**Содержание**

[ВВЕДЕНИЕ 4](#_Toc8977409)

[ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 7](#_Toc8977410)

[1.1 Обзор методики проведения и жюрейства CTF-соревнований 7](#_Toc8977411)

[1.2 Обзор функционала жюрейных систем для соревнований типа tack-based. 9](#_Toc8977412)

[1.3 Обзор существующих жюрейных систем 10](#_Toc8977413)

[1.3.1 Обзор жюрейной платформы «CTFd» 11](#_Toc8977414)

[1.3.2 Обзор жюрейной системы «Project Asya» 13](#_Toc8977415)

[1.3.3 Обзор жюрейной системы от Hackerdom 15](#_Toc8977416)

[1.3.4 Обзор жюрейной системы от Facebook 17](#_Toc8977417)

[1.3.5 Обзор жюрейной системы SchoolCTF 19](#_Toc8977418)

[1.3.6 Итоговая сравнительная таблица функционала обозреваемых систем 21](#_Toc8977419)

[1.4 Список планируемых доработок для жюрейной системы SchoolCTF 22](#_Toc8977420)

[1.5 Постановка задачи 23](#_Toc8977421)

[СПЕЦИАЛЬНЫЙ РАЗДЕЛ 25](#_Toc8977422)

[2.1 Особенности функционала жюрейной системы SchoolCTF 25](#_Toc8977423)

[2.1.1 Особенности запуска платформы 25](#_Toc8977424)

[2.1.2 Структура жюрейной системы 27](#_Toc8977425)

[2.2 Особенности реализации дополнительного функционала 32](#_Toc8977426)

[2.2.1 Описание реализации выбранного дополнительного функционала. 32](#_Toc8977427)

[2.2.2 Описание реализации изменений в существующем функционале. 35](#_Toc8977428)

[2.2.3 Структура обновленной жюрейной системы 37](#_Toc8977429)

[ТЕХНИЧЕСКИЙ РАЗДЕЛ 39](#_Toc8977430)

[3.1 Описание выбранного программного обеспечения для разработки 39](#_Toc8977431)

[3.1.1 Выбор языка программирования для разработки. 39](#_Toc8977432)

[3.1.2 Выбор СУБД для разработки 44](#_Toc8977433)

[3.1.3 Выбор среды разработки 45](#_Toc8977434)

[3.2 Объектное моделирование жюрейной системы SchoolCTF 46](#_Toc8977435)

[3.2.1 Обоснование технологии проектирования 46](#_Toc8977436)

[3.2.1 Диаграмма прецедентов (use-case diagram) 48](#_Toc8977437)

[3.2.2 Диаграмма последовательности (sequence diagram) 50](#_Toc8977438)

[3.2.3 Диаграмма деятельности (activity diagram) 52](#_Toc8977439)

[3.3 Обзор функциональности внесенных доработок в систему SchoolCTF 55](#_Toc8977440)

[3.3.1 Отображение дополнительной информации о задачах 55](#_Toc8977441)

[3.3.2 Составление краткой инструкции по разворачиванию системы 57](#_Toc8977442)

[3.3.3 Возможность редактирования информации на главной странице через панель администратора. 58](#_Toc8977443)

[ЭКОНОМИЧЕСКИЙ РАЗДЕЛ 60](#_Toc8977444)

[4.1 Планирование разработки программного продукта с построением графика 60](#_Toc8977445)

[4.1.1 Определение трудоемкости и продолжительности работ по созданию автоматизированной системы управления запасами готовой продукции 60](#_Toc8977446)

[4.1.2 Построение ленточного графика проведения исследования 65](#_Toc8977447)

[4.2 Расчет сметы затрат на разработку программного продукта 63](#_Toc8977448)

[4.3 Расчет основных технико-экономических показателей и эффективности использования программного продукта 69](#_Toc8977449)

[4.3.1 Характеристика решения задачи 69](#_Toc8977450)

[4.3.2 Определение трудоемкости обработки информации по базовому и проектному вариантам 69](#_Toc8977451)

[4.3.3 Расчет капитальных вложений 70](#_Toc8977452)

[4.3.4 Расчет годовых текущих затрат 72](#_Toc8977453)

[4.3.5 Расчет показателей экономической эффективности 74](#_Toc8977454)

[ЗАКЛЮЧЕНИЕ 78](#_Toc8977455)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 79](#_Toc8977456)

# **ВВЕДЕНИЕ**

**Актуальность работы.** Экспериментальное обучение – это современный метод исследования психолого-дидактических проблем. На данный момент различается два вида экспериментального обучения: индивидуальный обучающий эксперимент, и коллективное экспериментальное обучение.

Индивидуально обучающий эксперимент существует уже довольно продолжительный срок и основательно вошел в научный процесс. Данный вид эксперимента позволяет определить существующие черты у индивида, а также сформировать новые, направляя обучающегося в процессе исследования. Благодаря экспериментам такого типа можно наблюдать и положительно воздействовать на процесс усваивания и запоминания новой информации.

Коллективное экспериментальное обучение как правило проводится на контрольной группе обучающихся. За контрольную группу берется группа в детском саду, школьный класс, группа студентов, и т.д. Данные исследования проводятся с целью углубленного изучения процесса обучения человека при условии различных, влияющих на процесс обучения, обстоятельств извне для каждого обучающегося.

В совокупности, основной целью экспериментального обучения является выявление наиболее результативного способа усвоения информации с наименьшим количеством негативных последствий и рисков. Внедрение экспериментальных методик обучения или контроля знаний позволяет обеспечить систематичность, регулярность и преемственность необходимого воздействия, а также, дает возможность анализировать полученные данные для дальнейшей обработки и усовершенствования эксперимента.

Так как экспериментальное обучение отличается от обычного процесса обучения, оно вызывает интерес у людей, и получает больший отклик, нежели стандартные обучающие практики. Причем, чем необычнее, но легче и спокойнее эксперимент, тем, как правило, больше обучающихся (или их представителей) соглашаются на подобные методики. Внедрение новых методик образования позволяет разнообразить учебный процесс, давая возможность мозгу человека переключиться от обычного способа получения информации на необычный, что является одним из важных факторов в её успешном запоминании.

Такой способ обучения так же распространяется и в мире информационных технологий. В текущих реалиях IT-технологии прогрессируют с необычайной скоростью, поэтому для них необходимо обучать новых специалистов с использованием актуальных данных. Совместить прогресс в IT-сфере, который шагает широким шагом, и обыденные методики образования с каждым годом всё сложнее. На помощь в таком случае приходит экспериментальное обучение.

В IT-сферу новые методики обучения вводятся постоянно, так как данная область непрерывно растет и развивается. Именно с этим фактором и связана такая популярность экспериментального обучения в мире информационных технологий. Интерактивные модели, новые интересные задачи, обучение поиску нестандартных решений: всё это вводится для того чтобы сделать процесс обучения более продуктивным, захватывающим и приятным.

Не секрет, что обучение с использованием соревновательного процесса дает достаточно хорошие результаты, так как именно желание победить способствует более хорошему получению и усвоению материала у молодого поколения. Для этого и были созданы соревнования Capture the Flag, или, как их сокращенно называют, CTF. CTF – это соревнования по информационным технологиями и информационной безопасности, которые устраиваются ежегодно различными группами людей с целью выявления лучших специалистов в данных областях. Так же, путем участия в данных соревнованиях, обучающиеся стремительно улучшают свои навыки в IT-сфере. Такой прогресс достигается за счет частого чередования процесса поиска новой информации и применения ее на практике.

Для проведения данных соревнований используются специальное ПО, названное CTF-системами или жюрейными платформами. Данное ПО – это набор функций и средств, которые необходимы для корректного проведения соревнований. Так же, каждая система обладает своими достоинствами и недостатками, которые и отличают эти системы друг от друга.

Во введении работы обоснована актуальность выбранной темы выпускной квалификационной работы, определена цель и задачи работы.

В исследовательском разделе описана методика жюрейства CTF-соревнований, проведен обзор существующих систем, выполнен сравнительный анализ выбранного ПО, выбрана система для доработки, формулирован список доработок для платформы, определена цель выпускной квалификационной работы.

В специальном разделе описаны алгоритмы работы текущей версии выбранной жюрейной системы и алгоритмы работы новой её версии, обозначена структура обеих версий приложения, подробно описана структура базы данных приложения и вносимых в неё изменений.

В технологическом разделе описано обоснование выбора программных средств для реализации поставленной цели, с помощью Case-технологий описан функционал жюрейной системы, а также, продемонстрирован обновленный функционал системы.

В экономическом разделе проведён анализ стадий проведения исследований, рассчитано время, требующееся на разработку и тестирование информационно-измерительной системы для жюрейной системы «SchoolCTF» определены затраты и капитальные вложения на разработку системы, приведены основные технико-экономические показатели проекта.

В заключении сделаны основополагающие выводы по результатам проведенной работы.

# **ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ**

## Обзор методики проведения и жюрейства CTF-соревнований

Существующие CTF-соревнования бывают двух видов:

* attack defense;
* task-based.

Attack defense это тип соревнований, в котором участникам необходимо получить доступ к серверу противника, при этом удержав оборону своего сервера. Соревнования данного вида не совсем подходят для обучения, как правило, в них участвуют игроки, которые уже имеют достаточно серьезные знания в данной области.

Task-based – это тип соревнований, когда игрокам предоставляется список заданий, которые нужно решить. Решением является флаг, который необходимо ввести в специальное поле. За каждое решенное задание (таск) даются очки, сумма очков за каждый таск разная, зависит она от сложности задачи. Побеждает та команда, которая набирает наибольшее количество баллов за время, отведенное на соревнование. Данный вид соревнований отлично подходит для обучения, так как для решения заданий участникам необходимо самостоятельно добывать пласты недостающих знаний. В связи с этим, работа проведена именно на базе систем для task-based -соревнований.

Процесс проведения состязаний достаточно прост: игроки регистрируются в системе, присоединяясь к одной из команд. Далее, на платформе появляются задания, или, как их еще называют, таски или квесты. На усмотрение организаторов есть два вида отображения квестов:

* команды получают доступ ко всем заданиям сразу, и выбирают задания для решения самостоятельно;
* команды получают доступ к одному или нескольким заданиям, а остальные открываются для команд постепенно, по мере решения ими предыдущих задач.

За каждое решенное задание командам начисляются очки (баллы). Процесс начисления тоже может быть двух видов:

* количество баллов за решение задачи фиксированное, и начисляется командам не зависимо от времени и порядка решения ими задач;
* количество баллов за решение задачи динамически изменяется в зависимости от количества команд, которые решили эту задачу. Т.е. если за определенный промежуток времени задачу решила только одна команда, она получает 100% очков, если их было две, то каждая команда получает по 50% очков, и так далее.

Второй способ начисления очков как правило используется в соревнованиях, где доступ к заданиям открывается постепенно.

Соревнования проводятся определенное количество часов. После того, как время, отведенное на проведение соревнования, выходит, командам закрывается доступ к заданиям. Далее, организаторы состязаний с помощью средств аналитики оценивают, какая команда победила.

В жюрейных системах могут применяться следующие средства аналитики:

* таблицы с результатами очков команд;
* таблицы с количеством решенных командами задач;
* график, отображающий количество решенных командами задач;
* график, отображающий время, затраченное командами на поиск решения задач.
* И т.д.

Выбор, какие средства аналитики использовать в состязаниях, лежит на организаторах соревнований.

## 1.2 Обзор функционала жюрейных систем для соревнований типа tack-based.

Функционал жюрейных систем для соревнований типа tack-based может различаться, есть множество дополнительных функций, которые могут присутствовать в системах, однако, основной перечень необходимых функций, без которых не сможет обойтись ни одна платформа.

1. Редактирование пользователей и команд.

Под редактированием пользователей подразумевается возможность корректировки его анкетных данных, возможность его добавления и удаления, а также зачисления его в определенную команду. Так же, можно редактировать самих команды (название, численность, логотип и т.д.).

Пользователям также присваиваются права (участник/администратор).

Ко всему прочему, организаторам доступна еще и возможность временной блокировки определенного игрока или команды.

1. Редактирование заданий.

Так как основой CTF соревнований являются таски, которые выдаются участникам, у администраторов платформы есть возможность добавления квестов, их удаления, и редактирования их свойств. Также, квесты можно отсортировать по тематике заданий, или по тегам, которые им выставляются.

Не лишней является функция загрузки квеста в систему из архива.

1. Наличие оповещений.

