МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики  
Кафедра информационных технологий

**ОТЧЁТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЕ №10 ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИИ ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»**

Работу выполнила студентка 4ИТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. А. Арабова

Проверил доцент кафедры ИТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Н.Полетайкин

Краснодар

2023

ВВЕДЕНИЕ

Тема: Документирование и развертывание ПС

Цель: Освоение методики документирования ПС.

Задание

1. Собрать документированные материалы, разработанные при выполнении лабораторных работ №1–9 в единый документ с разбивкой на разделы. Темы лабораторных работ обозначить названиями разделов. В разделы не включать цель, задание и выводы из отчетов о выполнении лабораторных работ.

2. Дополнительно отдельным разделом 10 описать назначение, технические характеристики, принцип работы и меры безопасности при эксплуатации ПС.

3. Дополнительно отдельным разделом 11 составить руководство пользователя. Индивидуальная тема: Сервис проверки корректности технической документации.

Раздел 1

Объект информатизации: кафедра вычислительных технологий.

Кафедра вычислительных технологий была создана в 1993 году.Ф.И.О. заведующего кафедрой - Еремин Артем Александрович, кандидат физико-математических наук, окончил Кубанский государственный университет по специальности "Прикладная математика и информатика" в 2008 году .

Структурная схема представлена на рисунке 1.

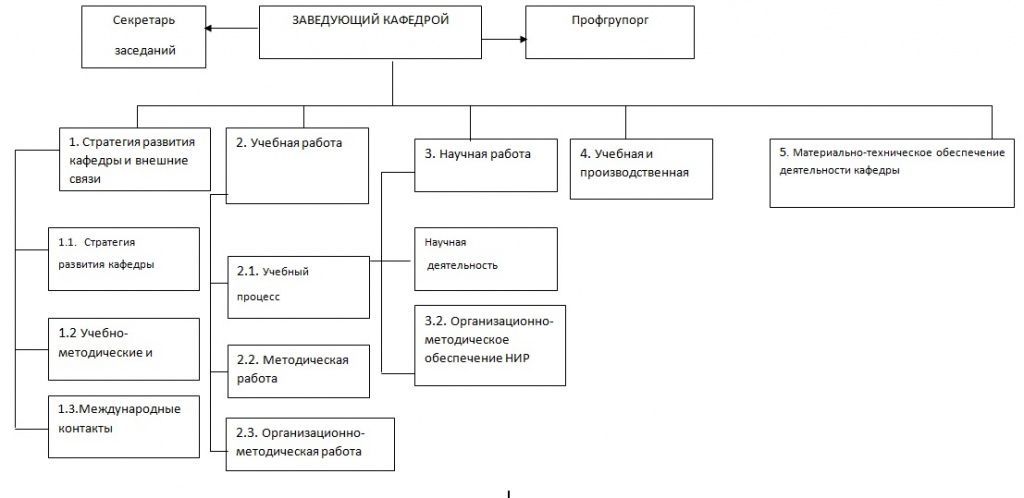


Рисунок 1 – Иллюстрация структуры объекта

Для информатизации выбирается процесс проверки технической документации на соответствие ГОСТу.

Проверка технической документации:

1. Загрузка технической документации
2. Загрузка шаблонов
3. Сравнение технической документации с шаблоном
4. Формирование отчёта о несходствах с ГОСТом

Действующие лица:

Технический эксперт

Процесс написания технической документации заключается в описании и сопровождении программного продукта в течение их жизненного цикла.

Входная и выходная информация, структурная схема типа “чёрный ящик” представлена на рисунке 2.

Изображение выглядит как текст, снимок экрана, Шрифт, Графика

Автоматически созданное описание

Рисунок 2 – Структурная схема проверки технической документации

Принцип декомпозиции бизнес-процесса представлен на рисунке 3.



Рисунок 3 – Декомпозиция бизнес-процесса

Диаграмма прецедентов представлена на рисунке 4.

Изображение выглядит как текст, снимок экрана, круг, луна

Автоматически созданное описание

Рисунок 4 – UML-диаграмма прецедентов

Описание входных потоков рассмотрим в таблице 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Наименование и назначение потока | Форма представления | Обработчик (Кто обрабатывает) | Корреспондент  (Откуда поступает) | Характеристики обработки | |
| Трудозатраты, чел. ч. | Периодичность, регламент |
| 1 | Техническая документация | Текстовый документ | Программа проверки | Внешние носители, интернет | 2 | 1 раз в сутки |

Таблица 1 - Реестр входных информационных потоков

Описание выходных потоков рассмотри в таблице 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Наименование и назначение потока | Форма представления | Обработчик (Кто обрабатывает) | Корреспондент  (Куда поступает) | Характеристики обработки | |
| Трудозатраты, чел. ч. | Периодичность, регламент |
| 1 | Отчет о проверке технической документации в соответствии с ГОСТом | Статьи | Программа проверки | Сайт | 2 | 1 раза в стуки |

Таблица 2 - Реестр выходных информационных потоков

Ручной процесс предполагает собой, что пользователю будет необходимо самому изучить ГОСТы, что увеличивает время проверки технической документации. Ручной процесс предполагает собой три задачи:

1) изучение ГОСТов;

2) проверка технической документации;

3) формирование отчёта о несоответствиях с ГОСТом.

Все задачи будут автоматизированы.

Недостатками ручного процесса является:

1) трудоёмкость;

2) человеческий фактор;

**Раздел 2**

Pymorphy2

Pymorphy2 — морфологический процессор с открытым исходным кодом, предоставляет все функции полного морфологического анализа и синтеза словоформ. На рисунке 2.1 представлена установка библиотеки.

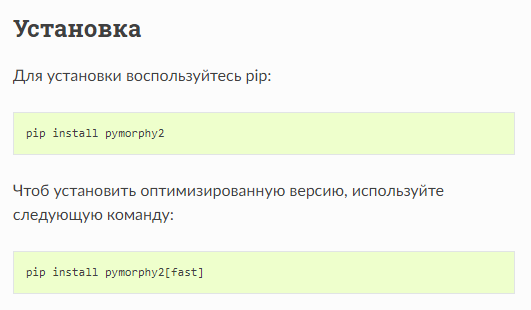


Рисунок 2.1 – Установка морфологического процессора

Плюсы:

1. является полностью открытым,
2. регулярно полоняется словарь;

Минусы:

1. проблема с при анализе прилагательных и глаголов,
2. проблема с при анализе предлогов.

Система Диалинг-АОТ.

Проект АОТ включает в себя практически все этапы автоматического анализа текстов на ЕЯ, в том числе и морфологический. На рисунке 2 представлен принцип обработки текста системой Диалинг-АОТ.

