Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных технологий

Отчет о прохождении производственной практики

Выполнил студент группы 22307
Клюшов Никита Константинович

Направление подготовки:
Программная инженерия

Место прохождения практики:
Кафедра информатики и
математического обеспечения

Сроки прохождения практики:
29.05.2023-11.06.2023

Руководитель практики:
к.т.н., доцент
Богоявленская Ольга Юрьевна

Оценка _____

Дата _____

Петрозаводск 2023 г.

Перечень терминов

В ходе описания отчёта о практике по научно-исследовательской работе будут использоваться следующие термины:

- Flask - веб-фреймворк для языка Python, предоставляющий инструменты для создания веб-приложений.
- Blueprint - концепция в веб-фреймворке Flask для создания компонентов приложений и поддержки общих шаблонов внутри приложения или между приложениями.

Оглавление

Введение

В рамках данного отчета по производственной практике будет представлен обзор проекта, выполненного в рамках курса ТППО, с фокусом на две основные темы: анализ существующих аналогов проекта и анализ использованных методов, научных статей, разработанных систем, которые были полезны при разработке нашей системы, а также книги/методы, использованные для ее разработки.

Анализ существующих аналогов проекта имеет решающее значение для понимания текущего состояния данной области и определения лучших практик. Рассмотренны системы, использующие технологию Blueprint. Также было обращено внимание на системы-аналоги, способные автоматически генерировать отчеты в формате Word из хранящейся в базе данных информации. Этот анализ позволил выделить основные преимущества и недостатки существующих решений и использовать их в качестве основы для разработки нашей системы.

Вторая тема отчета связана с использованными методами. Мы разработали подсистему, которая является частью более крупной системы. В процессе разработки нашей системы мы опиралась на методы и научные результаты, полученные в ходе исследований в данной области. Важно отметить ключевые методы, результаты и характеристики, которые привели нас к достижению поставленных целей.

Описание разрабатываемой системы.

AutoReport — сервис автоматизации формирования отчетов. Является расширением функциональности сайта ИМИТ ПетрГУ.

Проект заключается в создании дополнительной функциональности сайта ИМИТ ПетрГУ, раздела "отчётность студентов":

- добавление возможности автоматического создания отчётов на основании заранее имеющихся в файловой системе учебных планов образовательных направлений института;
- добавление возможности вставить информацию о практиках в процессе формирования документа отчёта;
- добавление возможности сохранения и редактирования отчётов на сайте.

Основные функции, реализуемые в сервисе Autoreport:

- заполнение данных о практиках групп при помощи диалоговой системы;
- формирование отчета - при нажатии пользователем на кнопку создания отчёта сервис формирует отчёт по заданным параметрам;
- возможность посмотреть вариант сформированного отчёта;
- возможность опубликовать сформированный отчёт на странице отчётности студентов.

Глава 1

Существующие аналоги.

1.1 Автоматическое формирование отчётов.

К сожалению, рассматривать аналоги систем для формирования отчётов на данный момент не имеет смысла по ряду причин:

1. На данный момент не существует отдельных систем для автоматизации формирования отчётов на языке программирования Python.

2. Подавляющее большинство существующих систем для автоматизации формирования отчётов, такие как Tableau, Crystal Reports и другие, представляют собой большие, самостоятельные приложения с множеством функций по визуализации и экспортированию данных, что не подходит для нашей системы.

Формирование отчётов в языке программирования Python легко осуществить модульно при помощи библиотеки *python-docx*, что не будет рассматриваться в данном отчёте, так как не являлось частью моей работы над проектом.

1.2 Использование технологии Blueprint.

Технология Blueprint в фреймворке Flask представляет собой механизм для организации и структурирования веб-приложений на основе Flask, который позволяет разделять функциональность приложения на отдельные модули. Blueprint предоставляет способ создания множества маршрутов, обработчиков и шаблонов внутри отдельных модулей, которые затем могут быть зарегистрированы в основном приложении Flask.

1.3 FlaskBB.

FlaskBB - это система управления форумом, основанная на фреймворке Flask для языка программирования Python. Она предоставляет инструменты и функциональность для создания и управления онлайн-форумом.