Оповещения от организаторов являются неотъемлемой частью жюрейной системы, с помощью них участники получают уведомления о начале и окончании игры, о появлении новых заданий, информацию о решенных заданиях командой участника. Кроме оповещений для всех игроков и для конкретной команды, есть оповещения для конкретного участника или нескольких участников. Такая функция очень удобна в случае, когда организатору необходимо донести информацию до одного или нескольких участников, потому что не нужно ставить в известность всю команду или всех игроков.

1. Установка времени начала и окончания игры.

Данный пункт один из самых важных: каждое соревнование проходит в четко оговоренных временных рамках, и чтобы не допустить возможность досрочного получения доступа к заданиям (или, наоборот, поздней сдачи), для каждого этапа соревнований выставляется время начала и окончания. До и после этого отрезка времени все задания этапа можно заморозить.

1. Наличие панели администратора.

Для организаторов соревнований администрирование большого количества команд не должно составлять большой труд. По возможности, панель администратора делается максимально удобной, функциональной и интуитивно понятной.

1. Просмотр аналитики соревнований.

Для наблюдения за участниками в режиме on-line в системах присутствуют средства аналитики. Они позволяют осуществлять контроль за скоростью решения заданий, просмотр успехов отдельно взятой команды или участника. Аналитика обычно выстраивается в виде графика.

## 1.3 Обзор существующих жюрейных систем

### 1.3.1 Обзор жюрейной платформы «CTFd»

CTFd – самая распространенная платформа для соревнований Capture The Flag. Она ориентирована на простоту использования и расширенную кастомизацию. В состав платформы входит всё необходимое для развертывания, настройка системы происходит с помощью множества тем.

CTFd – продукт группы разработчиков из США. Это полностью открытое программное обеспечение, его исходный код расположен на GitHub. Данную систему можно самостоятельно развернуть на хостинге, однако, для потребителей, которые не хотят заниматься такими вещами как настройка и развертывание, команда CTFd может предложить вариант «под ключ»: арендовать хостинг с развернутой на нем платформой за определенную сумму в месяц (в зависимости от перечня функций хостинга).

Платформа имеет множество функций, начиная от основных, описанных в предыдущем разделе, и заканчивая уникальными только для неё.

Из дополнительных функций можно перечислить:

1. Автоматическую защиту от brute force.

Данная особенность системы очень полезна для соревнований типа tack based, так как участники могут начать «перебирать» ответы в поисках верного.

1. Автоматическое скрытие заданий.

Эта функция срабатывает тогда, когда у участника (или у команды) закончились попытки для решения задания.

1. Временное замораживание статистики.

Может использоваться организаторами соревнований для различных целей: например, для того чтобы команды не знали предварительные результаты своих соперников.

1. Отложенная установка временных рамок соревнования.

Обеспечивает удобство для организаторов. Не обязательно нажимать на кнопку для старта состязаний - таймер сработает в определенный час и соревнование начнется автоматически.

1. Использование языка разметки Markdown.

Данная особенность упрощает управление системой, давая возможность легко редактировать контент на страницах.

1. Интеграция с MajorLeagueCyber.

Данная система тесно связана с MajorLeagueCyber (MLC). MLC – это средство отслеживание статистики для CTF, позволяющее отслеживать и планировать события.

1. Возможность переключения способа скорринга (начисления очков) с статического на динамический.

Как было уже ранее сказано в п.1.1 исследовательского раздела, существует два способа начисления очков. И CTFd дает возможность организаторам переключаться между этими способами.

1. И другие.

Установить систему можно несколькими путями. Разработчики в первую очередь советуют использовать Docker, так как это самый простой и удобный способ. Но если такой вариант не подходит, можно использовать сервер отладки Flask.

Для удобства использования разработчиками составлена краткая и подробная инструкция по развертыванию и использованию CTFd.

Как бы хороша система не была, в ней всегда будут какие-то недостатки. К недостаткам системы CTFd можно причислить:

1. Слишком сильная распространённость.

Данная система очень известна, её используют многие организаторы соревнований, а оттого сами состязания лишаются своей «изюминки» в лице интересной платформы.

### 1.3.2 Обзор жюрейной системы «Project Asya»

Project Asya – система для проведения CTF-соревнований, произведенная на свет российскими разработчиками. Его исходный код расположен в репозитории на GitHub, как и у предыдущей разработки.

В отличие от CTFd, Project Asya не является популярной жюрейной системой, которой пользуются тысячи пользователей, она малопопулярна, но не уступает по производительности своим аналогам. Продукт был создан разработчиками команды Keva (имя система унаследовала от одной из участниц команды), и использовался для проведения соревнований UralCTF.

У этой системы есть ряд преимуществ, благодаря некоторым из них она даже может обойти свои более известные аналоги:

1. Простой и удобный интерфейс.

Это было отмечено как и организаторами соревнований UralCTF, так и его участниками. Систему все чаще выбирают для проведения состязаний из-за её интуитивно понятного интерфейса и незагруженности.

1. Простая система мониторинга

Является несомненным преимуществом в проведении соревнований. Бонусом к данному пункту является то, что система мониторинга динамичная, и позволяет показывать прогресс участников в режиме online.

1. Быстрая установка.

Project Asya показывает отличные результаты в развертывании: процесс запускается всего одной командой и длится всего две минуты (что подтвердили организаторы соревнований).

1. Стабильность.

Является главным критерием любой системы: так как система поддается большой нагрузке, она должна уметь справляться с неравномерным напряжением, при этом не теряя в производительности.

1. Доступность.

Как уже было сказано выше, Project Asya совершенно бесплатна и доступна всем желающим для использования. Эта особенность дает возможность тренироваться в развертывании и управлении системой задолго до проведения собственных соревнований.

1. И другие.

Эту систему, как и любую другую, не обошли стороной недостатки. Их у данного проекта, к счастью, немного:

1. Отсутствие демонстрационной версии

Так как Project Asya была сделана не для коммерческого использования, и в основном её разворачивают как раз те же люди, которые ее разрабатывали, в демонстрационной версии для них она не нуждается. Но стороннему обывателю перед развертыванием системы было бы полезно увидеть ее демо-вариант, чтобы понять, подходит она или нет.

1. Малопопулярность

Платформа не так известна, как ее аналоги, и это может создать неудобства. К ней есть краткая инструкция, однако, она очень малоинформативна. Да, разработчики системы заявляют, что, при необходимости, они могут оказать помощь и поддержку в работе с системой, однако, отсутствие отзывов о продукте, а также наличие более подробной инструкции немного омрачает вышеперечисленные плюсы.

Тем не менее, будучи малоизвестной системой, Project Asya имеет очень хорошую поддержку в лице своих разработчиков. Во время разработки и доведения до совершенства данного проекта, система получала регулярные обновления, которые можно отследить по обновлениям на GitHub. Платформа была спроектирована по аналогии со своими зарубежными конкурентами, с ориентиром на простоту и удобство, но она не лишена своей уникальности и изящности.

### 1.3.3 Обзор жюрейной системы от Hackerdom

Hackerdom – русская команда, образовавшаяся в 2005 году на математико-механическом факультете Уральского государственного университета. Она развивается и по сей день, и в неё входят студенты самых разных направлений и курсов.

Команда занимается изучением компьютерной безопасности, организацией и участием в соревнованиях CTF. Поначалу, молодые хакеры только участвовали в соревнованиях, но затем начали понемногу организовывать свои. В настоящее время студенты организовывают всероссийские и международные соревнования QCTF, RuCTF и RuCTFE. Так же, Hackerdom проводит курсы для тех, кто хочет играть в CTF, но не обладает достаточными знаниями. Участие в курсах абсолютно бесплатно и не требует сильно специфических и углубленных знаний.

Использование сторонней жюрейной системы для соревнований перестало подходить команде, поэтому вскоре студенты создали свою собственную.

Основными разработчиками системы стали Михаил Гнедашев и Александр Гейн.

Интерфейс системы прост и незамысловат: его можно отредактировать под нужды участников, но изначально он сделан в черно-белых тонах. Нерешенные задания выделены синим цветом, решенные – зеленым. Есть доступ к своему профилю, квестам, а также к турнирной таблице.

Платформа от Hackerdom находится в открытом доступе, её исходный код выложен в репозитории на GitHub. Там же находятся и модификации системы для различных проведенных соревнований.

Одним из самых главных плюсов данной системы является то, что её можно протестировать, не раскатывая на своем сервере. Это дает ей неоспоримое преимущество перед остальными системами, так как все мы с большей вероятностью отдадим предпочтение тому сервису, который можем самостоятельно опробовать без лишних усилий. На сервис уже добавлено очень большое количество заданий (с прошлых CTF), которые используются в обучении школьников старших классов и студентов.

К другим полезным функциям системы можно отнести:

1. Стабильность

Была отмечена как разработчиками, так и участниками. За всё время проведения множества соревнований, система постоянно улучшалась, и с каждым годом становилась всё более стабильной.

1. Интуитивность в использовании

Все функции системы просты и понятны, она не нагружена лишними малоиспользуемыми возможностями ПО. Одновременно с этим, в ней присутствуют главные опции, перечисленные выше, что делает её приятной для использования.

1. Поддержка разработчиков

Несомненно, наиболее успешно используется та система, разработчики которой поддерживают свой продукт. Детище Hackerdom регулярно обновляется, для каждых соревнований выпускается своя, немного модифицированная, полноценная версия системы, которую потом разработчики любезно выкладывают на GitHub. Там же находятся и задания с прошедших соревнований, который можно решать, тем самым тренируясь, и даже использовать в своих собственных состязаниях.

1. Доступность.

Аналогично предыдущей системе, данная платформа обладает свободной лицензией и может свободно разворачиваться любыми пользователями.

1. И другие.

Как всегда, не обошлось и без недостатков. Их у системы немного:

1. Отсутствие инструкции по развертыванию

Несмотря на то, что система открытая и довольно известная, у неё отсутствует даже краткая информация о том, как правильно её установить. Это создает небольшие неудобства, так как новичку в данной области порой приходится довольно долго разбираться с настройкой.

Невзирая на недостатки, жюрейная система от Hackerdom является популярной и хорошей системой, которые пользуются многие организаторы для проведения соревнований.

### 1.3.4 Обзор жюрейной системы от Facebook

Facebook это популярная социальная сеть, основанная в 2004 году. Позже, так стала называться и владеющая ею компания. Поначалу, дальше сайта для общения между людьми компания не двигалась, но потом она разрослась, начав занимать всё большие IT-сферы. Так она поглотила несколько крупных сервисов, таких как Instagram и What’sApp.

Параллельно с развитием основной компании и поглощением дочерних, Facebook занялся развитием в сфере проведения соревнований Capture the Flag. Изначально соревнования проводились только на уровне колледжа, а целью компании было помочь детям школьного возраста открыть для себя мир информационных технологий и компьютерной безопасности. Со временем аудитория на соревнованиях расширилась, и Facebook стал организовывать соревнования уже по всему миру.

Все состязания уже с самого начала проводились компанией на специальной CTF-платформе, которая был разработана в Facebook для этой цели. Она сочетала в себе преимущества других платформ, ловко избегая популярных недостатков, а дополнительно обладала уникальным дизайном, который не был замечен больше ни на одной жюрейной системе.

Как и у всех других платформ, у системы от Facebook (сокращенно ее называют FBCTF) есть стандартные функции, такие как:

* Регистрация команд и участников
* Размещение информации о заданиях
* Система подсказок для участников
* И т.д.

Из преимуществ платформы от Facebook можно выделить:

1. Уникальный интерфейс

Такого интерфейса платформы, как у FBCTF больше нет нигде. Будучи платформой от разработчиков популярной социальной сети, Facebook смог удивить пользователей красотой и практичностью системы. В отличие от других жюрейных систем, у FBCTF тёмный интерфейс в неоновых тонах (преобладает чёрный, индиго и красный). Сами задания размещаются не в простой табличке, а на карте мира: каждый квест находится в своей стране. На этой же карте можно увидеть, какие еще команды решили задания. По бокам от карты, как в популярных играх, находятся меню с список других команд, списком решенных заданий и многим другим.

1. Доступность

FBCTF является бесплатным продуктом. Весь его исходный код выложен на GitHub.

1. Наличие демонстрационной записи

Facebook хорошо позаботились о том, чтобы у пользователей их платформы было четкое представление о продукте. Поэтому, для краткого ознакомления с интерфейсом и возможностями FBCTF разработчики сделали, пусть и короткую, но довольно информативную видеозапись использования системы.

1. Подробная инструкция по установке

Немногие платформы для проведения CTF-соревнований могут похвастаться наличием инструкции по установке для пользователей. Да и обычно она довольно короткая. Facebook в этом плане превзошел своих конкурентов и для своей платформы сделал подробную инструкцию по установке, которая прилагается к информационному файлу в репозитории на GitHub.