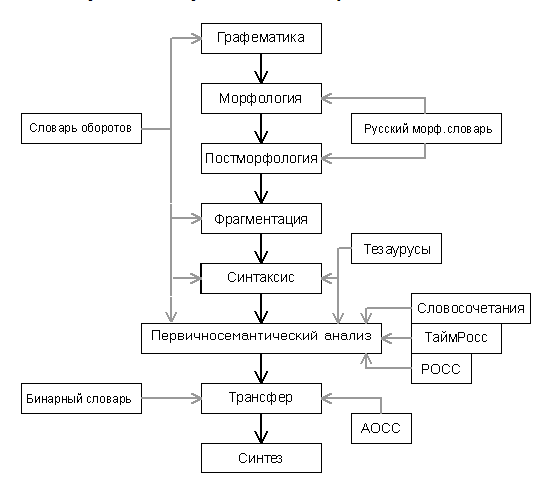


Рисунок 2 – Принцип обработки естественного языка

Плюсы:

1. изначальная направленность на лингвиста: поиск по предложениям или синтаксическим группам;
2. часто предоставляется полный исходный код программы,
3. низкая цена или даже отсутствие цены;

Минусы:

1. нет удобных средств пополнения словаря,
2. проект не поддерживается,
3. Обычно система работает либо на Linux, либо на Windows,
4. очень трудно, если возможно, расширять язык запросов, добавляя туда поиск по морфологическим и синтаксическим признакам,
5. результатом поиска является целый документ, тогда как лингвиста обычно интересует одно предложение.

В таблице 1 представлено сравнение Pymorpy2 и Диалинг-АОТ.

Таблица 1. Сравнение представленных систем

|  |  |  |
| --- | --- | --- |
| Показатель | Pymorpy2 | Диалинг-АОТ |
| Целостность | Сохраняет содержание данных | Сохраняет содержание данных |
| Рациональность | Оптимальное использование ресурсов | Оптимальное использование ресурсов |
| Работоспособность | Не требуется дополнительных подключений | Не требуется дополнительных подключений |
| Производительность | Информация отсутствует | Информация отсутствует |
| Гибкость | Регулярно обновляется словарь | Нет обновлений |
| Открытость | Информация отсутствует | Информация отсутствует |
| Потенциальная управляемость | Информация отсутствует | Информация отсутствует |
| Надёжность | Работает без сбоев | Работает без сбоев |
| Степень оперативности и надежности управления | Обеспечивает быстрый морфологический анализ | Обеспечивает быстрый морфологический анализ |
| Прогрессивность компьютерных и информационных технологий, задействованных в системе | Т.к производится обновление и поддержка работоспособности, поэтому является актуальным | Т.к не производится обновление и поддержка работоспособности, поэтому является актуальным |
| Сопровождаемость | Последнее обновление 27.09.2020 | Последнее обновление вышло в 2004 году |
| Стоимость | маленькая | большая |
| Эффективность | Позволяет произвести морфологический анализ за 2-3 минуты | Позволяет произвести морфологический анализ за 2-3 минуты |

Таблица 1 – Сравнение систем

Раздел 3

1. **Назначение и цель создания сервисa проверки технической документации**

Сервис проверки корректности технической документации – программа для проверки структуры технической документации и выявлении несоответствий с ГОСТом. Сервис предполагается для автоматизирования процесса проверки необходимых файлов.

Целью создания данного сервиса является минимизировать человеческий труд, тем самым уменьшив время проверки и улучшив ее качество.

1. **Разрабатываемый сервис должен решать такие задачи:**

* задача корректного составления структуры шаблона,
* задача проверки содержания технической документации,
* задача выделения ошибок,
* задача визуализации результата

**3.Функциональные требования**

**Требования к задаче «Задача корректного составления структуры шаблона»**

1. Шаблон должен быть составлен по ГОСТу 19.001-77 ЕСПД Общие положения, 19.201-78 ЕСПД Техническое задание. Требования к содержанию и оформлению, 19.105-78 ЕСПД Общие требования к программным документам.
2. Шаблон должен содержать все обязательные страницы технической документации.

**Требования к задаче «Задача проверки написанного текста»**

1. Написанный текст в содержании должен соответствовать ГОСТу 19.402-78 ЕСПД Описание программы, 19.401-78 ЕСПД Текст программы. Требования к содержанию и оформлению, содержать обязательные пункты и подпункты.

**Требования к задаче «Задача выделения ошибок»**

1. Найденные ошибки в содержании, нехватка обязательных страниц и прочие несоответствия должны быть выделены.

**Требования к задаче «Задача визуализации результата»**

1. В консоли написанного сервиса должны быть выведены ошибки, найденные в загруженном файле технической документации.

**Диаграмма требований (вариантов использования) UML представлена на рисунке 1:**

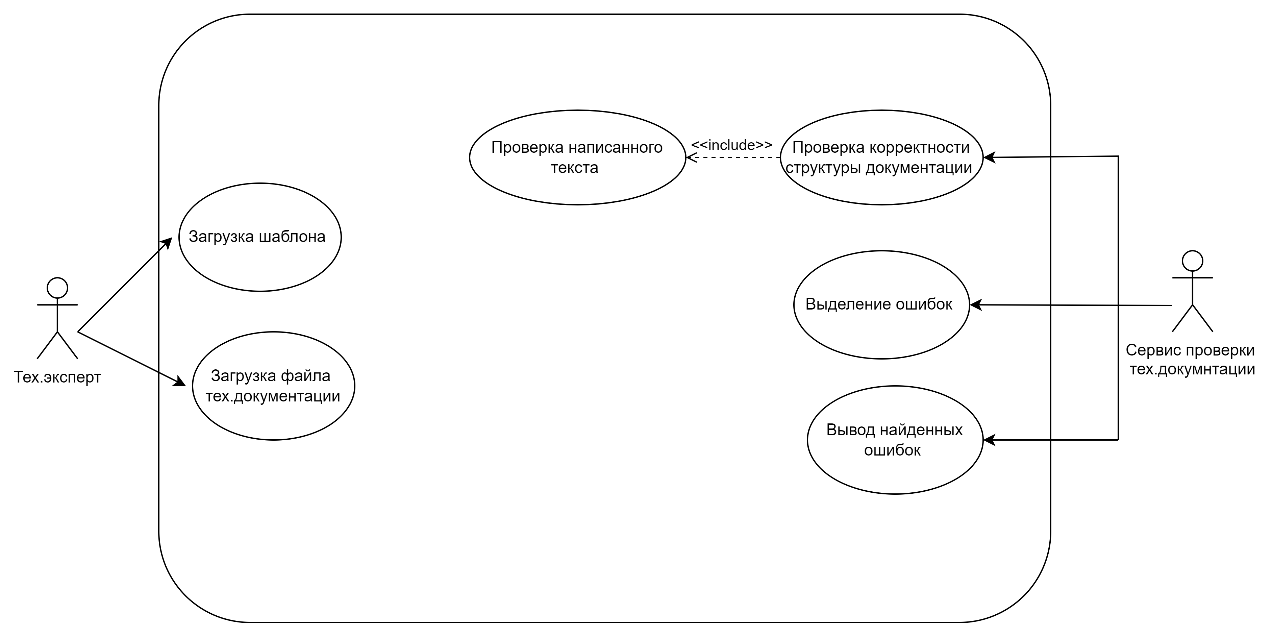


Рисунок 1 – Модель требований в нотации UML

**Требования к информационному обеспечению**

1. Установленный интерпретатор Python версии 3.2 и выше.
2. Библиотеки Python: `tkinter`, `pathlib`, `docx`, `docx2txt`.
3. Файлы DOCX : техническая документация и файл-шаблон.

