

Тема работы

**Специальность, Geekbrains**

**Щербак Сергей Андреевич**

**2023**

**Основные требования к оформлению дипломного проекта**

1. Текст проекта оформляйте только с одной стороны листа формата А4. При наборе используйте межстрочный интервал 1,5.  
   Объемные таблицы/иллюстрации возможно размещать на листах А3, А2 и выносить в приложения.
2. При наборе информации, используйте шрифт Times New Roman с выравниванием текста по ширине. Рекомендуем кегль — 12–14, для некоторых структурных единиц работы можно выбрать полужирное выделение.
3. Поля страницы, где размещается текст, выставляйте по определенным показателям: нижнее/верхнее – 20 мм, правое – 10 мм, левое (для последующей прошивки) – 30 мм.
4. Все листы дипломного проекта нумеруются по порядку, начиная с введения (используется сквозная нумерация). Номер указывайте в центре нижней части страницы без точки.
5. Заголовок раздела с одним подразделом или пунктом не нумеруется.
6. После текста перед следующим заголовком установите одиночный интервал.
7. После заголовка перед подзаголовком проставьте двойной интервал.
8. Имена собственные — наименования компаний/изделий, фамилии учёных и т. д., указывайте на языке оригинала.
9. Каждый структурный элемент начинается с нового листа.

*Вся работа должна составлять 50+ страниц формата А4, включая приложения.*

|  |
| --- |
| 💡 Инструкция по работе с шаблоном.   1. Создайте копию данного шаблона. Файл - Создать копию. 2. Изучите шаблон и начинайте работу. |

**Содержание**

[Содержание](#_heading=h.7mt9x4b37ce)

[Введение](#_heading=h.1fob9te)

[Теоретическая и практическая главы](#_heading=h.hst8x8f1r8tz)

[Заключение](#_heading=h.dhwmmf3wq1kk)

[Список используемой литературы](#_heading=h.m8jvbwhv7ujg)

[Приложения](#_heading=h.s6k7t2lwcp0k)

# Как оформить содержание?

В содержании дипломного проекта необходимо озаглавить основные блоки и подпункты работы с указанием страниц.

### Пример

*Данный пример не является эталонным, лишь один из вариаций.*

Введение (2–3 стр.)

Глава 1. Основы ручного тестирования веб-приложений (~15 стр.)

1.1 Что такое веб-приложение, его особенности

1.2 Что такое тестирование, кто такой тестировщик. Зачем нужно тестирование и когда оно начинается

1.3 Этапы ручного тестирования веб-приложений

1.4 Источники требований: что такое техническое задание и как проводить ручное тестирование, если техническое задание отсутствует

Глава 2. Подготовка к ручному тестированию веб-приложения (~15 стр)

2.1 Составление чек-листов

2.2 Составление тест-кейсов

2.3 Разработка тест-плана

2.4 Тест-дизайн и тест-аналитика

2.5 Баг-репорты и багтрекинговая система

Глава 3. Проведение тестирования и разработка предложений по улучшению тестирования Интернет-магазина РИВ ГОШ (~20 стр.)

3.1 Тестирование функциональности

3.2 Тестирование юзабилити

3.3 Тестирование интерфейса

3.4 Тестирование совместимости

3.5 Тестирование производительности

3.6 Тестирование безопасности

3.7 Подведение итогов. Разработка предложений по улучшению ручного тестирования веб-приложения РИВ ГОШ

Заключение (~ 4 стр.)

Список используемой литературы

Приложения

# Введение

**Введение** раскрывает обоснование необходимости исследования выбранной студентом задачи и представляет схему проведения дипломного исследования.

В общих чертах, в этом блоке должно быть:

* Что из себя представляет проект.
* Обоснование темы проекта.
* Цель проекта.
* План работы.
* Какую проблему будет решать ваш проект.
* По какой специализации вы будете делать дипломный проект.
* Есть ли у вас полезный опыт для решения этой задачи. Если да, опишите его.
* Какими инструментами вы будете пользоваться.
* Какие технологии вы планируете использовать при выполнении проекта.
* Состав команды. Даже если делали все сами, напишите, какие роли вы выполняли на проекте.

Хороший пример: Участвовала в командном проекте в роли Java-разработчика. Также на проекте были задействованы: проджект-менеджер, дизайнер и тестировщик.

### Пример

*Данный пример не является эталонным, лишь один из вариаций.*

**Тема проекта:** Исследование особенностей организации процесса ручного тестирования веб-приложений в условиях отсутствия технического задания на примере тестирования интернет-магазина парфюмерии и косметики.

**Цель:** Изучить особенности ручного тестирования веб-приложений в условиях отсутствия спецификации к программному продукту и разработать предложения по улучшению процесса ручного тестирования веб-приложения РИВ ГОШ.