Несмотря на все преимущества системы, её так же не обошли стороной недостатки:

1. Необходимость высокопроизводительных комплектующих

В отличие от других систем, FBCTF является довольно требовательным ПО, что является обратной стороной медали для всех вышеперечисленных достоинств.

1. Усложненная установка.

У FBCTF установка производится не в «два клика» и не за «две минуты», как у её конкурентов. Однако, она всё равно не является слишком сложной для пользователей.

В целом, FBCTF довольно сильно отличается от своих конкурентов, начиная интерфейсом и заканчивая кодом. Тем не менее, он является довольно хорошей платформой для проведения соревнований Capture the Flag.

### 1.3.5 Обзор жюрейной системы SchoolCTF

Платформа для проведения CTF-соревнований SchoolCTF была создана русскими разработчиками в 2014 году для проведения конкурса SchoolVolgaCTF среди учеников школ. Данный продукт бесплатный и свободно распространяемый, его исходный код выложен на ресурсе GitHub

Данное ПО задумывалась как легковесная и простое в управлении жюрейная система, которую под силу развернуть любому человеку, даже тому, у кого нет опыта в проведении подобных состязаний. SchoolCTF является не самой известной платформой, тем не менее она обладает рядом преимуществ:

1. Простота интерфейса

Система действительно несложная и удобная в использовании как организаторами, так и участниками. Интуитивно понятный интерфейс, малое количество разделов, отсутствие излишней нагруженности делают ее подходящим вариантом для проведения соревнований среднего масштаба.

1. Доступность

Данная платформа является полностью открытым ПО, его исходный код выложен на GitHub

Данную систему, как и любую другую, не обошли стороной недостатки:

1. отсутствие иных видов аналитики кроме табличных.

В данной жюрейной системе из способов аналитики соревнований существуют только некорректно работающая таблица с результатами решения участниками задач. Это неудобно для организаторов соревнований.

1. Отсутствие отображения дополнительной информации в заданиях.

Под дополнительной информацией подразумеваются второстепенные данные о задачах (количество команд, которые решили квест, автор задания и т.д.). Этот недостаток лишает участников соревнований полного представления о задании, а значит, не дает возможности выстроить более достоверную стратегию игры.

1. Отсутствие возможности отображения всплывающих уведомлений с произвольным текстом.

Из всплывающих уведомлений в списке своих опций SchoolCTF имеет только уведомления от самой платформы (информация о неверном логине и пароле, уведомления об ошибке в системе и т.д.). Соответственно, организаторы соревнований не имеют возможности внутри системы оповестить участников о каких-либо изменениях или новой информации в игре.

1. Отсутствие формы самостоятельной регистрации команд

Данная функция существенно упрощает процесс регистрирования участников, так как команды могут делать это самостоятельно, не возлагая эту задачу на организаторов.

### 1.3.6 Итоговая сравнительная таблица функционала обозреваемых систем

В таблице 1 приведен список жюрейных систем (первый столбец таблицы) и список функций (первый столбец таблицы) для последующего выбора системы для доработки:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | CTFd | Project Asya | Жюрейная платформа от команды Hackerdom | FBCTF | SchoolCTF |
| Система свободно распространяемая | да | да | да | да | Да |
| Простота интерфейса | да | да | да | нет | да |
| Наличие инструкции по установке | да | нет | нет | да | нет |
| Различные виды аналитики (таблицы, графики и т.д.) | да | нет | да | да | нет |
| Форма самостоятельной регистрации | да | да | да | да | нет |
| Отображение доп. информации в заданиях | да | да | да | да | нет |
| Наличие всплывающих уведомлений | да | да | да | да | нет |

Таблица 1. Сравнение функционала обозреваемых жюрейных систем (узнать каким шрифтом форматируются подписи к таблицам)

Путем сравнения выбранных жюрейных систем, можно заметить, что больше всего по количеству недостатков лидирует жюрейная платформа SchoolCTF. Таким образом, именно её решено выбрать в качестве системы для доработки в рамках данной работы.

## 1.4 Список планируемых доработок для жюрейной системы SchoolCTF

Путем сравнения системы SchoolCTF с её конкурентами, было выявлено, что в платформе отсутствуют функции, которые, в большинстве своем, присущи её аналогам. Исходя из данного факта, для доработки были кратко сформулированы следующие опции:

1. отображение дополнительной информации задачах:

2.1 рейтинг сложности задачи;

2.2 список названий команд, решивших задачу;

1. вывод аналитики в виде графиков;
2. составление краткой инструкции по разворачиванию системы;
3. отображение всплывающих уведомлений с текстом, заданным организаторами;
4. возможность самостоятельной регистрации команд участниками.

Так же, для повышения функциональности и уникальности платформы, были выбраны следующие опции для доработки:

1. включение и отключение возможности регистрации через панель администратора;
2. возможность использования online и offline версий системы.
3. возможность редактирования информации на главной странице через панель администратора.

Таким образом, путем сравнения различных жюрейных систем, был сформулирован список опций для доработки в функционале в выбранной для этой цели платформе.

## Постановка задачи

Целью выпускной квалификационной работы является доработка одной из существующих систем путем увеличения количества инструментов системы и улучшением текущего функционала.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Провести обзор методики проведения и жюрейства соревнований.
2. Рассмотреть функционал жюрейных систем.
3. Выявить достоинства и недостатки в нескольких выбранных системах.
4. Выделить систему для доработки.
5. Сформулировать список исправлений и доработок.
6. Описать алгоритмы работы новых и обновленных функций системы.
7. Внедрить новый функционал.
8. Продемонстрировать новый функционал выбранной для доработки системы.

# **СПЕЦИАЛЬНЫЙ РАЗДЕЛ**

## Особенности функционала жюрейной системы SchoolCTF

### 2.1.1 Особенности запуска платформы

Жюрейная система SchoolCTF была разработана как легковесное, простое в управлении, и обладающее приемлемым функционалом программное обеспечение. На выходе же получилась платформа, которая соответствует лишь минимальному количеству необходимых для работы функций.

Для успешного проведения соревнований необходимо, чтобы участники состязания могли подключиться к жюрейной системе, на которой происходит проведение соревнования. Было два варианта реализации данной задачи:

* арендовать web-адрес и загрузить туда систему;
* использовать виртуальную машину для запуска сервера с уже загруженной на него системой.

Достоинством первого варианта являлась простота: не было необходимости делать дополнительные действия для запуска системы, достаточно было бы просто зайти на нужный веб-сайт.

Однако, у данного метода был ряд своих недостатков, который вынудил отказаться от него в пользу второго варианта.

* За веб-хостинг и доменное имя сайта нужно платить.

Причем, чем глобальнее соревнование, тем больше необходимо было бы платить за сайт, так как объем места на сервере, объем базы данных и мощность сервера меняется в зависимости от цены.

* Нельзя проводить одновременно несколько соревнований на данной платформе.

Чтобы сделать платформу популярной, нужно, чтобы её мог использовать любой желающий для проведения состязаний в любое время. Если бы сервис находился в глобальной сети Интернет, использование его для проведения сразу нескольких соревнований одновременно в разных точках мира было бы невозможным или сильно затруднённым. Это сделало бы платформу непопулярной, и не дало бы ей развития.

* Нельзя закрыть доступ участникам к сети Интернет.

Для проведения состязаний порой необходимо ограничить участникам доступ к Всемирной Паутине, оставляя лишь возможность подключения к другим компьютерам в локальной сети. Использование стороннего ресурса в данной ситуации повлекло бы за собой дополнительные сложности в настройке оборудования, а значит, большую вероятность отказа оборудования в период проведения соревнований.

* Ненадежность.

Веб-сайты, конечно, бывают недоступны не так часто, но такое всё же происходит, и произойти это может в любой, даже самый неподходящий момент. Поэтому, использование внешней платформы для размещения на ней системы было бы менее надежным способом гарантии её работоспособности в момент соревнований. Дополнительно, доступ к ней напрямую зависел бы от доступа к глобальной сети Интернет, в то время как использование своего, собственно собранного локального ПО, гарантирует доступ к сервису ровно на столько, на сколько необходимо.

Выбрав второй вариант для реализации задачи, были выявлены и его недостатки:

* Повышенная сложность в использовании

Запустить виртуальную машину, запустить на ней сервер, узнать необходимый для подключения ip-адрес является большим количеством действий, чем просто вход на веб-сайт, и может быть сложным для некоторых пользователей.

* Необходимость дополнительного ПО

Для развертывания виртуальной машины в системе необходимо наличие специализированного ПО (VirtualBox, VMWare Workstation и др.), которое не всегда является бесплатным. Для его использования порой необходимо купить лицензию, так как использование пиратского программного обеспечения является незаконным, а стоимость такого ПО может не подходить организаторам. Этот факт осложняет проведение соревнований.

* Необходимость повышенных технических требований

Для развертывания виртуальной машины так же есть необходимость наличия достаточного технического оснащения хостовой машины, чтобы была возможность выделения определенного количества памяти на жестком диске и оперативной памяти. Этот вариант подходит для тех пользователей, техническое оснащение компьютеров которых среднее или выше среднего, что не всегда является таковым в текущих реалиях.

Несмотря на наличие недостатков, данный вариант реализации задачи стал более подходящим для проекта.

Версия проекта без доработок была сделана с использованием сервера Apache, который запускался из специально собранной виртуальной машины с ОС Debian 9.0. Этот вариант являлся оптимальным для проведения соревнований, так как система использовалась людьми с достаточным количеством знаний для её успешного развертывания.

* + 1. Структура жюрейной системы

Само приложение жюрейной платформы (её веб-интерфейс) написан на языке программирования Python, с использованием фреймворка Django. Таким образом, весь проект имеет следующую структуру:

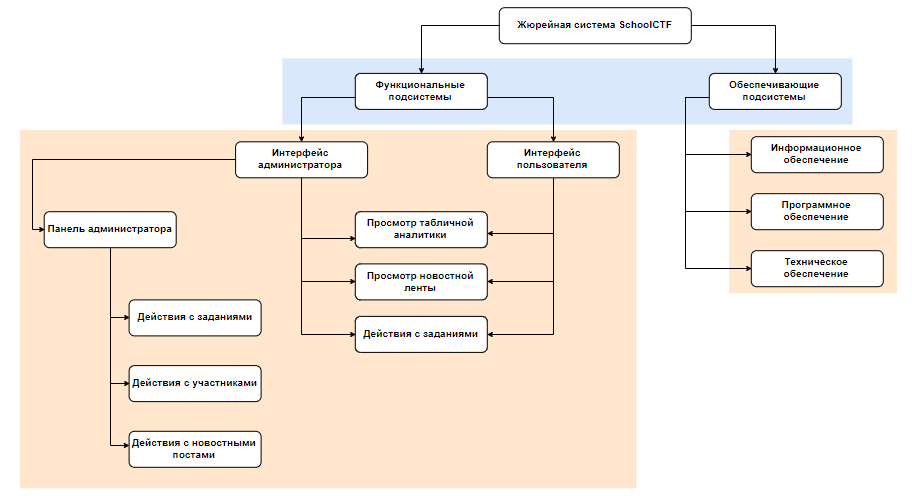


Рисунок 1. Структура жюрейной системы SchoolCTF

* Функциональные подсистемы
  + Панель администратора
* Действия с заданиями
* Действия с участниками
* Действия с новостными постами
* Интерфейс пользователя
  + Просмотр табличной аналитики
  + Просмотр новостной ленты
  + Ввод ответа на задание
* Обеспечивающие подсистемы
* Информационное обеспечение
* Программное обеспечение
* Техническое обеспечение

Каждый элемент данной структуры является компонентом жюрейной системы SchoolCTF и необходим для корректной её работы.

Функциональные подсистемы включают в себя ряд подсистем, который охватывает выполнение конкретных функций в работе жюрейной платформы.