**Требования к инструментальному программному обеспечению:**

Требования к операционной среде:

Необходимо обеспечить поддержку операционной системы Windows 10 и новее.

Требования к инструментальным средствам разработки ПО:

Язык программирования для создания приложения: версии Python 3.2 и выше.

**Нефункциональные требования к ПО:**

1. Сервис должен быстро выполнять задачи по анализу и сравнению документов, вне зависимости от размеров файлов.
2. Сервис должен надежно анализировать и сравнивать документы, не вызывая сбоев или ошибок.
3. Интерфейс пользователя должен быть простым и понятным, обеспечивая удобный способ выбора файлов для сравнения и анализа.
4. Сервис должен эффективно использовать ресурсы, чтобы снизить энергопотребление и уменьшить воздействие на окружающую среду.
5. Система должна иметь возможность модификации обеспечивающей части системы.

**Радел 4**

Контекстная диаграмма в нотации IDEF0.



Рисунок 1 – Диаграмма в нотации IDEF0

Диаграмма декомпозиции А0.

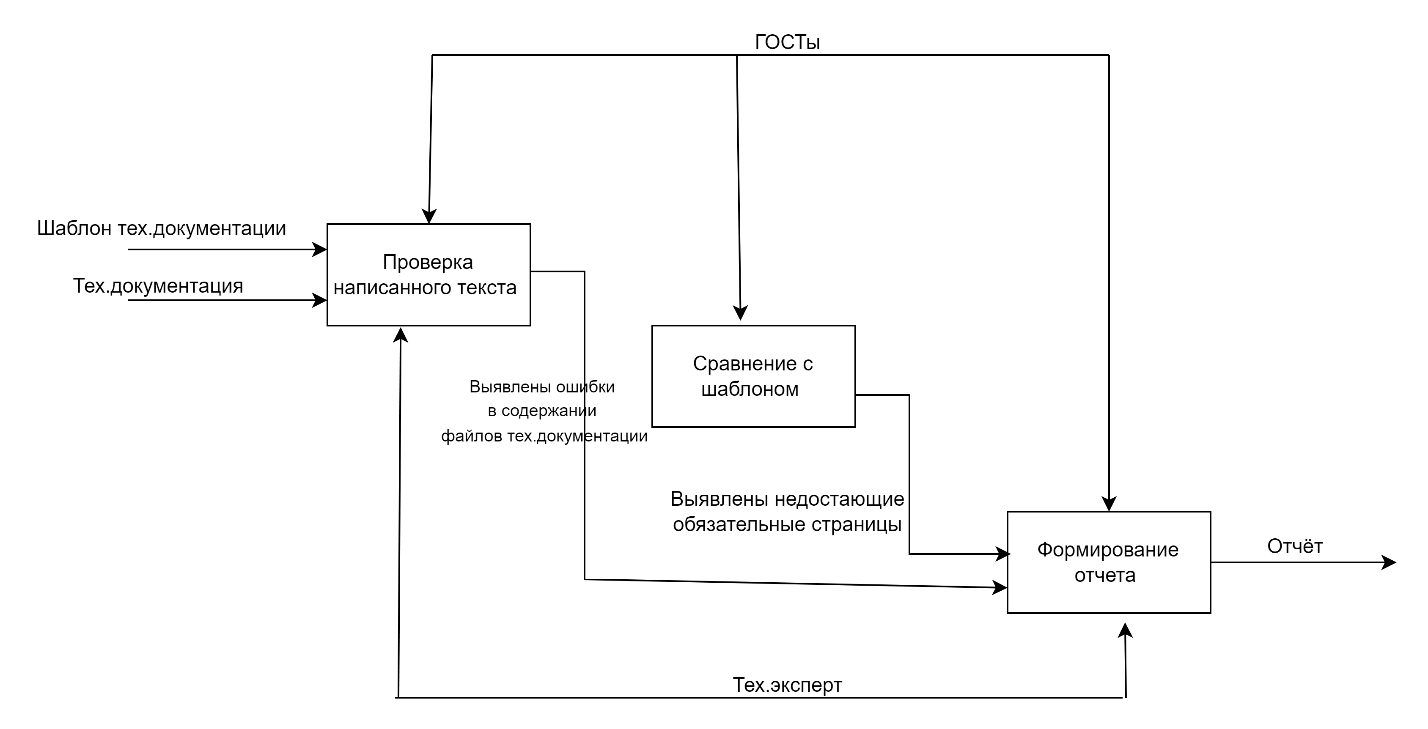


Рисунок 2 – Диаграмма декомпозиции А0

Таблица 1— описание элементов функциональной модели

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование стрелки | Источник стрелки | Тип стрелки источника | Приёмник стрелки | Тип стрелки приёмника |
| Загрузка тех.документации | Внешняя граница | Input | Создание шаблона тех.документации | Input |
| Готовый шаблон | Внешняя граница | Input | Загрузка шаблона тех.документации | Input |
| ГОСТы | Внешняя граница | Control | Проверка написанного текста | Control |
| Создание шаблона тех |
| Загрузка технической документации |
| Сравнение с шаблоном |
| Тех.эскперт | Внешняя граница | Mechanism | Создание шаблона тех.документации | Mechanism |
| Загрузка тех.документации |
| Формирование отчёта |
|  |
| Данные для отчёта | Проверка написанного текста | Output | Формирование отчётов | Input |
| Создание шаблона |
| Сравнение с шаблоном |
| Выделение ошибок |
| Отчёт | Формирование отчётов | Output | Внешняя граница | Output |

Декомпозиция на подзадачи

Проверка написанного текста



Рисунок 3 – Декомпозиция «Проверка написанного текста»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование стрелки | Источник стрелки | Тип стрелки источника | Приёмник стрелки | Тип стрелки приёмника |
| Тех.документации | Внешняя граница | Input | Проверка написанного текста | Input |
| Шаблон тех.документации | Внешняя граница | Input | Проверка написанного текста | Input |
| ГОСТы | Внешняя граница | Control | Проверка написанного текста | Control |
|
|
|  |  |
| Тех.эскперт | Внешняя граница | Mechanism | Проверка написанного текста  Шаблон тех.документации | Mechanism |
| Тех.документация |
| Формирование отчёта |
|  |
| Данные для отчёта | Проверка написанного текста | Output | Формирование отчётов | Input |
|
|
| Отчёт | Формирование отчётов | Output | Внешняя граница | Output |

Раздел 4

1. Абстракции подсистемы

Выделим основные абстракции подсистемы, разделим их по типам, приведём описание и запишем это в следующую таблицу:

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Абстракция** | **Тип** | **Описание** |
| 1 | Тех.эксперт | Сущность | Пользователь системы, создающий шаблон и файл технической документации, загружает их для проверки корректности по ГОСТу и анализирует результат |
| 2 | Интерфейс тех.эксперта | Интерфейс | Набор инструментов для взаимодействия инженера с системой |
|  | Загрузка тех.документации | Поведение | Способность загружать файл технической документации для проверки |
| 3 | Загрузка шаблона | Поведение | Способность загружать шаблон для сравнения с файлом технической документации, для нахождения недостающих обязательных страниц |
| 4 | Формирование отчёта | Поведение | Способность создавать с предсказаниями вольтамперных характеристик, сгенерированных НС |

Таблица 1 – Абстракция подсистемы

1. Классификация абстракций

|  |  |  |
| --- | --- | --- |
| **№** | **Класс** | **Список абстракций** |
| 1 | Люди | Тех.эксперт |
| 2 | Предметы | Отчёт |
| 3 | Концепции | ГОСТы, определяющие структуру и содержание тех.документации |
| 4 | События | * Загрузка файлов тех.документации * Загрузка шаблона |
| 5 | Показатели | Данные об ошибках и несоответствиях в тех.документации |

Таблица 2 – Классификация абстракций

1. Абстракции системы и их поведение

Проанализируем поведение выделенных абстракций. Выделим возможное поведение каждой абстракции в пределах рассматриваемой функции.