**Какую проблему решает:** Случается такое, что необходимо протестировать и выпустить продукт без требований, или на их изучение нет времени.

**Задачи:**

1. Изучить литературу, касающуюся темы исследования.
2. Рассмотреть основные виды и методы ручного тестирования веб-приложений.
3. Ознакомиться с основными принципами составления, такой тестовой документации, как чек-листы, тест-кейсы, тест-планы, баг-репорты.
4. Составить план ручного тестирования веб-приложения РИВ ГОШ с использованием техник тест-дизайна.
5. Выполнить ручное тестирование веб-приложения РИВ ГОШ.
6. Разработать предложения по улучшению ручного тестирования веб-приложения РИВ ГОШ.

**Инструменты:** Postman, Qase.io, TestRail или Jira, DevTools, Git, Charles Proxy, FastStone Capture 7.2.

**Состав команды**: ФИО (Тестировщик)

# Теоретическая и практическая главы

**Основная часть** составляет 80% от всего объёма написанного. В нём последовательно рассказывается, как были решены вопросы, перечисленные во вступлении, какие цели достигнуты и с каким результатом.

Содержит 2 главы — теоретическую и практическую, где автор выражает свою позицию относительно гипотез.

В практической части студент должен пошагово описать:

* как он делал проект;
* какие инструменты использовал в работе;
* на каком этапе использовал инструменты и как они помогли в достижении цели работы.

В основной части должны быть практические примеры, результаты и выводы после каждого элемента исследования.

# Заключение

В заключение необходимо включить следующее:

1. Краткие и ёмкие теоретические и практические выводы, которые были получены во время анализа теоретической базы и практического исследования.
2. Оценка проведённого исследования, описание его результатов.
3. Практическая значимость работы, рекомендации и планы на дальнейшие исследования.
4. Общий итог — достижение цели, выполнение задач, доказательство гипотезы.
5. Предложения по совершенствованию объекта исследования.

# Список используемой литературы

Здесь нужно будет указатьсписок используемой литературы, ссылки на все ресурсы, которые нужны были для создания проектной работы.

Основные правила оформления использованной литературы и ресурсов:

1. Каждый источник упоминается единожды, независимо от того, насколько часто на него ссылаются.
2. Список литературы оформляется в алфавитном порядке по фамилии автора, сначала русскоязычная литература, затем иностранная, далее интернет-сайты.
3. Библиографическая запись обязательно включает:
   * Фамилию автора или фамилии их группы, инициалы (при наличии).
   * Название статьи, книги, справочника, закона, иного документа.
   * Населённый пункт, в котором был издан источник, наименование издательства.
   * Год публикации.
   * Число страниц.

## Пример

* *Книга: Автор. Название книги. Город: Издательство, Год.*
* *Статья: Автор. "Заголовок статьи." Название журнала Том, номер (Год): страницы.*

# Приложения

В **приложения** обычно входят артефакты, получившиеся в процессе создания проекта:

1. Объёмные графики и таблицы, которые не помещаются на лист А4.
2. Длинные математические формулы и расчёты по ним.
3. Характеристики аппаратуры, которая использовалась для проведения исследования.
4. Авторские методики.
5. Вспомогательный материал: тесты, карточки, схемы, рисунки.
6. Материалы, полученные на предприятии: отчёты, прочие документы.

СОДЕРЖАНИЕ:

1. Введение.

Django - это бесплатный и открытый фреймворк для веб-разработки, написанный на языке программирования Python. С его помощью разработчики могут создавать мощные веб-приложения, включающие в себя функциональность аутентификации, управления базами данных, обработки форм, административного интерфейса и многое другое.

Одним из главных преимуществ Django является его модульность. Он поставляется вместе с широким набором готовых компонентов, которые можно использовать для разработки различных типов веб-приложений. Более того, Django также обеспечивает четкую структуру и рекомендации по организации кода, что делает проекты на этом фреймворке легко читаемыми и поддерживаемыми.

Ещё одной замечательной особенностью Django является его ORM (объектно-реляционное отображение). Он позволяет разработчикам взаимодействовать с базами данных, используя объектно-ориентированный подход, простые Python-классы и методы. Это существенно упрощает задачи, связанные с манипуляцией данными, и позволяет сконцентрироваться на бизнес-логике приложения.

В Django также встроена система аутентификации и авторизации, что делает процесс создания пользовательских систем веб-приложений очень простым. Фреймворк предоставляет множество готовых компонентов для работы с пользователями, таких как регистрация, восстановление пароля, управление правами доступа и многое другое.

Ещё одна важная возможность Django - это его встроенный административный интерфейс. Он позволяет быстро создавать и настраивать административные панели для управления данными вашего веб-приложения. Административный интерфейс поставляется с готовыми компонентами, такими как фильтры, поиск, сортировка и дополнительные настраиваемые поля.