**Интерфейс администратора**

Функционал интерфейса администратора располагает следующими опциями:

1. Действия с заданиями
2. Действия с участниками
3. Действия с новостными постами

1. Действия с заданиями

Действия с заданиями следующие:

* Добавление
* Редактирование
* Удаление

Для того, чтобы иметь возможность реализации данного функционала, в SchoolCTF имеется база данных, в которой есть запись, содержащая следующие поля:

* Название
* Текст задания
* Ответ

2. Действия с пользователями

Действия с пользователями имеются следующие:

* Добавление пользователя
* Редактирование данных пользователя
* Удаление пользователя

Для реализации поставленных функций в базе данных SchoolCTF имеется таблица с информацией об участниках. В таблице есть записи, в которых содержатся следующие поля:

* Название команды
* Логин команды
* Пароль участника
* Флаг администратора
* Флаг активности

3. Действия с новостями

Действия с новостями имеются следующие:

* Добавление новости
* Редактирование новости
* Удаление новости

Для реализации заявленного функционала в базе данных SchoolCTF имеется таблица, в которой содержится информация о новостной ленте. В таблице имеются записи, в которых содержатся следующие поля:

* Заголовок новости
* Текст новости

**Интерфейс пользователя**

Функционал интерфейса пользователя располагает следующими опциями:

1. Просмотр табличной аналитики
2. Просмотр новостной ленты
3. Действия с заданиями

1. Просмотр табличной аналитики

Для того, чтобы иметь возможность просмотра табличной аналитики, в SchoolCTF имеется сущность SolvedTasks. Данные для табличной аналитики берутся из таблицы с решенными заданиями. В таблице имеются записи о решенных командами заданиях, в которой содержатся следующие поля:

* Команда, решившая задание
* Наименование задания
* Время решения задания командой

2. Просмотр новостной ленты

Просмотр новостной ленты возможен благодаря выводу полей записей новостей из соответствующей таблицы, описанной в действиях с новостями в интерфейсе администратора.

3. Действия с заданиями

В списке действий с заданиями у пользователя имеются следующие действия:

* Просмотр задания
* Ввод ответа

Для реализации просмотра задания на веб-страницу выводится содержимое полей «Название» и «Текст задания» из таблицы с заданиями, описанной в действиях с заданиями в интерфейсе администратора.

Для реализации проверки правильности введенного ответа, введенный ответ сравнивается с ответом, записанным в базе данных в поле «Ответ» в таблице с заданиями, описанной в действиях с заданиями в интерфейсе администратора.

Обеспечивающие подсистемы включают в себя информационное, программное, техническое обеспечение.

Информационное обеспечение – совокупность форм документов, классификаторов, нормативной базы и реализованных решений по объемам, размещению и формам существования информации, применяемой в АС при ее функционировании.

Программное обеспечение – совокупность программ на носителях данных и программных документов, предназначенная для отладки, функционирования и проверки работоспособности АС.

Техническое обеспечение – совокупность всех технических средств, используемых при функционировании АС.

## Особенности реализации дополнительного функционала

### 2.2.1Описание реализации выбранного дополнительного функционала.

Путем сравнения функционала различных жюрейных систем, было выявлено, что наибольшее отставание по функционалу имеется именно у системы SchoolCTF, посему был сформулирован список функций для её доработки.

Каждая опция была сформулирована на основе наличия подобного функционала у других систем, а также путем выявления новых опций, которых нет в системах-конкурентах.

**Отображение дополнительной информации в задачах**

1. Рейтинг сложности задачи.

Данная информация о задаче поможет участникам в планировании очередности решения задач на соревновании. Если рейтинг задачи высок, то имеет смысл приступать к ней после разминки или решения более легких задач.

Для реализации данной подзадачи необходимо добавить в базу данных жюрейной платформы следующее поле в запись, которая содержится в таблице с задачами:

* Рейтинг сложности

Так же, необходимо обеспечить вывод информации о сложности задачи из базы данных на веб-страницу, где отображается список задач.

2. Список названий команд, решивших задачу.

Информация о командах, которые уже решили задание, может помочь командам: как известно, рейтинг сложности задания выставляется организаторами, а значит, он субъективен. Количество команд, которые уже решили задание, поможет участникам более точно понять, на сколько задание сложное, и стоит ли приступать к его решению немедленно.

Реализация данной подзадачи строится следующим образом: необходимо вывести на страницу с задачей таблицу, в которой присутствуют следующие данные:

* Номер (порядковый)
* Название команды

Название команды в данном случае вызывается из базы данных, путем запроса таблице с решенными заданиями. Запрос выделяет поля «задание» и «команда, решившая задание» из записей, которые содержатся в таблице. Далее, происходит сравнение: если название задачи из таблицы соответствует запрошенному, то в таблицу на веб-странице выводится наименование команды, которая решила задачу.

**Вывод аналитики в виде графиков**

Данная жюрейная система располагает только табличной аналитикой: списком команд и количеству набранных ими очков. Такая аналитика универсальна, но не совсем удобна, так как недостаточно наглядна. Для большей наглядности было решено добавить аналитику в виде графика, где указано, какая команда идет впереди, а какая пока не очень хорошо справляется с решением заданий.

Для создания аналитики в виде графика используется инструмент <Название. И пару слов о нём>

Для того, чтобы реализовать данную задачу, необходимо:

* Подключить инструмент <инструмент для создания аналитики>.
* Добавить пункт меню «аналитика в виде графика» в меню жюрейной системы.
* Обеспечить вывод созданного графика на соответствующую веб-страницу.

**Составление краткой инструкции по разворачиванию системы.**

Для человека, который даже впервые столкнулся с жюрейной системой, необходимо, чтобы её установка была простой и понятной. Для этого разработчиками пишутся инструкции по развертыванию системы в своей ОС.

Для решения данной задачи необходимо:

* Создать текстовую инструкцию по развертыванию системы.

**Отображение всплывающих уведомлений с текстом, заданным организаторами.**

Для оповещения участников соревнования о различных новостях удобнее всего использовать внутренние всплывающие уведомления, текст которых может задать администратор соревнований.

Проблема отсутствия всплывающих уведомлений легко решается подключением приложения django-notifications.

Django-notifications – это приложение, которое обеспечивает отображение уведомлений, которые отправляются через панель администратора.

Для реализации данной задачи необходимо:

* Подключить django-notifications.
* Обеспечить возможность управления приложением через панель администратора.

**Возможность самостоятельной регистрации участников**

В данной жюрейной системе отсутствует возможность регистрации участников самими участниками. Это действие выполняют организаторы соревнований: они вручную регистрируют каждого участника. Это неудобно. Предложенная доработка упростит работу организаторов, избавив их от лишних действий.

Для решения данной задачи хорошо подходит приложение django-registration

Django-registration – это приложение, которое обеспечивает возможность самостоятельной регистрации пользователей на веб-странице.

Чтобы реализовать данную доработку, необходимо:

* Подключить приложение django-registration
* Создать веб-страницу для регистрации пользователей
* Обеспечить работу приложения на веб-странице

### 2.2.2 Описание реализации изменений в существующем функционале.

**Включение и отключение возможности регистрации через панель администратора**

Соревнования проходят в определенный промежуток времени. Для упрощения управления соревнованиями поможет функция отключения регистрации по желанию организаторов соревнований. Когда регистрация отключается, никакая команда больше не может зарегистрироваться, а значит, соревнования можно считать открытыми.

Для реализации данной задачи необходимо:

* Создать в базе данных сущность для управления приложением django-registration.
* Создать поле типа boolean, с помощью которого можно управлять работоспособностью функционала регистрации участников.

**Возможность использования online и offline версий системы.**

Для повышения популярности и удобства жюрейной системы было пересмотрено решение использования SchoolCTF исключительно с помощью запуска виртуальной машины.

Для удобства пользователей SchoolCTF решено установить на веб-хостинг для использования системы online. Данное решение в разы упрощает начало работы с системой, избавляя организаторов соревнований от лишних действий.

Для реализации данной доработки необходимо:

* Скачать копию файлов приложения системы SchoolCTF с сервера, расположенного на виртуальной машине с ОС Debian 9.

С этой задачей прекрасно справляется приложение WinSCP (для ОС семейства Windows) или приложение CyberDuck (для MacOS).

Данные приложения подключаются к удаленному серверу и скачивают оттуда необходимые данные приложения.

* Упаковать приложение для запуска в docker-контейнер.

Docker – это приложение для автоматизации развертывания и управления приложениями в средах, с поддержкой контейнеризации.

Контейнеризация – метод виртуализации, при котором ядро операционной системы поддерживает несколько экземпляров пространства пользователя вместо одного.

Другими словами, docker обеспечивает автоматизацию развертывания приложения в любой linux-системе, избавляя пользователя от ручной настройки.

* Арендовать веб-сервер в облаке
* Арендовать домен
* Настроить DNS-запись домена на веб-сервер в облаке
* Установить docker-контейнер с приложением на веб-сервер в облаке.

Таким образом, при поступлении запросов на выбранное доменное имя, сервер будет посылать запрос в облачный веб-сервер, на котором установлен и развернут docker-контейнер с приложением.

Так же, для использования SchoolCTF в соревнованиях, где отсутствует доступ к сети Интернет, была оставлена версия жюрейной системы, которая разворачивается с помощью запуска виртуальной машины, на которую установлена данная система.

**Возможность редактирования информации на главной странице через панель администратора.**

В исходной версии данной системы редактирование кода главной страницы осуществлялось исключительно через редактирование кода приложения. Это является неудобным, трудоемким и требующим специализированных знаний процессом. Гораздо легче и удобнее редактировать главную страницу так же, как и страницу новостей: через запись в базе данных.

Для того чтобы реализовать данную задачу, необходимо:

* Создать в базе данных сущность, которая будет отвечать за добавление, изменение и удаление материалов на главной странице.
* Обеспечить вывод информации из базы данных на соответствующую веб-страницу.

### 2.2.3 Структура обновленной жюрейной системы

С учетом описанных доработок, обновленное приложение имеет следующую структуру:



Рисунок 2. Структура обновленной жюрейной системы SchoolCTF

Внесенные в систему изменения, а так же внесенные изменения в базу данных были описаны в п.2.2.1 специального раздела.

# **ТЕХНИЧЕСКИЙ РАЗДЕЛ**

## 3.1 Описание выбранного программного обеспечения для разработки

### 3.1.1 Выбор языка программирования для разработки.

К выполнению поставленной задачи можно подойти двумя путями:

* Переписать с нуля имеющийся проект
* Использовать готовую основу имеющегося проекта для последующих доработок

Рассмотрим первый вариант выполнения поставленной задачи.

Для качественного выполнения поставленной задачи необходим гибкий, современный язык программирования, который осуществляет поддержку следующих пунктов:

* серверных фреймворков для разработки сайта;
* использования HTTP-серверов;
* написания HTML-страниц с нуля.

Рассмотрим несколько подходящих языков программирования, являющихся самыми популярными для данной задачи в настоящее время:

* Java

Java — язык программирования, используемый, в основном, для разработки родных Android-приложений для смартфонов и планшетов, а так же для разработки ПО для ОС Windows. Популярность Java у разработчиков связана с простотой и надежностью языка, который обеспечивает долгосрочную совместимость написанных на нём продуктов. Для разработки web-страниц Java поддерживает интерфейс под названием Servlet. Он взаимодействует с клиентом посредством принципа запрос-ответ. Для реализации HTML-страниц в Java используется технология jsp, задача которой динамически генерировать веб – страницы. Она имеет как статические, так и динамические компоненты. Статической частью jsp являются исходные данные, которые сохранены в одном из текстовых форматов (обычно HTML). Краткая схема работы веб-программы на Java выглядит следующим образом:



Рисунок 3. Краткая схема работы веб-приложения, написанного на Java

* PHP

Язык программирования PHP был сконструирован специально для веб-разработки, что является его главным достоинством. Его код можно встраивать напрямую в HTML-страницу. Он отделяется от основного кода HTML-страницами специальными начальными и конечными тегами. Каждый PHP-скрипт обрабатывается на сервере и генерирует HTML-страницу, которая отправляется клиенту. Данный язык программирования является очень простым в освоении, однако, большинство действий, которые в других языках можно не писать «руками» в PHP необходимо прописывать вручную.

Для того чтобы PHP-скрипты корректно работали, необходимо подключать сервер к проекту. Как правило, разработчики подключают готовую сборку из сервера, самого языка PHP, и базы данных. Хорошим примером такого случая может служить Denwer – набор, где содержатся все инструменты для веб-разработчика.

Краткая схема работы сайта, написанного на PHP выглядит так (рисунок 3):



Рисунок 4. Краткая схема работы сайта, написанного на PHP

В качестве серверного фреймворка PHP поддерживает такой продукт как Zend Framework, который свободно распространяется и основывается на принципах MVC (Model-View-Controller). MVC – это схема разделения данных (архитектурный шаблон) пользовательского приложения таким образом, что каждый из трех его компонентов (модель, представление, контроллер) может модифицироваться независимо друг от друга.

Процесс работы архитектурного шаблона можно проиллюстрировать следующим изображением:



Рисунок 5. Иллюстрация работы архитектурного шаблона MVC

MVC является популярной и современной схемой разделения данных, которая используется в большом количестве фреймворков.