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Абстракция** | **Поведение** | **Описание поведения** |
| 1 | Тех. эксперт | Сотрудник, работающий с тех. документацией | Создает шаблон для тех.документации, проверяет ее находит ошибки и соответствия |
| 2 | Тех.эсперт | Сотрудник, который участвует в работе компании | Проверяет работу алгоритма на различных этапах |
| 3 | Пользователь сервиса | Лицо, использующее сервис | Загружает файлы для проверки |
| 4 | ГОСТы | Установка общих требований к тех.документации | Регулирование содержания и структуры файлов |
| 5 | Загрузка тех.документации | Работа тех.документацией | Часть системы, ответственная за загрузку, проверку тех.документации |
| 6 | Загрузка шаблона | Работа с шаблоном | Часть системы, ответственная за загрузку и сравнение файлов |
| 7 | Формирование отчёта | Формирование отчёта о несоответствиях с ГОСТом | Часть системы, ответственная за составление отчёта |

Таблица 3 – Абстракции подсистемы и их поведение

1. Диаграмма классов

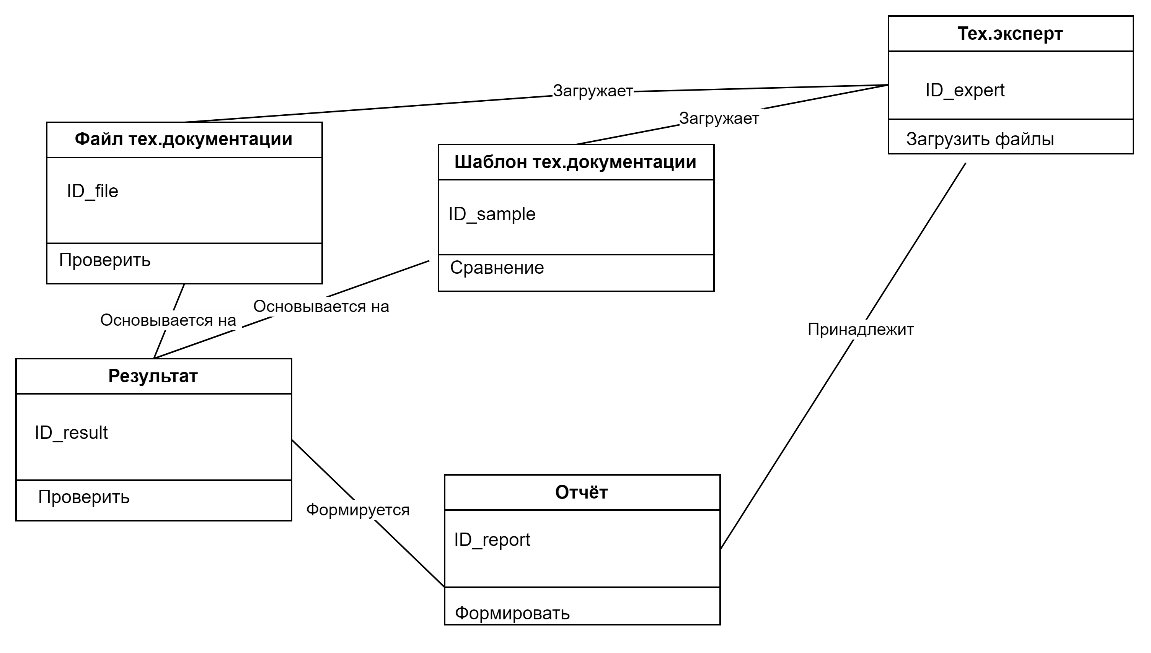


Рисунок 1 – Диаграмма классов

1. Диаграмма состояний (statechart diagram)

Диаграмма, описывающая состояние проверки технической документации на протяжении всего цикла работы сервиса:

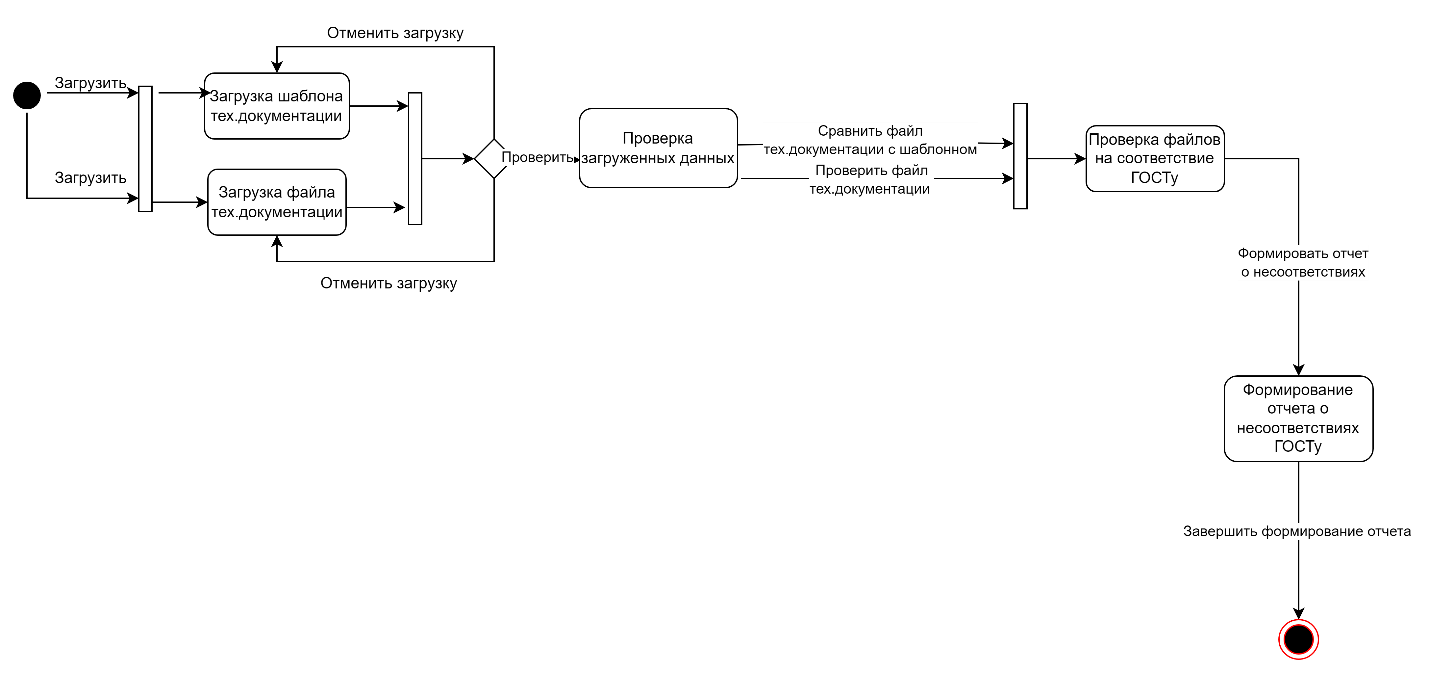


Рисунок 2 – Диаграмма состояний

1. Диаграмма активности (activity diagram)