Кроме того, документация Django является одной из лучших среди фреймворков веб-разработки. Она обеспечивает подробные объяснения и примеры использования каждой функции, что делает процесс изучения и работы с Django очень комфортным.

В целом, Django - это мощный и гибкий фреймворк, который позволяет разрабатывать сложные веб-приложения с минимальными усилиями. Он активно поддерживается сообществом разработчиков и имеет множество расширений и пакетов для решения различных задач. Изучение Django может быть полезным шагом для любого веб-разработчика, стремящегося создавать высококачественные и эффективные веб-приложения.

Другим значимым преимуществом Django является его встроенная система обработки форм. Фреймворк предоставляет разнообразные классы и методы для создания и валидации форм на стороне сервера. Это упрощает процесс получения и отправки данных от пользователей, а также обеспечивает проверку их правильности. Django также имеет встроенную защиту от атак CSRF (межсайтовая подделка запроса), что повышает безопасность веб-приложений и защищает пользователя от потенциальных угроз.

Еще одно преимущество Django заключается в его способности работать с различными базами данных, включая PostgreSQL, MySQL, SQLite и другие. Фреймворк предоставляет абстракцию базы данных, что делает возможным использование разных провайдеров без необходимости изменения кода приложения. Это позволяет разработчикам выбирать наиболее подходящую базу данных для их проекта и обеспечивает гибкость в работе с данными.

Большим плюсом Django является также его активное сообщество разработчиков. Django имеет обширную документацию и множество сторонних библиотек, расширяющих его функциональность. Это означает, что вы можете получить помощь и поддержку в разработке своего проекта, а также воспользоваться готовыми решениями и инструментами, разработанными сообществом.

И наконец, Django очень надежный и масштабируемый фреймворк. Он способен обрабатывать высокие нагрузки и обеспечивать стабильную работу веб-приложений даже при большом количестве пользователей. Фреймворк предлагает различные инструменты для оптимизации производительности и масштабирования приложения, такие как кэширование, асинхронные задачи и горизонтальное масштабирование.

В общем, Django - это мощный и гибкий фреймворк для веб-разработки, который предоставляет широкие возможности для создания разнообразных веб-приложений. Он позволяет разработчикам сосредоточиться на бизнес-логике, вместо написания основного инфраструктурного кода, и обеспечивает быструю и надежную разработку веб-приложений.

Хочу также обратить внимание на понятность и интуитивность Django. Он имеет простую и логичную структуру файлов и каталогов, что позволяет разработчикам быстро ориентироваться в проекте. Кроме того, Django использует паттерн проектирования Model-View-Controller (MVC), что способствует логическому разделению кода и повышает его поддержку и переиспользование.

Одним из ключевых преимуществ Django является его возможность разработки масштабируемых и гибких REST API. Фреймворк предоставляет инструменты для создания и манипулирования REST-ресурсами, а также для автоматической генерации документации API. Это делает Django отличным выбором для разработки веб-сервисов и микросервисов.

Кроме того, Django предлагает удобную систему миграции базы данных, которая позволяет разработчикам вносить изменения в модели данных и автоматически применять их к существующим базам данных. Это упрощает процесс обновления и поддержки базы данных и минимизирует время простоя приложения.

Наконец, Django обладает широким спектром инструментов для разработки веб-приложений, таких как аутентификация и авторизация пользователей, административная панель, работы со статическими файлами и многое другое. Фреймворк также предлагает готовые инструменты для интеграции с другими технологиями, такими как Celery для асинхронных задач, Elasticsearch для поиска и Redis для кэширования.

Еще одним замечательным преимуществом Django является его обширная и активная сообщество разработчиков. Django обладает огромной базой документации, онлайн ресурсов, сообществ фанатов и разработчиков, которые готовы помочь и поделиться своим опытом. Это обеспечивает отличную поддержку и быстрое решение возникающих проблем.

Кроме того, Django имеет встроенную систему безопасности, которая помогает предотвратить такие уязвимости, как подделка формы (CSRF), инъекция SQL и многое другое. Фреймворк предоставляет набор инструментов и методов для обеспечения безопасности вашего приложения и защиты данных пользователей.

Важным аспектом Django является его возможность интеграции с другими технологиями и сторонними сервисами. Django предлагает разнообразные сторонние пакеты и расширения, которые позволяют вам интегрировать ваше приложение с популярными сервисами, такими как Amazon Web Services, Google Cloud, Twilio и другие. Это позволяет создавать мощные и гибкие приложения, а также использовать современные технологии для улучшения функциональности вашего проекта.

В целом, Django - это мощный фреймворк, который обладает множеством преимуществ и предлагает множество возможностей для разработки веб-приложений. Он позволяет создавать высококачественные, масштабируемые и безопасные приложения, сокращает время разработки и упрощает поддержку вашего проекта.

Надеюсь, эта информация помогла вам лучше понять преимущества Django.