* Ruby

Ruby – это язык программирования, в котором соединяются части таких языков как Perl, Smalltalk, Lisp, Eiffel и Ada для того чтобы обеспечивать баланс между парадигмой функционального программирования и принципами императивного программирования. Данный язык, по словам его автора, «прост снаружи, но сложен внутри, совсем как человек». В данном языке всё является объектами. Его создатель оперировал убеждением о необходимости более мощного языка, чем Perl и более объектно-ориентированного, чем Python. Ruby так же очень гибкий и позволяет свободно менять его части, он отличается высокой переносимостью и работает на большинстве современных ОС.

Для разработки веб-приложений Ruby поддерживает фреймворк под названием Ruby on Rails. Он обеспечивает интеграцию с веб-сервером, сервером баз данных, является открытым ПО. Как и Zend Framework, Ruby on Rails реализует архитектурный шаблон MVC.

* Python

Python – это высокоуровневый язык программирования, который прежде всего ориентирован на простоту, читаемость кода, и повышение производительности разработчиков. Данный язык поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Он активно развивается, новые его версии выходят примерно раз в два года. Python не подвергался официальной стандартизации. Его код понятен, удобен и прост в изучении.

Для реализации веб-приложений на языке Python существует фреймворк Django. Это свободно распространяемое ПО. Фреймворк, как и его аналоги на других языках программирования, использует шаблон проектирования MVC. Однако, в Django есть существенные отличия от своих аналогов: сайты на нём строятся из одного или нескольких приложений, которые рекомендуется делать подключаемыми. Так же, обработчики URL в Django конфигурируются с помощью регулярных выражений, что более удобно.

В Django присутствует собственный ORM (объектно-реляционное отображение), что позволяет фреймворку генерировать собственную схему базы данных, не прибегая к сторонним разработкам, а так же защита от SQL-иньекций. Ко всему прочему, у фреймворка имеется уже готовый веб-интерфейс панели администратора, что очень облегчает и ускоряет работу.

**Вывод:** для реализации поставленной задачи необходим простой язык программирования, который может работать с фреймворками. Java для решения данной задачи слишком сложен в исполнении: для того чтобы отобразить даже одну веб-страницу необходимо совершить достаточно много действий.

SchoolCTF написан на Python с использованием Django. Ruby и PHP так же подходят для доработки этой жюрейной системы, однако, для того, чтобы использовать любой из этих языков программирования, придется переписывать проект с нуля. Поэтому было решено выбрать язык программирования Python: благодаря использованию именно его в проекте, нет необходимости переписывать проект заново под другое ПО, к тому же, проект реализован на Django, который, как было сказано выше, выделяется среди своих аналогов тем, что он более гибок, удобен, безопасен, и у него есть свой веб-интерфейс панели администратора.

Таким образом, был выбран второй вариант решения поставленной задачи: использовать готовую основу имеющегося проекта для последующих доработок.

* + 1. Выбор СУБД для разработки

Система управления базами данных (СУБД) играет одну из ключевых ролей в проекте. В рамках выполнения поставленной задачи главным ее критерием является удобство в обращении с ней и стабильность в работе.

Ниже перечислены три популярных базы данных, которые могут подойти для выполнения данной задачи:

* MySQL

MySQL является свободно распространяемой реляционной СУБД от компании Oracle. В основном используется для малых и средних приложений. Входит в состав некоторых серверов, а так же в портативную сборку Denwer. Достоинствами данной СУБД являются её гибкость, поддержка большого количества видов таблиц, а так же наличие примеров таблиц (EXAMPLE ) в составе СУБД, которые демонстрируют принципы создания новых таблиц.

* PostgreSQL

PostgreSQL – это свободно распространяемая объектно-реляционная СУБД. Используется для средних и больших приложений. Имеет ряд преимуществ перед своими конкурентами (например, перед MySQL), так как то, что СУБД объектно-реляционная дает возможность сохранять такие данные как многомерные массивы или сетевые адреса без использования обходного пути в виде дополнительно й таблицы. Так же, PostgreSQL поддерживает JSON в полной мере, в отличие от других СУБД, что обеспечивает дополнительную гибкость в работе.

* SQLite

SQLite – это компактная встраиваемая СУБД. Под понятием «встраиваемая» подразумевается то, что движок данной СУБД является не отдельно работающим процессом, а библиотекой, с которой взаимодействует программа. Поэтому для обмена информацией используются вызовы функций (API). Этот способ гораздо проще, удобнее и быстрее. Так же, преимуществом данной СУБД является динамическое типизирование данных.

Хоть SQLite и обладает более скудным функционалом, чем её полноценные аналоги, она вполне подходит для работы с малыми и средними приложениями. Она так же используется многими популярными программными продуктами для хранения данных (такими как ОС Android или браузер Google Chrome), привязана ко многим языкам программирования и входит в состав фреймворков Qt и Django.

**Вывод**: для реализации поставленной задачи по минимальным требованиям подходят все три приведенные СУБД. Однако, СУБД MySQL и PostgreSQL более сложные в использовании, и могут быть более нестабильны в работе, так как должны подключаться «со стороны». Так как в состав фреймворка Django уже входит SQLite, предпочтение в используемой СУБД было отдано этому варианту.

### 3.1.3 Выбор среды разработки

Для выполнения поставленной задачи необходима среда разработки, которая будет удовлетворять следующим требованиям:

* поддержка языка Python;
* удобная работа с фреймворком Django;
* прямое взаимодействие с распределенной системой контроля версий (Git);
* быстрый запуск проекта;
* запуск кода из среды разработки;
* поддержка отладки.

Всем этим требованиям удовлетворяет полнофункциональная интегрированная среда разработки (IDE) под названием PyCharm. Данная IDE так же выделяется хорошей поддержкой, регулярными обновлениями и возможностью работы прямо «из коробки».

В связи с вышеперечисленными достоинствами, PyCharm был выбран для реализации поставленного задания.

## 3.2 Объектное моделирование жюрейной системы SchoolCTF

### 3.2.1 Обоснование технологии проектирования

Для успешной реализации проекта объект проектирования (ИС) должен быть прежде всего адекватно описан, должны быть построены полные и непротиворечивые функциональные и информационные модели ИС. Для этих целей применяются CASE-технологии, которые представляет собой методологию проектирования ИС, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех этапах разработки и сопровождения ИС и разрабатывать приложения в соответствии с информационными потребностями пользователей.

Для разработки информационных систем также проводят моделирование бизнес процессов, целью которого является систематизация знаний о компании и ее бизнес-процессах в наглядной графической форме более удобной для аналитической обработки полученной информации. Моделирование бизнес-процессов позволяет проанализировать не только, как работает предприятие в целом, как оно взаимодействует с внешними система, но и как организована работа в каждом отдельно взятом модуле системы.

На этапе структурного моделирования в модели должны быть отражены:

* существующая организационная структура;
* документы и иные сущности, используемые при исполнении моделируемых бизнес-процессов и необходимые для моделирования документооборота, с описаниями их основного смысла;
* структуру бизнес-процессов, отражающую их иерархию от более общих групп к частным бизнес-процессам;
* диаграммы взаимодействия для конечных бизнес-процессов, отражающие последовательность создания и перемещения документов (данных, материалов, ресурсов и т.п.) между действующими лицами.

Для отображения основных частей информационной системы используют UML - язык для визуализации, специфицирования, конструирования и документирования артефактов программных систем. Язык состоит из словаря и правил, позволяющих комбинировать входящие в него слова и получать осмысленные конструкции. В языке моделирования словарь и правила ориентированы на концептуальное и физическое представление системы. Язык моделирования, подобный UML, является стандартным средством для составления описания бизнес процессов.

Моделирование необходимо для понимания системы. При этом единственной модели никогда не бывает достаточно. Напротив, для понимания любой нетривиальной системы приходится разрабатывать большое количество взаимосвязанных моделей. В применении к программным системам это означает, что необходим язык, с помощью которого можно с различных точек зрения описать представления архитектуры системы на протяжении цикла ее разработки.

В связи с основными требованиями, предъявляемые к модели информационной системы разработана следующая технология проектирования:

* разработать диаграмму вариантов использования, на которой отобразить отношения, существующие между актерами и прецедентами;
* разработать диаграмму последовательности, на которой показать взаимодействия объектов, упорядоченных во времени;
* разработать диаграмму классов, с помощью которой описать структуру системы, показав ее классы, их атрибуты и операторы, а также взаимосвязи этих классов.

Под выбранную технологию проектирования наиболее эффективным является выбор CASE-средства Draw.io

Draw.io – это веб-сервис, предназначенный для формирования диаграмм и схем. Draw.io работает в браузерах, поддерживающих HTML5. В нем доступны базовые и продвинутые возможности построения схем и диаграмм, а так же синхронизация с Google-диском и возможность конвертации построенной схемы в формат изображений (JPEG, PNG и др.).

### 3.2.1 Диаграмма прецедентов (use-case diagram)

Диаграмма прецедентов (или диаграмма вариантов использования) – диаграмма, построенная с помощью языка моделирования UML, которая отражает отношения между актерами и прецедентами и позволяющая описать систему на концептуальном уровне.

Прецедент – это часть функциональности системы, благодаря которой пользователь может получить желаемый им результат. Прецедент соответствует отдельно взятому сервису системы и определяет один из вариантов её использования.

Наглядно проиллюстрировать планируемый функционал жюрейной платформы можно с помощью use-case диаграммы (рисунок 6):



Рисунок 6. USE-CASE диаграмма жюрейной системы SchoolCTF

### 3.2.2 Диаграмма последовательности (sequence diagram)

Диаграмма последовательности отражает поток событий, происходящих в рамках варианта использования.

В диаграммах последовательности действий взаимодействие объектов в системе происходит посредством приема и передачи сообщений объектами-клиентами и обработки этих сообщений объектами-серверами. При этом в разных ситуациях одни и те же объекты могут выступать и в качестве клиентов, и в качестве серверов.

На диаграмме последовательности изображено упорядоченное во времени взаимодействие объектов при отправке участников ответа на задание.

Основным содержимым диаграммы являются сообщения, которые изображаются горизонтальными стрелками, переходящие от одной вертикальной линии жизни объекта к другой. Соответствующее сообщение ассоциируется с некоторым действием внутренних объектов: базой данных, сервером, пользователем, графическим интерфейсом. Каждое сообщение обладает именем, рядом передаваемых параметров, а также возвращаемым значением.

Диаграмма последовательности жюрейной системы SchoolCTF представлена на рисунке 7:



Рисунок 7. Диаграмма последовательности (sequence diagram)

Так как диаграмма последовательности описывает динамические аспекты жюрейной системы SchoolCTF, то можно выделить следующие действия, протекающие в течение линии жизни объектов:

* отправка ответа на сервер;
* проверка информации о правильности решения;
* выдача ответа;
* внос изменений в базу данных.

**Отправка ответа на сервер** происходит путем ввода участником правильного ответа в специализированное окно в веб-интерфейсе платформы.

**Проверка информации о правильности решения** происходит путем запроса сервера к базе данных. Сервер запрашивает у базы данных верный вариант ответа на задачу, база данных его выдает, сервер сверяет введенный участником ответ и ответ, который получен от базы данных.

**Выдача ответа** так же проводится сервером, плавно перетекая из проверки информации о правильности решения. Если решения участника было верным, сервер передает данную информацию в веб-интерфейс, который отображает её участнику. Если же ответ был неверный, то сервер так же передает информацию о неверности ответа в веб-интерфейс, который далее отображает её участнику.

**Внос изменений в базу данных** происходит в случае ввода участником правильного ответа на задание. В данной ситуации сервер перед отправкой сообщения участнику делает запрос к базе данных на внос изменений в список решенных участником заданий. После того, как он получает утвердительный ответ о том, что изменения внесены успешно, он оповещает участника о правильности его решения.

### 3.2.3 Диаграмма деятельности (activity diagram)

Диаграмма деятельности — UML-диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов — вложенных видов деятельности и отдельных действий, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Диаграммы деятельности состоят из ограниченного количества фигур, соединённых стрелками. Основные фигуры:

* Прямоугольники с закруглениями — действия
* Ромбы — решения
* Широкие полосы — начало (разветвление) и окончание (схождение) ветвления действий
* Чёрный круг — начало процесса (начальный узел)
* Чёрный круг с обводкой — окончание процесса (финальный узел)
* Стрелки идут от начала к концу процесса и показывают потоки управления или потоки объектов (данных).

На данной диаграмме (рисунок 8) представлены действия, выполняемые пользователем и системой при отправке участником ответа на задание в системе.