Диаграмма, описывающая процесс проверки технической документации:

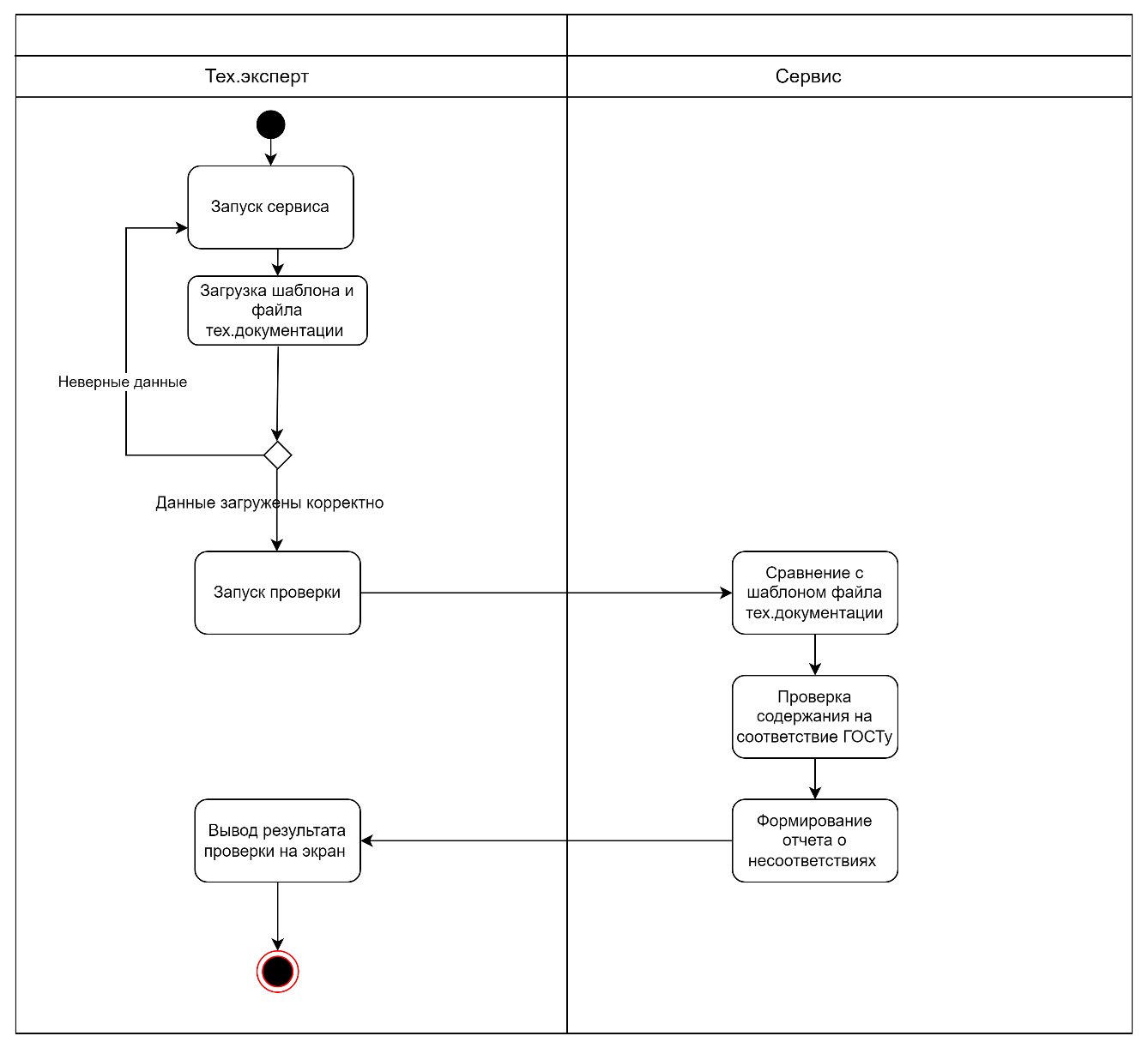


Рисунок 3 – Диаграмма активности

5.1 Список объектов для потока событий

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Прецедент** | **Объект** | **Описание объекта** |
| 1 | Запуск сервиса | Тех.эксперт | Тех.эксперт запускает сервис |
| 2 | Загрузка шаблона и файла тех.документации | Тех.эксперт | Загружает шаблон и файл тех.документации |
| 3 | Запуск проверки | Тех.эксперт | Запуск процесса проверки |
| 4 | Сравнение шаблона и файла тех. документации | Сервис | Сравнение шаблона и файла тех.документации |
| 5 | Проверка содержания на соответствие ГОСТу | Сервис | Проверка содержания файла тех.документации |
| 6 | Формирование отчета о несоответствиях ГОСТу | Сервис | Формирование отчета |
| 7 | Вывод результата проверки на экран | Тех.эксперт | Вывод результата |

Таблица 4 – Список объектов для потока событий

1. Диаграмма последовательностей (sequence diagram)

Диаграмма, описывающая жизненный цикл объекта в рамках прецедента.