FlaskBB следует модульной архитектуре, где каждый Blueprint представляет отдельный модуль или компонент форума. Например, Blueprint может быть создан для компонента пользователей, другой Blueprint для компонента тем и т.д. Blueprint определяет маршруты, функции представления и другие компоненты, связанные с конкретным модулем.

Если упростить, свести к нашей задаче рассмотрения технологии Bluprint, структуру кода FlaskBB можно представить следующим образом:

- flaskbb/ (корневая папка приложения):
  - *__init__.py*: Основной файл приложения, который инициализирует экземпляр Flask и регистрирует Blueprint'ы.
  - *config.py*: Файл конфигурации приложения, где можно определить настройки и переменные окружения.
  - *models.py*: Модуль, содержащий определения моделей данных, таких как пользователи, темы, сообщения и другие.
  - *views/*: Папка, содержащая модули с функциями представления для каждого Blueprint'а.
  - *templates/*: Папка, содержащая шаблоны HTML-страниц.
  - *static/*: Папка, содержащая статические файлы, такие как изображения, CSS и JavaScript файлы.

1.4 Анализ существующего решения.

В конечном счёте, удалось выделить следующие преимущества и недостатки использования технологии Blueprint в нашем проекте Autoreport.

Преимущества:

1. Blueprint позволяет разбить приложение на отдельные модули или компоненты, связанные с определенной функциональностью. Это улучшает организацию кода и делает приложение более модульным и понятным.

2. Blueprint позволяет повторно использовать код, определенный в одном Blueprint, в других частях приложения. Например, если у вас есть Blueprint для аутентификации пользователей, вы можете повторно использовать его в разных частях приложения без необходимости дублирования кода.

3. Blueprint позволяет добавлять или удалять компоненты из приложения без значительных изменений в других частях кода. Это упрощает масштабирование приложения и добавление новых функциональных возможностей.

4. Каждый Blueprint является независимым модулем, который может иметь свои собственные маршруты, представления и статические файлы. Это обеспечивает изоляцию компонентов и предотвращает конфликты между ними.

5. Blueprint упрощает тестирование отдельных компонентов приложения, поскольку они могут быть протестированы независимо от других частей приложения. Это позволяет более эффективно писать и запускать тесты для каждого компонента.

Недостатки:

1. Использование Blueprint может добавить некоторую сложность в структуру и организацию приложения, особенно для небольших проектов. Если приложение достаточно простое, Blueprint может показаться излишним сложным инструментом.

2. Blueprint может привести к увеличению количества файлов в проекте, особенно если у вас есть множество Blueprint'ов. Это может затруднить навигацию по проекту и увеличить время разработки.

3. Использование Blueprint может привести к синтаксическим зависимостям между Blueprint'ами и приложением в целом. Некорректное определение или регистрация Blueprint может привести к ошибкам в приложении.

4. При использовании Blueprint отладка может быть сложнее, особенно при обработке ошибок. Использование Blueprint требует более внимательного отношения к трассировкам стека и местам возникновения ошибок.

В конечном счёте было принято решение об использовании технологии Blueprint при разработке сервиса Autoreport, ввиду того, что:

- отсутствует непосредственный доступ к самой системе сайта ИМИТ;
- отсутствует опыт при разработке таких больших систем;
- в случае серьёзных неполадок изменение и/или отключение одного модуля, подключённого при помощи технологии Blueprint, гораздо проще и безопаснее, чем откат до предыдущего состояния или подобных методов.

Глава 2

Использованные методы.

В процессе разработки сервиса Autoreport были использованы некоторые методы, которые будут перечислены в данной главе.

2.1 Методы исследования.

- Литературный обзор - анализ существующей научной и технической литературы, связанной непосредственно с фреймворком Flask, формированием отчётности с помощью языка программирования Python, использованием технологии Blueprint и прочим.

- Исследование предметной области - изучение особенностей и требований предметной области, для которой разрабатывается ПО. В частности это сайт ИМИТ, процесс формирования отчётов, их внешний вид и хранение учебной информации из файлов с учебными данными.