Рисунок 8. Диаграмма деятельности

Диаграмма содержит следующие наборы действий, образующие деятельность:

* отправка ответа на задачу в систему;
* отправка запроса на выдачу результата на сервер;
* отправка запроса в базу данных;
* выдача правильного ответа базой данных;
* сверка введенного и правильного ответов:
* ответ верный;
* ответ неверный;
* отправка запроса на изменение данных в базу данных;
* вывод сообщения участнику.

## 3.3 Обзор функциональности внесенных доработок в систему SchoolCTF

### 3.3.1 Отображение дополнительной информации о задачах

**Уровень сложности задачи.**

Уровень сложности задачи расположился в третьем столбце страницы с списком задач (рисунок 9):



Рисунок 9. Отображение уровня сложности задач.

Где «very easy» – это самый легкий уровень, а «very hard» –самый сложный.

Уровень сложности так же отображается в базе данных (рисунок 10):



Рисунок 10. Список заданий, доступных для изменения.

Добавление уровня сложности происходит путем редактирования или добавления конкретного задания (рисунок 11):



Рисунок 11. Добавление уровня сложности.

**Отображение списка команд, уже решивших задачу.**

Данная доработка нашла своё место на странице каждой задачи под кнопкой отправки флага (рисунок 12):



Рисунок 12. Отображение списка команд, уже решивших задачу.

В данной таблице отображается команда и время решения каждой командой задачи.

### 3.3.2 Составление краткой инструкции по разворачиванию системы

Вышеупомянутая функция была составлена и добавлена в файл readme.txt, который находится в открытом репозитории по адресу https://github.com/PolMarzia/schoolctf.

Внутренняя составляющая данной инструкции отображена на рисунке 13:

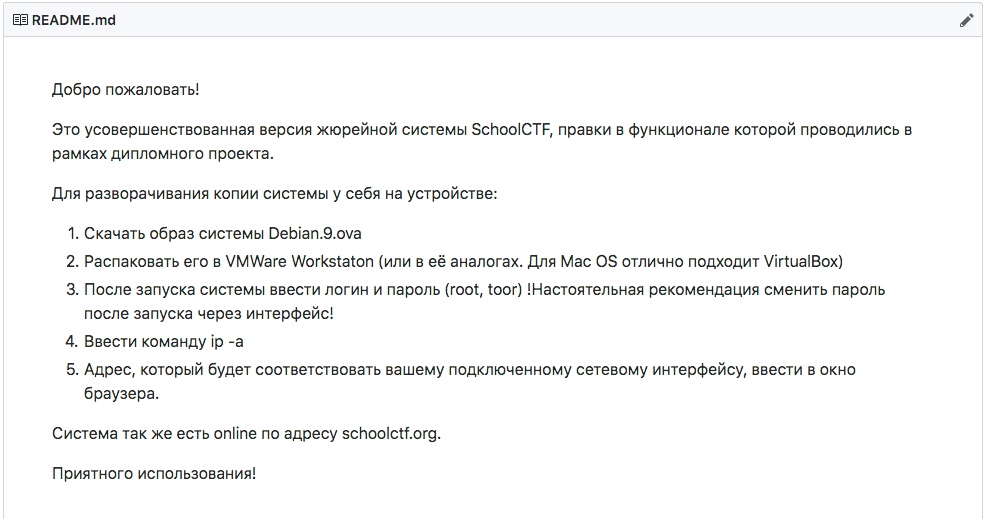


Рисунок 13. Внутренняя составляющая инструкции по разворачиванию системы.

### 3.3.3 Возможность редактирования информации на главной странице через панель администратора.

Возможность редактирования главной страницы была реализована по аналогии с редактированием блока новостей.

Путем создания новой модели, а соответственно, и таблицы с данными, было осуществлено написание заголовков на главной странице.

Так выглядит список статей для изменения (рисунок 14):



Рисунок 14. Список статей для изменения.

Редактирование статьи осуществляется на новой странице (рисунок 15):



Рисунок 15. Редактирование статьи с главной страницы.

Результат сохранения новой страницы виден на рисунке 15:



Рисунок 15. Результат редактирования главной страницы.

3.3.4 Вывод аналитики в виде графиков.

Графическая аналитика была реализована с помощью скрипта, написанного на языке JavaScript. Её код расположен в файле loader.js.

Аналитика отображает список команд, которые решили задачу, а также информацию о том, какая команда какое задание решила.

График находится на странице Scoreboard, сразу под таблицей с рейтингом участников.

Так выглядит график (рисунок 16):

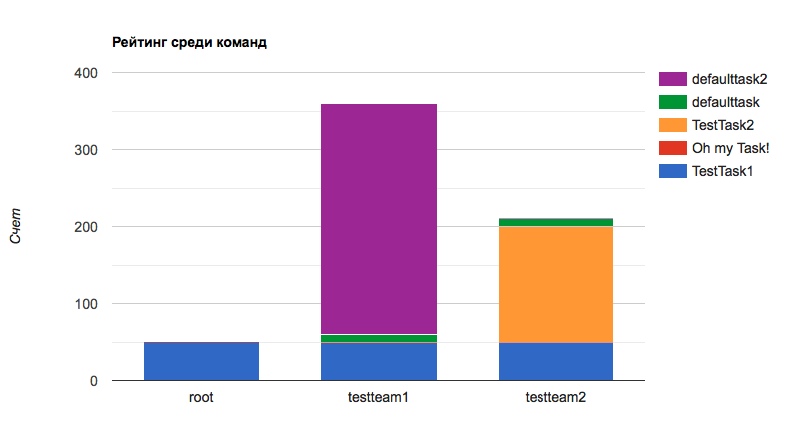


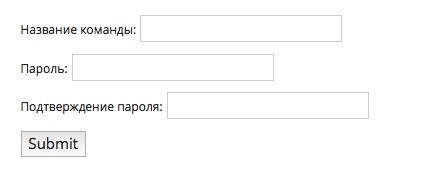
Рисунок 16. График рейтинга команд.

3.3.5 Возможность самостоятельной регистрации команд участниками.

Возможность самостоятельной регистрации команд была реализована с помощью функции def register\_user, которая написана в файле views.py и html-страницы reg\_scorry.html.

При регистрации участник должен ввести логин и пароль. Затем, если введенные пароли совпали, регистрация считается успешной.

Форма регистрации выглядит следующим образом (рисунок 17):



# Рисунок 17. Форма регистрации.

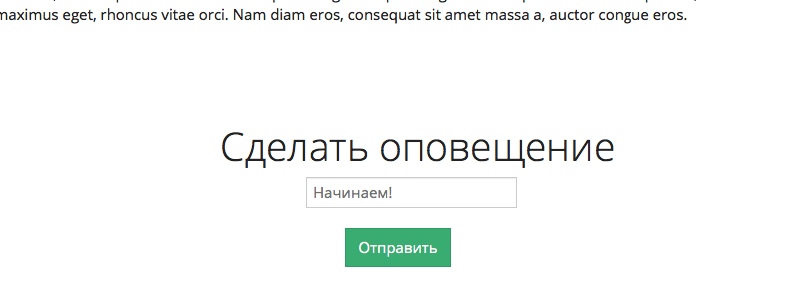
3.3.6 Наличие оповещений.

Оповещения реализованы с помощью скрипта, написанного на JavaScript. Скрипт находится в файле notify.js.

Данная функция задействует систему оповещений браузера, поэтому для её корректного функционирования необходимо разрешить всплывающие уведомления в браузере на данном сайте.

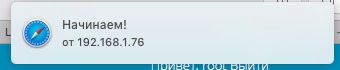
Оповещение можно ввести на главной странице, сразу под текстом на ней. Этот функционал доступен только администратору. Выведенное оповещение сохраняется в базе данных в таблице Notifications.

Окно для ввода оповещений выглядит следующим образом (рисунок 18):



# Рисунок 18. Окно для ввода оповещений.

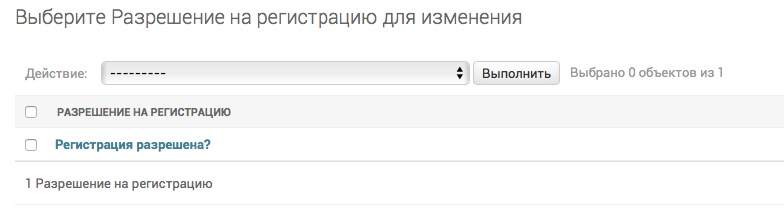
Выведенное оповещение выглядит следующим образом (рисунок 19):



# Рисунок 19. Выведенное оповещение.

3.3.7. Включение и отключение возможности регистрации через панель администратора.

Данная функция реализована через ввод модели AllowedReg. Данная модель отображается в базе данных в виде таблицы «Разрешение на регистрацию». В данной таблице одна запись, которая не подлежит удалению (рисунок 20):



# Рисунок 20. Запись в таблице "Разрешение на регистрацию".

Функция регистрации включается и отключается с помощью редактирования записи «Регистрация разрешена?». Флаг, установленный на поле “Is Allowed” означает, что регистрация доступна, и кнопка «Регистрация» отображена. Если флаг снят, то кнопка «Регистрация» отсутствует.

Так выгляд

# **ЭКОНОМИЧЕСКИЙ РАЗДЕЛ**

## 4.1 Планирование разработки программного продукта с построением графика

В выпускной квалификационной работе проводится «Разработка автоматизированной системы управления запасами готовой продукции». В данном разделе определяется трудоемкость и затраты на создание ПО, а также производится расчет экономического эффекта, который может быть получен от применения разрабатываемого ПО.

### 4.1.1 Определение трудоемкости и продолжительности работ по созданию автоматизированной системы управления запасами готовой продукции

Процесс разработки включает: анализ предметной области, обзор и анализ программных средств схожей тематики, анализ и выбор программных продуктов для создания программы; отладка; испытание. В свою очередь каждый из этих этапов можно подразделить на отдельные подэтапы.

Регламентируются следующие стадии проведения исследования:

* техническое задание
* эскизный проект
* технический проект
* рабочий проект
* внедрение

На всех стадиях проведения исследования выполняются следующие виды работ, перечень которых показан в таблице 2.

| **Стадии разработки** | Перечень работ |
| --- | --- |
| Техническое задание | * + постановка задачи;   + подбор литературы;   + сбор исходных данных; |
|  | * + определение требований к системе;   + определение стадий, этапов и сроков разработки ПО; |
| Эскизный проект | * + анализ программных средств схожей тематики;   + разработка общей структуры ПО;   + разработка структуры программы по подсистемам;   + документирование; |
| Технический проект | * определение требований к ПО; * выбор инструментальных средств; * определение свойств и требований к аппаратному обеспечению; |
| Рабочий проект | * + - верстка и дизайн;     - программирование;     - тестирование и отладка ПО;     - разработка программной документации;     - согласование и утверждение работоспособности системы; |
| Внедрение | * опытная эксплуатация; * анализ данных, полученных в результате эксплуатации; * корректировка технической документации по результатам испытаний |

Таблица 2. Перечень работ на каждой стадии проведения исследования

Трудоемкость выполнения работ по созданию ПО определяется по сумме трудоемкости этапов и видов работ, оцениваемых экспертным путем в человеко-днях, и носит вероятностный характер, так как зависит от множества трудно учитываемых факторов.

Трудоемкость каждого вида работ определяется по формуле (1).

, (1)

где:

*t*min – минимально возможная трудоемкость выполнения отдельного вида работ;

*t*max – максимально возможная трудоемкость выполнения отдельного вида работ.

Продолжительность каждого вида работ в календарных днях (*Ti*) определяется по формуле (2), в днях:

, (2)

где:

*ti* – трудоемкость работ, человек-дней;

*Чi* – численность исполнителей, человек;

*Kвых* – коэффициент, учитывающий выходные и праздничные дни:



где:

*Ккал. –* число календарных дней;

*Краб.*– рабочие дни;

*Kвых*=1,3.