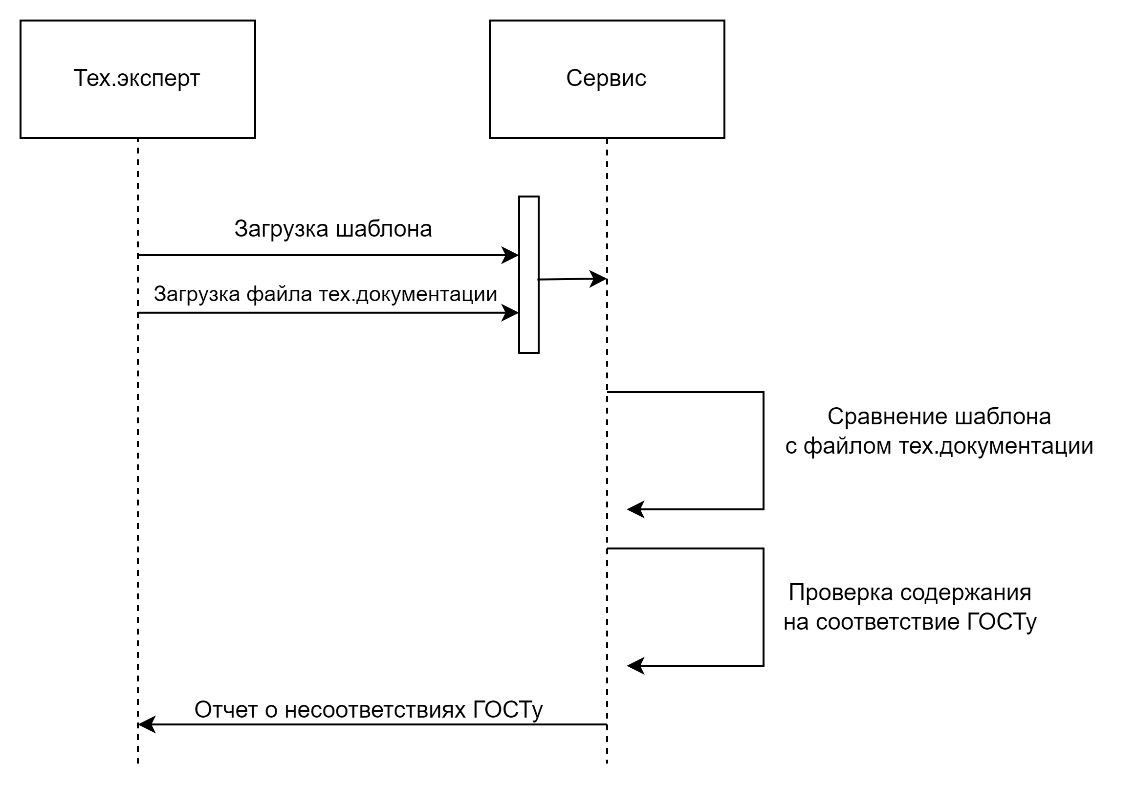


Рисунок 5 – Диаграмма последовательностей

**Раздел 6**

1. Идентификация сущностей информационной базы ПС и связей между ними.

|  |  |
| --- | --- |
| Документ | Сущность |
| Шаблон тех. документации | Sample |
| Тех. документация | Documentation |
| Тех. эксперт | Expert |

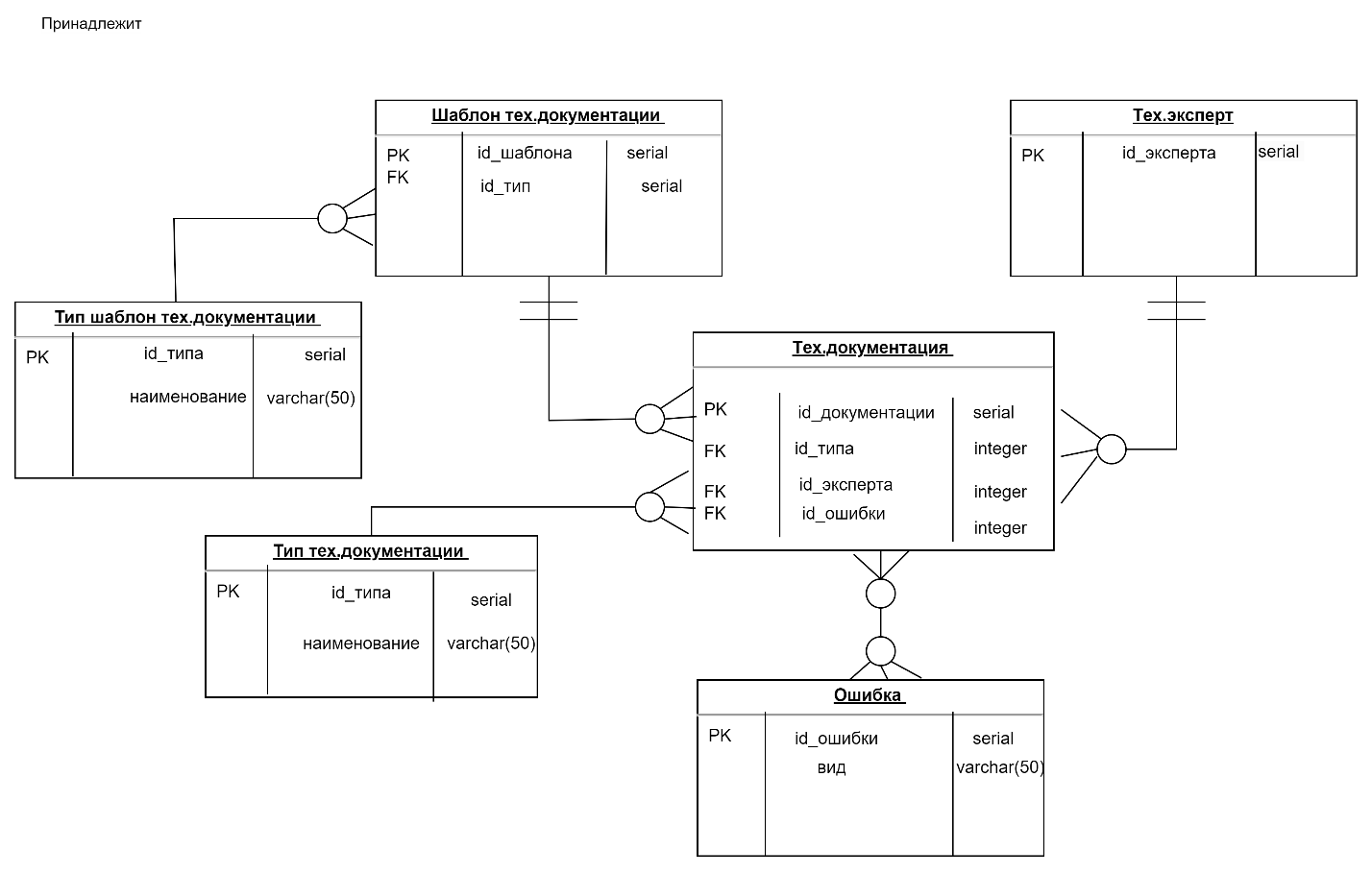
2. Логическая модель

Изображение выглядит как текст, снимок экрана, диаграмма, дизайн

Автоматически созданное описание

3. Исходя из построенной диаграммы классов и логической модели данных можно сделать вывод, что была выполнена правильная проектировка будущей базы данных. Была проведена нормализация данных.

4. Физическая модель данных:



|  |  |  |
| --- | --- | --- |
| Сущность | Атрибуты | Описание |
| Справочные | | |
| 1. Шаблон тех.документации | id\_шаблона, id\_типа | Информация о шаблонах |
| 1. Ошибки | id\_ошибки, вид | Информация об ошибках |
| 1. Тип шаблона тех.документации | id\_типа, наименование | Информация о типах шаблона тех.документации |
| 1. Тех.эксперт | id\_эксперта | Информация о тех.экспертах |
| 1. Тип тех.документации | id\_типа, наименование | Информация о типах тех.документации |
| Оперативные | | |
| 1. Тех.документация | id\_документации, id\_типа, id\_эксперта, id\_ошибки | Информация о тех.документации |

5.

