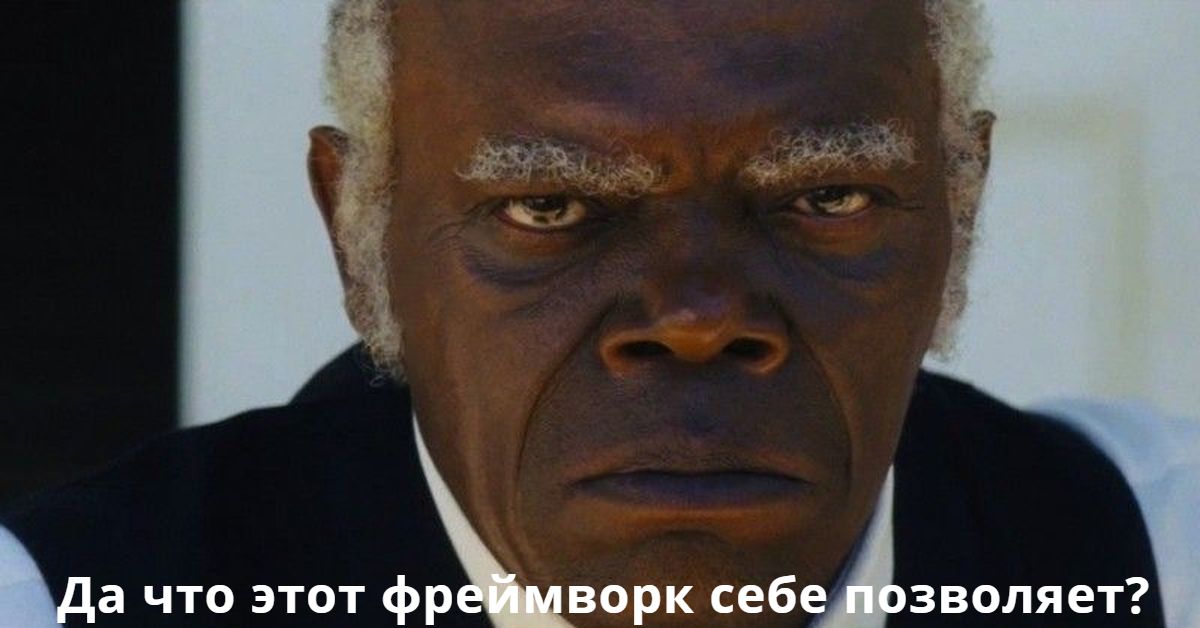
Изначально идея была использовать Flask, но как точно подметил Халк после избиения Локи в башне Старка – “Мелковат”. *(с интонацией)*

Выбор пал на Django, который как раз заточен под написание web-проектов средней степени сложности.

Стоит сказать про плюсы, из-за которых Джанго стал единственным кандидатом на роль серверной части проекта.

Во-первых, это хорошие отношения с базами данных, о которых я скажу немного позже. Во-вторых, встроенная админ-панель, что упрощает разработку в раз тридцать. В-третьих, собственный удобный язык шаблонов, по синтаксису похожий на python.

Это только одни из тех многих преимуществ, которые дает это технология разработчику. Вы спросите –



* Он позволяет себе и программисту многое, и именно поэтому я пока что не вижу конкурентов на языке python в той области, в которой я провожу свою разработку.

Осталось определиться с ~~названием~~ базой данных. И тут, конечно же, Django снова всех переиграл и уничтожил, предоставив из коробки использование своей собственной БД – Sqlite. Эта утилита подойдет для написания несложных приложений по типу личного блога или новостного сайта, но нам нужны более продвинутые возможности.

Мы люди серьезные, поэтому выберем PostgreSQL, который довольно-таки в familiar отношениях с python, в особенности с нашим framework’ом. Для её использования также понадобится пакет (библиотека) Psycopg2, который предоставляет возможность работы с БД прямо из кода проекта.

Python 3.8 + Django 3, HTML5, CSS3, PostgreSQL – минимальный набор, который понадобился мне для успешной разработки.

В процессе написания постоянно возникала необходимость использовать сторонние библиотеки, пакеты и утилиты, облегчающие жизнь.