Полный список видов и этапов работ по созданию ПО, экспертные оценки и расчетные величины их трудоемкости, а также продолжительность каждого вида работ, рассчитанные по формулам (1) и (2), представлены в таблице 3.

| **№ работы** | **Стадии разработки** | **Трудоемкость, чел.дни** | | | **Количество работников, чел.** | | **Продолжительность работ, календарные дни** | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***tmin*** | ***tmax*** | ***ti*** | ***Чi*** | | ***Ti*** | |
| Техническое задание | | | | | | | |
| 1 | - постановка задачи | 2 | 4 | 2,8 | 1 | | 3,6 | |
| 2 | - подбор литературы | 3 | 3 | 3 | 1 | | 3,9 | |
| 3 | - сбор исходных данных | 3 | 4 | 3,4 | 1 | | 4,4 | |
| 4 | - определение требований к системе | 2 | 2 | 2 | 1 | | 2,6 | |
| 5 | - определение стадий, этапов и сроков разработки ПО | 2 | 3 | 2,4 | 1 | | 3,1 | |
| Эскизный проект | | | | | | | |
| 6 | - анализ программных средств схожей тематики | 7 | 8 | 7,4 | 1 | | 9,6 | |
| 7 | - разработка общей структуры ПО | 3 | 7 | 4,6 | 1 | | 5,9 | |
| 8 | - разработка структуры программы по подсистемам | 4 | 7 | 5,2 | 1 | | 6,7 | |
| 9 | - документирование | 2 | 3 | 2,4 | 1 | | 3,1 | |
| Технический проект | | | | | | | |
| 10 | - определение требований к ПО | 2 | 2 | 2 | 1 | | 2,6 | |
| 11 | - выбор инструментальных средств | 2 | 3 | 2,4 | 1 | | 3,1 | |
| 12 | - определение свойств и требований к аппаратному обеспечению | 1 | 2 | 1,4 | | 1 | 1,8 | |
| Рабочий проект | | | | | | | | |
| 13 | - верстка и дизайн | 6 | 7 | 6,4 | | 1 | 8,3 | |
| 14 | - программирование | 10 | 19 | 13,6 | | 1 | 17,6 | |
| 15 | - тестирование и отладка ПО | 7 | 7 | 7 | | 1 | 9,1 | |
| 16 | - разработка программной документации | 3 | 4 | 3,4 | | 1 | 4,4 | |
| 17 | - согласование и утверждение работоспособности системы | 2 | 3 | 2,4 | | 1 | 3,1 | |
| Внедрение | | | | | | | | |
| 18 | - опытная эксплуатация | 8 | 9 | 8,4 | | 1 | 10,9 | |
| 19 | - анализ данных, полученных в результате эксплуатации | 2 | 4 | 2,8 | | 1 | 3,6 | |
| 20 | - корректировка технической документации по результатам испытаний | 1 | 2 | 1,4 | | 1 | 1,8 | |
|  | **Общая трудоемкость разработки** |  |  | 84 | |  | 109 | |

Таблица 3. Расчет трудоемкости и продолжительности работ по созданию ПО

Таким образом, общая продолжительность проведения работ составит 110 календарных дней.

4.1.2 Построение ленточного графика проведения исследования

В качестве инструмента планирования работ используем ленточный график. Ленточный график позволяет наглядно представить логическую последовательность и взаимосвязь отдельных работ, срок начала и срок окончания работ. Он представляет собой таблицу, где перечислены наименования стадий разработки и видов работ, длительность выполнения каждого вида работ. Продолжением таблицы является график, отражающий продолжительность каждого вида работ в виде отрезков времени, которые располагаются в соответствии с последовательностью выполнения работ.

Ленточный график разработки ПО, построенный по данным таблицы 2, приведен на рисунке 16.



Рисунок 16 – Ленточный график разработки ПО

## 4.2 Расчет сметы затрат на разработку программного продукта

Сметная стоимость проектирования и внедрения программы включает в себя следующие затраты, определяемые по формуле (3):

*Спр*=*Сосн* + *Сдоп* + *Ссоц* + *См* + *Смаш.вр* + *Сн*, (3)

где:

*Спр* – стоимость разработки ПО;

*Сосн* – основная заработная плата исполнителей;

*Сдоп* – дополнительная заработная плата исполнителей, учитывающая потери времени на отпуска и болезни (принимается в среднем 10% от основной заработной платы);

*Ссоц* – отчисления во внебюджетные фонды государственного социального страхования (пенсионный фонд, фонд обязательного медицинского страхования, фонд социального страхования), рассчитываются как 0,26% от основной и дополнительной заработной платы;

*См* – затраты на используемые материалы;

*Смаш.вр* – стоимость машинного времени.

*Сн* – накладные расходы включают затраты на управление, уборку, ремонт, электроэнергию, отопление и др. (принимаются в размере 60% от основной и дополнительной заработной платы);

**Основная заработная плата исполнителей.**

На статью «Заработная плата» относят заработную плату научных, инженерно-технических и других работников, непосредственно участвующих в разработке ПО. Расчет ведется по формуле (4):

*Зисп* = *Зср* ⋅ *Т*, (4)

где:

*Зисп*– заработная плата исполнителей (руб.);

*Зср* – средняя тарифная ставка работника организации разработчика ПО (руб./чел./дни);

*Т* – трудоемкость разработки ПО (чел.дни).

*Зср*определяется по формуле (5):

*Зср* = *С* / *Фмес*, (5)

где:

*С* – зарплата труда на текущий момент времени (руб./мес.);

*Фмес*– месячный фонд рабочего времени исполнителя (дни).

Затраты на статью «Заработной платы» приведены в таблице 4.

| **№** | **Исполнитель** | **Оклад, руб./мес.** | **Оклад, руб./день** | **Трудоемкость, чел.дни** | | **Сумма, руб.** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Инженер-программист | 40000 | 2000 | | 84 | 168000 |
| Общая основная заработная плата исполнителей, *Сосн* | | | | | 84 | 168000 |

Таблица 4. Затраты на заработную плату

Дополнительная заработная плата

Дополнительная заработная плата на период разработки ПО рассчитывается относительно основной и составляет 10% от ее величины:

*Сдоп* = *Сосн* **·** 0,1 = 168000 **·** 0,1 = 16800 (руб.)

Расчет отчислений на социальное страхование

Социальное страхование включает отчисления во все внебюджетные фонды, в том числе пенсионный, занятости, обязательного медицинского страхования, социального страхования. Отчисления на социальное страхование рассчитываются относительно выплаченной заработной платы (суммы основной и дополнительной заработной платы). Составляют 26%:

*Ссоц* = (*Сдоп* + *Сосн*) **·** 0,26 (6)

*Ссоц* = (168000 + 16800) **·** 0,26 = 48048 (руб.)

**Расчет расходов на материалы**

На эту статью относят все затраты на магнитные носители данных, бумагу, для печатных устройств, канцтовары и др. Затраты по ним определяются по экспертным оценкам. Расчет расходов на материалы приведен в таблице 5.

| **№** | **Материалы** | **Количество, штуки** | **Стоимость, рубли** |
| --- | --- | --- | --- |
| 1 | Бумага писчая, листов | 1000 | 600 |
| 2 | Картридж для принтера, шт | 1 | 1200 |
| 3 | Другие канцтовары | ­­­- | 1000 |
| Общая стоимость материалов, *См* | | | 2800 |

Таблица 5. Расчет расходов на материалы

**Накладные расходы**

На статью «Накладные расходы» относят расходы, связанные с управлением и организацией работ. Накладные расходы рассчитываются относительно основной заработной платы. Величина накладных расходов принимается равной 60% от основной зарплаты исполнителей. Формула расчета (7):

*Сн* = *Сосн* **·** *К*,(7)

где:

*Сн* – накладные расходы (руб.);

*Сосн* – основная заработная плата исполнителей (руб.);

*К* – коэффициент учета накладных расходов (*К* = 0,6)

*Сн* = 168000**·** 0,6 = 100800 (руб.)

**Расчет стоимости машинного времени**

Затраты на машинное время, необходимое для разработки ПО, расходы на приобретение и подготовку материалов научно-технической информации, расходы на использование средствами связи. Расчет затрат на машинное время осуществляется по формуле (8):

*Смаш.вр = Кмаш.вр · Змаш.вр* (8)

где:

*К****маш.вр*** – тарифная стоимость одного часа машинного времени (*К****маш.вр***=60 руб./ч.)

*Змаш****.вр*** – машинное время, используемое не проведение работ.

Необходимое количество машинного времени для реализации проекта по разработке программы рассчитывается по формуле:

*Змаш.вр* = *ti* **·** *Tсм* **·** *Tср.маш*, (9)

где:

*ti* – трудоемкость работ, чел.дней;

*Tсм* – продолжительность рабочей смены (При пятидневной рабочей неделе *Tсм* = 8 ч.);

*Tср.маш* – средний коэффициент использования машинного времени (*Tср.маш* = 0,7).

Тогда:

*Змаш.вр* = 84**·** 8**·** 0,7 = 470,4 (ч..)

Стоимость машинного времени составит:

*Смаш.вр* = 60**·** 470,4 = 28224 (руб.)

Результаты расчета затрат на проектирование программного обеспечения сведены в таблице 6.

| **№** | **Наименование статей** | **Обозначение** | **Сумма, руб.** | **В % к итогу** |
| --- | --- | --- | --- | --- |
| 1 | Основная заработная плата | *Сосн* | 168000 | 42,05 |
| 2 | Дополнительная заработная плата | *Сдоп* | 16800 | 4,20 |
| 3 | Отчисления на социальные нужды | *Ссоц* | 48048 | 12,02 |
| 4 | Материалы | *Смат* | 2800 | 1,14 |
| 5 | Стоимость машинного времени | *Смаш.вр* | 28224 | 15,36 |
| 6 | Накладные расходы | *Сн* | 100800 | 25,23 |
| Итого: | | *Спр* | 364672 | 100 |

Таблица 6. Смета затрат на разработку и внедрение программы

Таким образом, себестоимость разработки составляет **364672** руб.

Данная программа может быть реализована на рынке. При расчетном количестве реализованных программ (*n*=5), оптовая цена программы (*Цопт*) может быть рассчитана по формуле:

*Цопт* = ;

где:

*Спр* – себестоимость разработки программы;

*П* – прибыль, определяется по формуле:

;

где:

*Ур* – средний уровень рентабельности (*Ур* = 20%).

Таким образом, оптовая цена программы составит:

*Цопт* **=**364672/5 + (364672/5)**·** 0.2 = 72934,4 + 14586,88 **= 87521,28** (руб.)

Отпускная цена реализации программы потребителям (*Цотп*), рассчитывается по формуле:



где:

*НДС* – налог на добавленную стоимость, рассчитывается в соответствии с действующей ставкой этого налога – 20% от оптовой цены программы.

*Цопт =*87521,28 + 87521,28 \*0.20 **=** 87521,28 + 17 164,25= **104 685,53** (руб.)

Таким образом, отпускная цена программы составит **104 685,53** руб., в том числе НДС – **17 164,25** руб.

## 4.3 Расчет основных технико-экономических показателей и эффективности использования программного продукта

Экономический эффект представляет собой абсолютную величину, характеризующую достигнутые благодаря созданию или совершенствованию ПС дополнительные (в сравнении с базовым вариантом) экономические результаты.

Экономический эффективность – это относительная величина, отражающая соотношение полученного эффекта и затрат на его достижение.

### 4.3.1 Характеристика решения задачи

**Базовый вариант**

Количество сотрудников, выполняющих данный объем работ без применения ПО – 3 человека.

**Проектный вариант**

Количество сотрудников, которые будут выполнять данный объем работ с использованием разработанного ПО – 1 человек и 1 ПЭВМ.

4.3.2 Определение трудоемкости обработки информации по базовому и проектному вариантам

Расчет трудоемкости по обработке данных на ЭВМ, в человеко-часах, для обоих вариантов осуществляется по формуле (10):

*Qчi* = *Fдр***·** *Ni*, (10)

где:

*Fдр* – действительный годовой фонд времени работы одного работника, чел. ч.;

*Ni* – количество работников по заданному варианту, чел.

Трудоемкость обработки информации при среднем годовом действительном фонде времени работы одного работника за год, равного

*Fдр* = 1798 (чел./ч.), составит:

**Базовый вариант**

*Qчi =* 1798**·** 3 = 5394 (чел.ч.)

**Проектный вариант**

*Qчi =* 1798**·** 1 = 1798 (чел.ч.)

При среднем коэффициенте использования персонального компьютера для решения задачи по проектному варианту, равном 0,7, трудоемкость обработки информации на персональном компьютере за год вычисляется по формуле (11):

*Qmi* = *Qчi* **·** *Ki*  (11)

**Проектный вариант**

*Qmi* = 1798 **·** 0,7 = 1259 (маш.ч.)

4.3.3 Расчет капитальных вложений

**Базовый вариант**

Капитальные вложения складываются из стоимости производственных помещений, необходимых для размещения работников без ПК, исходя из расчета по норме 3 кв.м. на человека и стоимости 1 кв.м. площади – 7000 руб.

*Кбаз* = 3**·** 3**·** 7000=63000 (руб.)

**Проектный вариант**

Капитальные вложения складываются из:

* + - Стоимости производственных помещений из расчета 6 кв.м на 1 рабочее место работника с ПК и стоимостью 1 кв.м площади – 8000 руб.:

*Кпомещ.* = 1**·** 6**·** 8000 = 48000 руб.