CREATE TABLE IF NOT EXISTS Sample (

id\_sample serial primary key,

id\_type integer NOT NULL references Type\_sample(id\_type) ON DELETE CASCADE

);

CREATE TABLE IF NOT EXISTS Type\_sample (

id\_typesam serial primary key,

description varchar(50) NOT NULL

);

CREATE TABLE IF NOT EXISTS Expert (

id\_expert serial primary key

);

CREATE TABLE IF NOT EXISTS Error (

id\_error serial primary key,

view varchar(50) NOT NULL

);

CREATE TABLE IF NOT EXISTS Type\_documentation(

id\_typedoc serial primary key,

description varchar(30) NOT NULL

);

CREATE TABLE IF NOT EXISTS Documentation(

id\_documentation serial primary key,

id\_type integer NOT NULL references Type\_sample (id\_type) ON DELETE CASCADE,

id\_expert integer NOT NULL references Expert(id\_expert) ON DELETE CASCADE,

data date NOT NULL,

id\_error integer NOT NULL references Error(id\_error) ON DELETE CASCADE,

id\_type integer NOT NULL references Type\_documentation(id\_type) ON DELETE CASCADE

);

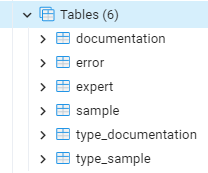
6. Для разработки была выбрана СУБД PostgreSQL по следующим причинам:

1) Бесплатная объектно-реляционная СУБД с открытым исходным кодом

2) Наличие удобного интерфейса при работе с БД

3) Высокая мощность и широкая функциональность

1. Созданные таблицы



1. Структура таблицы «Шаблон тех.документации»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_шаблона | serial | 4 | Not null |  | Primary key |
| id\_типа | integer | 4 | Not null |  | Foreign key |

Структура таблицы «Тех.эксперт»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_эксперта | serial | 4 | Not null |  | Primary key |

Структура таблицы «Тех.документация»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_документации | serial | 4 | Not null |  | Primary key |
| id\_объекта | integer | 4 | Not null |  | Foreign key |
| id\_эксперта | integer | 4 | Not null |  | Foreign key |
| дата | date | 5 | Not null |  |  |
| id\_ошибка | integer | 4 | Not null |  | Foreign key |
| id\_типа | integer | 4 | Not null |  | Foreign key |

Структура таблицы «Ошибка»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_ошибки | serial | 4 | Not null |  | Primary key |
| вид | varchar | 50 | Not null |  |  |

Структура таблицы «Тип шаблона тех.документации»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_тип | serial | 4 | Not null |  | Primary key |
| наименование | varchar | 50 | Not null |  |  |

Структура таблицы «Тип тех.документации»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| id\_типа | serial | 4 | Not null |  | Primary key |
| наименование | varchar | 50 | Not null |  |  |

9. Краткое описание таблиц

|  |  |
| --- | --- |
| Имя таблицы | Описание |
| 1. Объект | Информация об объектах |
| 1. Тип шаблона тех. документации | Информация о типе шаблона тех.документации |
| 1. Тех.эксперт | Информация о тех. документации |
| 1. Тип тех.документации | Информация о типе тех. документации |
| 1. Тех.документация | Информация о тех. документации |
| 1. Ошибка | Информация об ошибках |

Связи между таблицами

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Родительская таблица | | Дочерняя таблица | | Тип связи |
| Название | Атрибут PK | Название | Атрибут FK |
| Тип шаблона тех.документации | id\_типа | Шаблон тех.документации | id\_типа | Один ко многим |
| Тип тех.документации | id\_типа | Тех.документация | id\_типа | Один ко многим |
| Шаблон тех.документации | id\_шаблона | Тех.документация | id\_шаблона | Многие ко многим |
| Объект | id\_объекта | Тех.документация | id\_объекта | Один ко многим |
| Эксперт | id\_эксперт | Тех.документация | id\_эксперт | Многие ко многим |

**Раздел 7**

1. Идентификация сущностей информационной базы ПС и связей между ними.

Структурная схема общесистемного программного обеспечения:

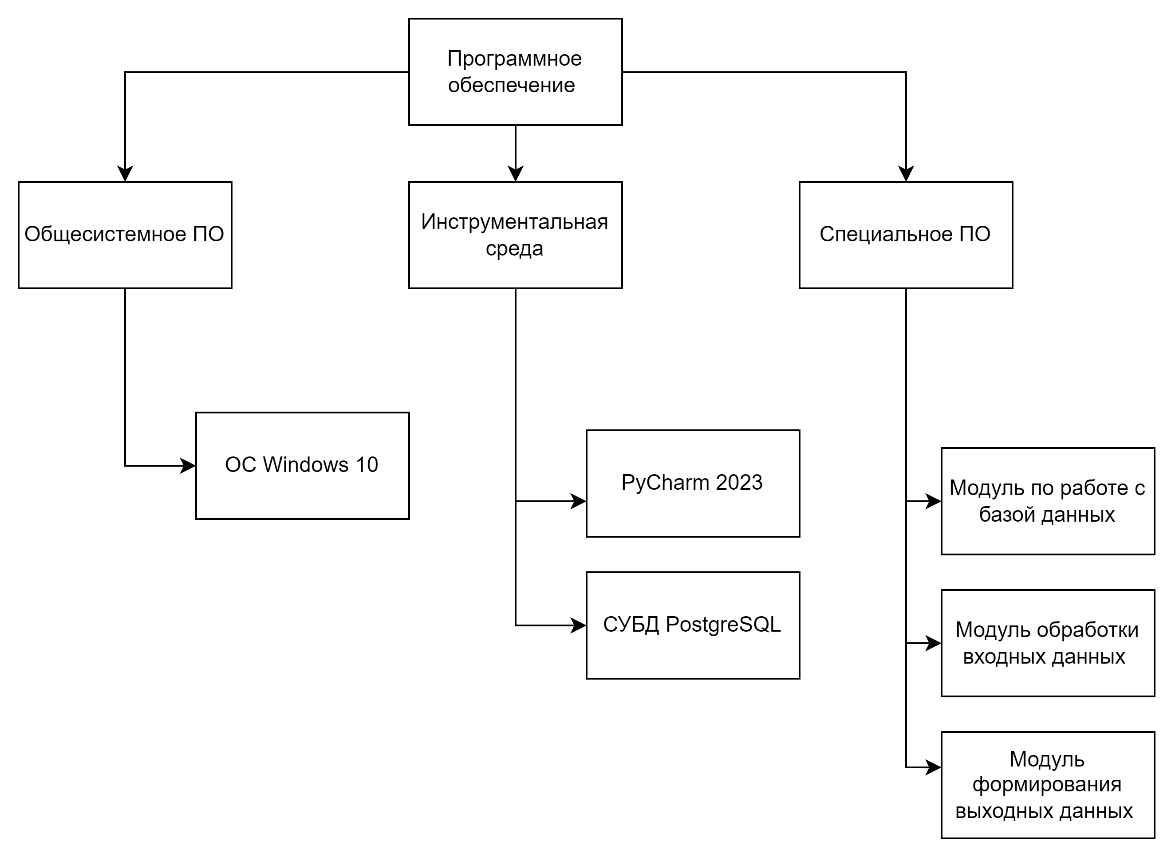


Рисунок 1 - Структурная схема

Перечень разработанных компонентов приложения

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя | Тип | Описание |
| 1 | checkingcorrectness.py | «source» | Реализация программного интерфейса. |
| 2 | checkingcorrectness.py | «executable» | Исходный файл программы, из которой загружаются зависимости |
| 3 | checkingcorrectness.py | «source» | Реализация основных функциональных возможностей программы в классах модулей с помощью простого и понятного графического интерфейса пользователя. |
| 4 | tkinter | «library» | Библиотека для создания графического интерфейса пользователя |
| 5 | sample | «table» | Таблица для хранения данных о шаблоне |
| 6 | expert | «table» | Таблица для хранения данных об эксперте |
| 7 | docx2txt | «library» | Библиотека, предназначенная для работы с файлами |
| 8 | psycopg2 | «library» | Библиотека, предназначенная для работы с базой данных PostgreSQL |
| 9 | documentation | «table» | Таблица для хранения данных о документации |
| 10 | docx | «library» | Библиотека, предназначенная для работы с файлами |
| 11 | pathlib | «library» | Библиотека, предназначенная для работы с путем файла |
| 12 | error | «table» | Таблица для хранения данных об ошибке |