- Анкетирование и опросы - выявление данных и мнений у заинтересованных сторон, таких как пользователи (через заказчика) и разработчики.

- Прототипирование - создание пробной версии ПО изучения его функциональности, удобства использования и взаимодействия с пользователями. Как пример - прототип интерфейса, который согласовывался с заказчиком.

2.2 Инженерия требований.

- Сбор требований - выявление полного и точного списка требований, учитывая потребности пользователей и ограничения системы.

- Анализ требований - оценки, структурирования и анализа собранных требований. Это включает идентификацию противоречий, неоднозначностей и недостаточностей в требованиях, а также установление связей между требованиями и их приоритизацию, что помогает обеспечить согласованность и полноту требований, а также идентифицировать возможные проблемы или риски.

- Документирование требований - создание формальных документов, спецификаций требований и моделей, которые описывают функциональные и нефункциональные требования к системе. Документирование требований

служит основой для коммуникации между заказчиками, пользователем и командой разработки.

- Валидация требований - процессы проверки и подтверждения, что собранные требования действительно отражают потребности заказчика и пользователя, включая взаимодействие с заинтересованными сторонами, проведение проверок достижимости требований, проведение прототипирования и тестирования, а также использование формальных методов верификации и анализа.

2.3 Архитектурный анализ.

- Разбор архитектуры - изучение архитектуры системы с целью понимания ее основных компонентов, связей и взаимодействий между ними, что помогает определить ключевые аспекты системы и ее структуру. Данный метод использовался как для анализа похожих систем, так и для анализа архитектуры сайта ИМИТ.

- Статический анализ кода - оценка структуры кода, качества и соответствия архитектурным принципам и правилам.

- Динамический анализ кода - исследование поведения кода во время выполнения для обнаружения потенциальных проблемм, таких как недостаточно обработанные ошибки и исключения, неоптимальное использование или утечка ресурсов и т.д.

2.4 Конфигурационное управление.

- Контроль версий - отслеживание изменений в конфигурации ПО и параллельная работа участников команды над проектом. В качестве системы контроля версий сервиса Autoreport наша команда использовала git.

Заключение

В ходе данной практики был проведен обзор проекта, который включал две основные темы: анализ существующих аналогов проекта и анализ использованных методов. Также была рассмотрена система FlaskBB, представляющая собой аналог (в рамках использования технологии Blueprint), приведены причины отсутствия необходимости в анализе существующих систем для автоматической генерации отчёта и приведён список методов, использование которых помогло во время разработки сервиса Autoreport.

Литература

[1] Документация на Flask (русский перевод) Выпуск 0.10.1 февр. 24, 2019

[2] Pypi.org: [Электронный ресурс]. - [Б.м], 2022. - URL: https://pypi.org/ (дата обращения: 16.05.2023). - Текст: электронный.

[3] Flask.palletsprojects.com: [Электронный ресурс]. - [Б.м], 2022. - URL: https://flask.palletsprojects.com/ (дата обращения: 21.05.2023). - Текст: электронный.

[4] Система FlaskBB: сайт. - URL: https://flaskbb.org (дата образения: 03.06.2023). - Текст: электронный.

[5] Официальная документация к системе FlaskBB: сайт. - URL: https://flaskbb.readthedocs.io (дата обращения: 03.06.2023). - Текст: электронный.

[6] Flask.palletsprojects.com: [Электронный ресурс]. - [Б.м], 2022. - URL: https://flask.palletsprojects.com/ (дата обращения: 21.05.2023). - Текст: электронный.

[7] Вигерс, К.И. Разработка требований к программному обеспечению / К.И. Вигерс. - Пер. с англ. - М.: Издательско-торговый дом "Русская Редакция 2004. - 576 с. - ISBN 5-7502-0240-2

[8] Орлов, С.А. Программная инженерия. Технологии разработки программного обеспечения. 5-е издание обновленное и дополненное. Стандарт третьего поколения. СПб.: Питер, 2016. 640 с. (Серия «Учебник для вузов»).