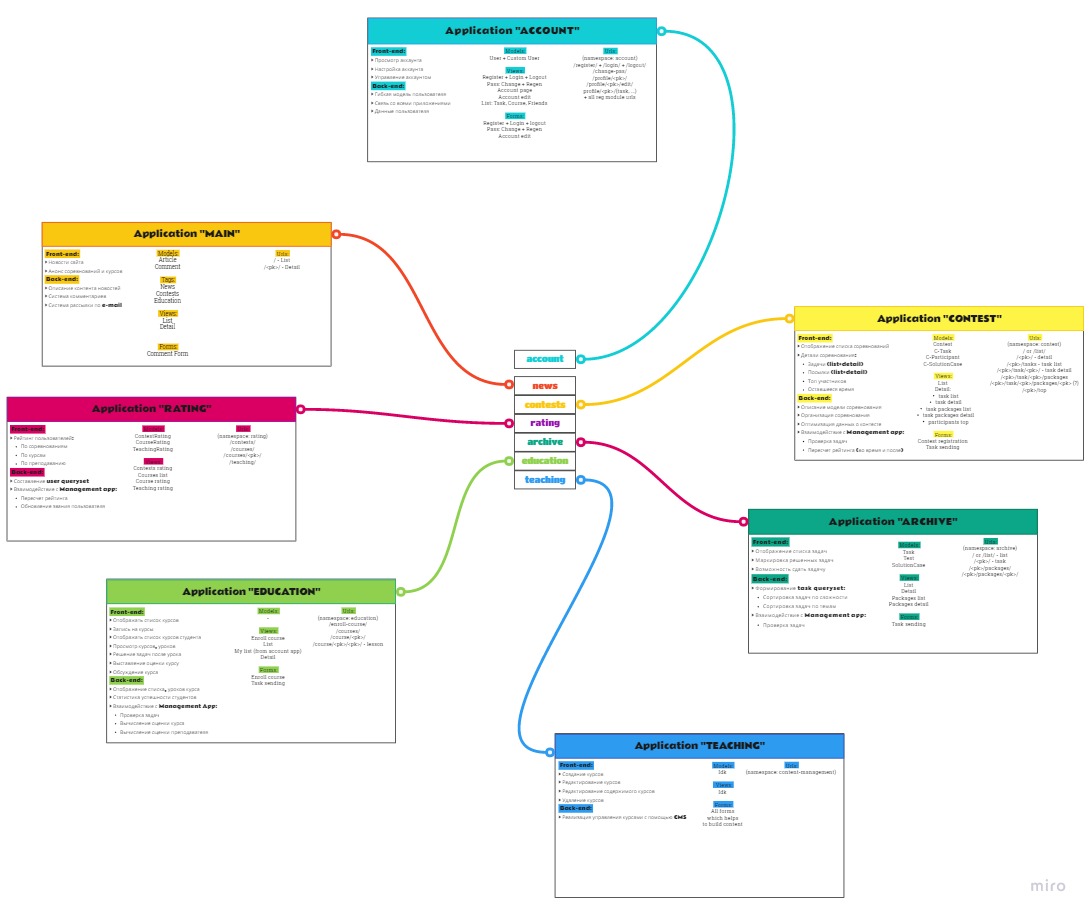
Их список представлен ниже:

* GNU Compiler Collection (GCC)
* Pillow
* Django-braces
* Django-embed-video
* Django-hashedfilenamestorage
* Django-taggit
* Djangorestframework
* Python pip (А можно без него?)
* Psycopg2
* Setuptools
* SQLparse
* Wheel

О некоторых из них я скажу, где это потребуется. О некоторых ничего не скажу, потому что их использование либо подразумевается изначально, либо в этом нет надобности.

Так-с, про инструменты все сказал, а что с разработкой?

Процесс был нелегкий, хотя-бы из-за объема работы, который был выполнен. Для поддержания последовательности разработки, она велась по изначально составленной схеме (Сверху вниз):



Для каждого отдельного приложения, из которого состоял проект, были указаны ключевые особенности и функционал, которых я придерживался на всем пути разработки. От начала и до конца.

Разработка была построена по принципу Down-to-top. Что это означает?

Это значит, что изначально разрабатывалась серверная часть и функционал. Т.е. скелет проекта, и только потом на всё это наслаивался пользовательский интерфейс.

Стоит сказать, что в процессе разработки я понимал, что то, что я написал неделю назад, я бы написал по-другому, потому что за эту неделю я смог придумать этому более оптимальную реализацию и узнал что-то новое, что использовал бы за место того, что есть.

Это казалось проблемой, до тех пор, пока сроки не начали поджимать, и было принято решение оставить как есть. Тем более, то что было – уже было хорошим решением в рамках той работы, которой я занимался.

Хотел бы еще отметить, что на исправление ошибок и фикс багов уходило немного больше времени, чем на написание кода. Думаю, это обычная практика в разработке чего-то хоть сколько-то серьезного, если ты работаешь один. О разработке сказали, давайте смотреть на результат.

**ОБЗОР ПОЛУЧИВШЕГОСЯ РЕЗУЛЬТАТА**

**Приложения: Кто за что отвечает**

Проект писался так, чтобы быть готовым к рефакторингу и расширению. С данной задачей справился концепт модульности. Что это значит? Проект состоит из нескольких приложений, каждое из которых отвечает за свой аспект.