* + - Стоимости оборудования:
      * + системный блок: Intel Core Duo CPU E4700 2600 МГц, RAM 2048 Mb, 3.5 FDD, 320 Gb HDD, DVD/RW стоимость 8000 руб.;
        + монитор: Samsung SyncMaster 943SN 18.5", широкоформатный, 1360x768, 250 кд/м2, 1000:1, 5 мс, 170°/160°, VGA, стоимость 6000 руб.;
        + принтер: HP LaserJet P3005 A4 и A3, печать лазерная черно-белая, 14 стр/мин ч/б, 600x600 dpi, подача: 150 лист., вывод: 100 лист., стоимость 4100 руб.

*Ктех.*= 8000+6000+4100 = 18100 руб.

* + - Стоимости разработки программы:

*Кр.прог.* = 52052,73  руб.

Общие капитальные вложения рассчитываются по формуле (12):

*Кпроект*.= *Кпомещ* + *Ктехн*. + *Кр.прог.*  (12)

где:

*Кпомещ* – стоимость производственных помещений;

*Ктехн*. – стоимость приобретаемого оборудования.

*Кпроект*.= 48000+18100+52052,73= 118152,73 (руб.)

### 4.3.4 Расчет годовых текущих затрат

Производится по формуле:

*Сi* = *См* + *Сосн* + *Сдоп* + *Ссоц*+ *Сн* + *Саморт* + *Ссодерж*., (13)

где:

*С*м – затраты на используемые материалы (руб.);

*Сосн* – основная заработная плата исполнителей за год (руб.);

*Сдоп* – дополнительная заработная плата работников, учитывающая потери времени на отпуска и болезни (принимается в среднем 10% от основной зарплаты работников) (руб.)

*Ссоц* – отчисления во внебюджетные фонды государственного социального страхования (пенсионный фонд, фонд обязательного медицинского страхования, фонд социального страхования), рассчитываются как 30 % от основной и дополнительной заработной платы (руб.);

*Сн* – накладные расходы включают затраты на управление, уборку, ремонт, электроэнергию, отопление и др. (принимаются в размере 60% от основной и дополнительной заработной платы) (руб.);

*Саморт.* – годовая сумма амортизационных отчислений по соответствующим группам основных производственных фондов и нематериальных активов (рассчитывается исходя из следующих норм амортизации: 12,5% - от стоимости оборудования; 2,5% - от стоимости производственных площадей и 30% - от стоимости программного обеспечения, исходя из расчетного срока службы ПО – три года) (руб.).;

*Ссодерж.* – затраты на содержание продукта в год (руб.).

**Базовый вариант**

Текущие затраты состоят из:

* Заработная плата исполнителей (по 30000 руб. в месяц)

30000**·** 12**·** 3=1 080 000 (руб.);

* Дополнительная заработная плата (10% от основной заработной платы) 1 080 000**·** 0,1= 108 000 (руб.);
* Отчисления на социальное страхование (26% от основной и дополнительной заработной платы)

(1 080 000 + 108 000)**·** 0,3= 356 400 (руб.);

* Накладные расходы (60% от основной и дополнительной заработной платы)

(1 080 000 + 108 000)**·** 0,6= 712 800‬ (руб.)

* Амортизационные отчисления (2,5% от стоимости помещений)

63000**·** 0,025=1575 (руб.)

* Затраты на содержание продукта в год (руб.) (*Ссодерж.* = 3000 руб.)

*Сбаз*= 1 080 000+108 000+356 400 +712 800+1575+3000=2 261 775(руб.)

**Проектный вариант**

* Заработная плата исполнителей (по 40000 руб. в месяц)

40000**·** 12**·** 1=480 000‬ (руб.);

* Дополнительная заработная плата (10% от основной заработной платы)

480 000‬**·** 0,1=48 000 (руб.);

* Отчисления на социальное страхование (26% от основной и дополнительной заработной платы)

(480 000‬+48 000‬)**·** 0,26= 137 280‬ (руб.);

* Накладные расходы (60% от основной и дополнительной заработной платы)

(480 000‬+48 000‬) **·** 0,6=316 800‬ (руб.);

* Амортизационные отчисления по производственным площадям (2,5% от стоимости помещений)

48000**·** 0,025=1200 (руб.)

* Амортизационные отчисления по оборудованию, исходя из годовой нормы – 12,5% (от стоимости оборудования)

(8000+6000+4100)**·** 0,125=2262,5 (руб.)

* Затраты на содержание продукта в год (*Ссодерж.* = 3000 руб.)

*Спроект* = 480 000+48 000 +137 280+316 800+1200+2262,5+3000 = 988 542,5‬ (руб.)

4.3.5 Расчет показателей экономической эффективности

Экономический эффект достигается за счет уменьшения годовых затрат и повышения качества работы после внедрения предлагаемого АРМ.

Годовая экономия текущих затрат рассчитывается по формуле (14):

*Сm* = *Cбаз*– *Спр*, (14)

где:

*Cбаз* – годовые текущие затраты на обработку данных в базовом варианте;

*Спр* – годовые текущие затраты на обработку данных в проектном варианте;

*Сm* = 1106415 – 469950,5 = 636464,5 (руб.)

Годовые приведенные затраты для базового и для проектного варианта рассчитываются по формуле (15):

*Зi* = *Сi* + *Eн* **·** *Кi*, (15)

где:

*Сi* – годовые текущие затраты для базового и проектного варианта соответственно;

*Eн* – нормативный коэффициент сравнительной экономической эффективности капиталовложений (*Eн* = 0,15);

*Кi* – капиталовложения для базового и проектного варианта соответственно.

**Базовый вариант**

*Збаз* = 2 261 775+ 0,15**·** 63000 = **2 271 225‬** (руб.)

**Проектный вариант**

*Зпроект* = 988 542,5+ 0,15**·** 118152,73 = **1 006 265,41** (руб.)

Годовой экономический эффект рассчитывается по формуле (16):

*Э* = *Збаз* – *Зпроект*, (16)

где:

*Збаз* – годовые приведенные затраты по базовому варианту;

*Зпроект* – годовые приведенные затраты по проектному варианту.

*Э* = 2 271 225 – 1 006 265,41 = **1 264 959,59‬** (руб.)

Основные технико-экономические показатели проекта приведены в таблице 7.

| **Наименование показателя** | **Ед. измерения** | **Базовый вариант** | **Проектный вариант** | **Проектный вариант в % к базовому** |
| --- | --- | --- | --- | --- |
| Способ обработки информации | \_\_\_\_ | Вручную | С применением ЭВМ и программных средств | \_\_\_\_ |
| Используемое оборудование | \_\_\_\_ | Конфигурация базового ПК | Intel Core Duo CPU E4700 2600 МГц, RAM 2048 Mb, 3.5, монитор Samsung, принтер HP LaserJet P3005 | \_\_\_\_ |
| Годовые затраты на обработку информации | Чел. ч.  Маш. ч. | 5394 | 1798  1259 | 33 |
| Количество работников | Чел | 3 | 1 | 33 |
| Потребность в производственных площадях | м2 | 9 | 6 | 67 |
| Капитальные вложения | Руб. | 63000 | 118152,73 | 188 |
| Годовые текущие затраты на обработку информации | Руб. | 1106415 | 469950,5 | 42 |
| Годовые приведенные затраты на обработку информации | Руб. | 1115865 | 487673,41 | 44 |
| Годовой экономический эффект | Руб. | \_\_\_\_ | 1 264 959,59‬ | \_\_\_\_ |

Таблица 7. Основные технико-экономические показатели проекта

Таким образом, согласно проведенным расчетам, внедрение проектных организационно-технических мероприятий и ПО позволяет получить годовой экономический эффект в сумме **1 264 959,59** рубля и снизить:

* Годовые приведенные затраты труда на 44%;
* Количество работников на 33%
* Годовые текущие затраты на 42%.

# **ЗАКЛЮЧЕНИЕ**

В результате выпускной квалификационной работы была осуществлена доработка жюрейной системы «SchoolCTF» путем внедрения в неё нового современного функционала.

В исследовательском разделе описана методика жюрейства CTF-соревнований, проведен обзор существующих систем, выполнен сравнительный анализ выбранного ПО, выбрана система для доработки, формулирован список доработок для платформы, определена цель выпускной квалификационной работы.

В специальном разделе описаны алгоритмы работы предыдущей версии выбранной жюрейной системы и алгоритмы работы новой её версии, обозначена структура обеих версий приложения, подробно описана структура базы данных приложения и вносимых в неё изменений.

В технологическом разделе описано обоснование выбора программных средств для реализации поставленной цели, с помощью Case-технологий описан функционал жюрейной системы, а также, продемонстрирован обновленный функционал системы.

В экономическом разделе проведён анализ стадий проведения исследований, рассчитано время, требующееся на разработку и тестирование информационно-измерительной системы для жюрейной системы «SchoolCTF» определены затраты и капитальные вложения на разработку системы, приведены основные технико-экономические показатели проекта.

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. «Национальная психологическая энциклопедия, экспериментальное обучение», электронный ресурс: https://vocabulary.ru/termin/eksperimentalnoe-obuchenie.html, дата обращения 15.03.2019;

2. «Всё о CTF в России», электронный ресурс: https://ctfnews.ru/what-is-ctf/, дата обращения 16.03.2019;

3. «GitHub, CTFd», электронный ресурс: https://github.com/CTFd/CTFd, дата обращения 20.03.2019;

4. «GitHub, Basic-Deployment», электронный ресурс: https://github.com/CTFd/CTFd/wiki/Basic-Deployment, дата обращения 20.03.2019;

5. «CTFd», электронный ресурс: https://ctfd.io, дата обращения 20.03.2019;

6. «GitHub, Project Asya», электронный ресурс: https://github.com/KevaTeam/ProjectAsya, дата обращения: 25.03.2019;

7. «Вконтакте, сообщество KevaTeam», электронный ресурс: https://vk.com/kevactf, дата обращения 26.03.2019;

8. «Вконтакте, сообщество жюрейной системы Hackerdom», электронный ресурс: https://vk.com/hackerdom, дата обращения 26.03.2019;

9. «Habr, SibirCTF 2015: как это было», электронный ресурс: https://habr.com/ru/post/263473/, дата обращения 27.03.2019;

10. «GitHub, qCTF-School-2018», электронный ресурс: https://github.com/HackerDom/qctf-school-2018, дата обращения 28.03.2019;

11. «GitHub, drapo», электронный ресурс: https://github.com/andgein/drapo, дата обращения 28.03.2019;

12. «Википедия: свободная энциклопедия, Facebook», электронный ресурс: https://ru.wikipedia.org/wiki/Facebook, дата обращения 29.03.2019;

13. «Xakep.ru, Facebook открыла исходные коды платформы Capture The Flag», электронный ресурс: https://xakep.ru/2016/05/12/facebook-ctf/, дата обращения: 01.04.2019;

14. «Facebook, Facebook CTF is Now Open Source», электронный ресурс: https://www.facebook.com/notes/facebook-ctf/facebook-ctf-is-now-open-source/525464774322241, дата обращения: 02.04.2019;

15. « GitHub, Volalex, SchoolCTF», электронный ресурс: https://github.com/volalex/schoolctf, дата обращения: 03.04.2019;

16 « GitHub, PolMarzia, SchoolCTF», электронный ресурс: https://github.com/PolMarzia/schoolctf, дата обращения: 05.04.2019;

17. «Django Packages», электронный ресурс: https://djangopackages.org/, дата обращения 10.04.2019;

18. «Документация Django 1.9», электронный ресурс: https://djbook.ru/rel1.9/, дата обращения 15.04.2019;

19. «Как и для чего использовать Docker», электронный ресурс: https://guides.hexlet.io/docker/, дата обращения: 20.04.2019;

20. «Docker, документация на русском», электронный ресурс: https://docker.crank.ru/docs/, дата обращения: 02.05.2019;

21. Сергей Колесников. «Беспроводные сети. Стандарты», электронный ресурс: http://old.ci.ru/inform19\_05/p\_24.htm, дата обращения 10.05.2019;

22. Чаплыгин В.А. Организационно-экономический раздел дипломных работ научно-исследовательского характера. М., МГУПИ. 2007; 57 Методические указания по сбору материалов на преддипломной практике и выполнению организационно-экономического раздела дипломных проекторов. М., МГУПИ, 2015;

23. Соснина С.А., Титков А.М., Плиева З.Р. «Экономика, методические указания. М., МГУПИ, 2012.