Почти каждое приложение изолированно, но имеет немало связей с другими. Такой подход обеспечивает богатый функционал и удобство с точки зрения разработчика, так как все аккуратно распределено, а не валяется в одной куче.

Всего в проекте есть 8 приложений. Думаю, что стоит сказать пару слов о каждом из них.

Начнем с Account application. Приложение отвечает за пользователей, и всё, что с ними связано.

Сам по себе пользователь сайта представлен моделью UserProfile, унаследованной от стандартной модели User из модуля *django.contrib.auth.models.*

Давайте составим список того, что может пользователь с помощью нашего приложения:

* Просматривать профиль любого пользователя. (в том числе и свой, так, если что)
* Просматривать список своих посылок к задачам по программированию.
* Просматривать список своих курсов.
* Сменить пароль от аккаунта.
* Редактировать информацию об аккаунте.

Не забываем о политике конфиденциальности, потому что есть возможность указать свое имя и фамилию.

Профиль пользователя во многом служит “мостом” взаимодействия реального пользователя и сайта. Все другие приложения используют account application, что автоматически делает его основным.

Давайте посмотрим на реализацию. В данном листинге я продемонстрирую обработчик, который отвечает за регистрацию пользователей.



Как выглядит страница регистрации, можно посмотреть непосредственно на сайте.

А мы двигаемся дальше, и переходим к приложению Main application.

Почему main? Потому что оно отвечает за новостную ленту – куда пользователь попадает по ссылке на сайт. Возможность написания и публикации новостей есть у администратора сайта. В дальнейшем такая возможность будет не только у него.

Пользователь же, в свою очередь, имеет возможности читать ~~то, что заблагорассудится написать админу~~ важные для сайта новости и оставлять к ним комментарии от своего лица (от лица своего профиля).

Здесь я приведу фрагмент кода, отвечающий за описание статьи (новости) для базы данных. Каждый атрибут класса – колонка в бд.



И снова прыгаем дальше. На очереди самое интересное – Contest application.



Вообще онлайн-соревнования, это то, что немного не вписывается в стандартное поведение моделей Django, поэтому пришлось напридумывать всяких непотребств, с помощью коих всё работает так, как требуется.

Реализация соревнований пока что похожа на ту, что используется на платформе codeforces. Она мне понравилась, поэтому я решил пока что сделать похожую систему. В дальнейшем, конечно, доработка неизбежна.

Давайте поговорим о том, как всё происходит.

Изначально, соревнование нужно подготовить к проведению. Это делается администратором сайта. Необходимо придумать условие задач, а также тесты и решения к ним. Решения, очевидно, пишет сам администратор, поэтому ему необходимо иметь достаточную компетенцию в олимпиадном программировании и алгоритмах.

Администратор – я.

*(Здесь должна играть композиция “Curb your enthusiasm”)*

Когда всё готово, хорошим дополнением будет анонсировать соревнование в разделе новостей, чтобы ~~нагнать трафик~~ привлечь на соревнование больше участников.

Для участия в соревновании необходимо в нём зарегистрироваться. При регистрации есть формальное соглашения с правилами участия, которые регламентируют общие соображения и тонкости проведения его на платформе.

Соревнование ограничено по времени – это контролируется таймером на стороне клиента и обработчиком запросов на сервере.

Внутри этого временного промежутка участнику доступен список задач, к которым он может отправлять посылки с кодом. На стороне сервера эти посылки обрабатываются, и пользователю возвращается результат, но об этом немного позже.

В это же время работает таблица с рейтингом, который обновляется онлайн. Участник в любой момент может посмотреть свою позицию.

По завершению соревнования идет пересчет рейтинга пользователей и изменение их звания, если это потребуется.

Также все задачи с тестами и посылки пользователей переносятся в модели приложение Archive application.

Для примера реализации я выбрал обработчик, который отвечает за отображение условия задачи и отправку посылок. Думаю, нет смысла здесь что-то либо комментировать.

Могу сказать лишь, что обработчик отвечает также и за формирование таблицы результатов и подсчет штрафа, выполняя одновременно несколько важных функций.



Немного позже я расскажу, как проверяются посылки пользователей. А сейчас мы переходим к следующему приложению!

На очереди архив задач.

Скажу сразу – здесь нет ничего особенного, но приложение важно, и вот почему.

У участников появляется возможность дорешивать нерешенные на соревновании задачи, чем они могу повысить уровень своих умений и навыков.

Казалось, что почти все готово для приложения, потому что все может понадобится есть в соревнованиях? Вот тут и выходит на сцену модульность и наследование, которые решают проблему дублирования кода.

Посмотрите на все модели, которые используются в этом приложении:



Маловато, не правда ли?