Таблица 1 – Компоненты приложения

Диаграмма компонентов

Изображение выглядит как текст, снимок экрана, Самоклеющийся листок, Шрифт

Автоматически созданное описание

Рисунок 2 – Диаграмма компонентов

Физические элементы ПС и диаграмма развёртывания

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя | Тип | Описание |
| 1 | Компьютер | «processor» | Рабочая станция для взаимодействия с системой |

Таблица 2 - Физические элементы ПС

Изображение выглядит как текст, диаграмма, снимок экрана, Прямоугольник

Автоматически созданное описание

Рисунок 3 – Диаграмма развёртывания

Анализ сложности кода:

В результате анализа кода сложности были получены следующие результаты:

• 5 функций и методов

• 22 переменных

• 27 оператора

Экранные формы компонентов приложения

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 4 – Окно выбора файлов

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 5 – Пример работы модуля с техническим документом «Описание программы»

**Раздел 8**

1. Системное (ручное) пользовательское тестирование ПС.

Системное тестирование – это тестирование всей системы в целом, как правило, через ее пользовательский интерфейс. Следует проверить соответствует ли разработанная ПС заявленным требованиям.

Система адаптивна под различные разрешения экрана, интерфейс простой и незапутанный.

Для корректного тестирования необходимо провести 2 теста: со стороны пользователя и со стороны администратора.

1.1 Тест №1 (Инженер)

Состоит из проверки функционала пользователя состоит из следующих действий:

1) Запуск программы

2) Выбор файлов с неправильным расширением

3) Проверить валидацию данных

4) Выбор файлов с верным расширением

5) Выбор несоответствующего шаблона

6) Загрузить файл с экспериментальной ВАХ

7) Проверить, что программа корректно загружает файл и шаблон

8) Нажать на кнопку «Обработать» и проверить корректность работы

Ожидается, что все элементы интерфейса корректно работают. Этот тест успешно пройден.

Скриншоты срабатывания некоторых пунктов проверки:

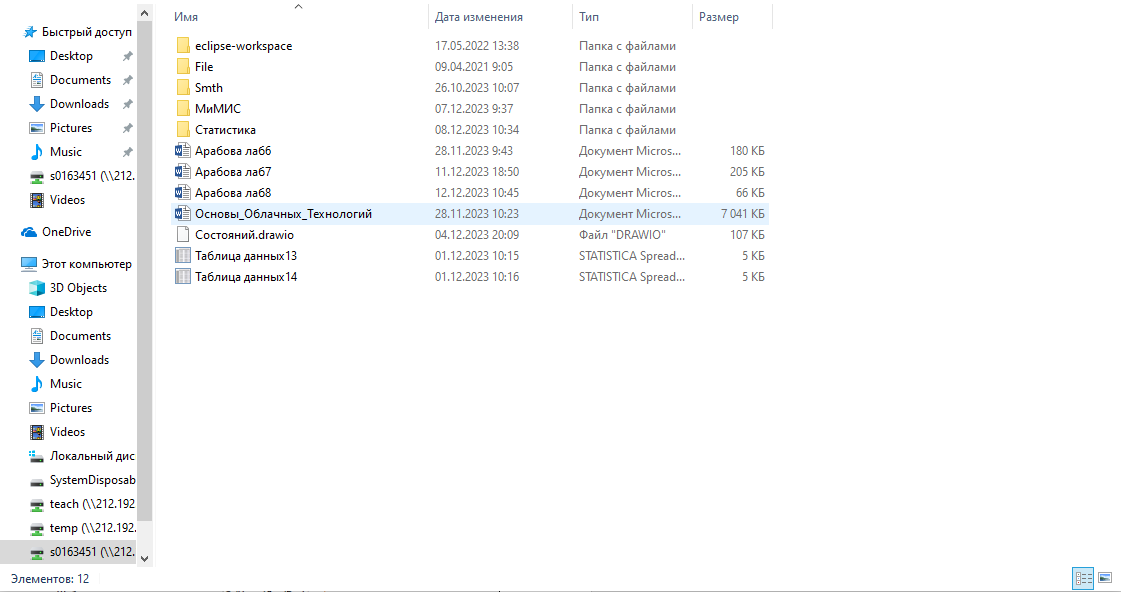


Рисунок – Окно выбора файла до нажатия «Выбрать файл»

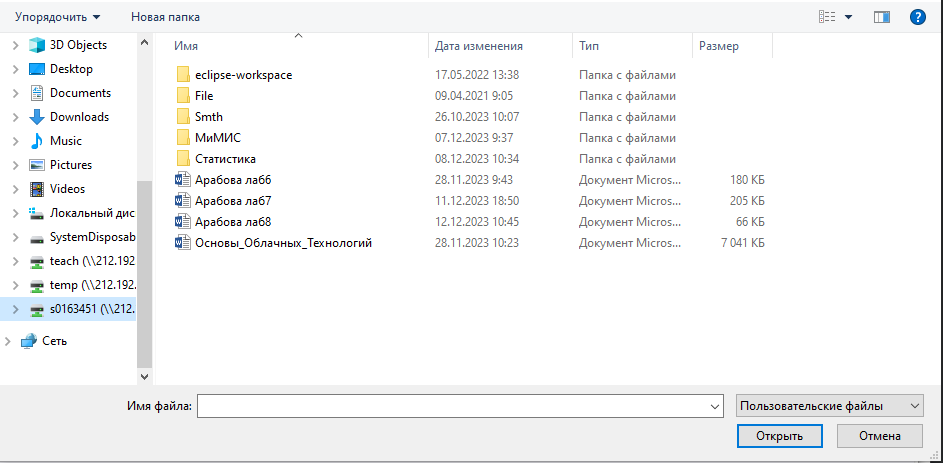


Рисунок – Окно выбора файла, отсеяны все не doc и docx

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 2 – Интерфейс экрана вывода ошибок

2. Нагрузочное тестирование

Выполним нагрузочное тестирование программы и оценим эффективность автоматизированных функций ПС, запросов к БД. В качестве критерия эффективности используем время выполнения функции. Для тестирования подготовим 5 массивов исходных данных на 10, 50, 100, 500 и 1000. Тестирование проводилось на таблице клиентов. Результаты тестирования представлены в таблице 1.

Технические характеристики используемой машины:

• Оперативная память: 16 Гб

• Процессор: 4 ядра с тактовой частотой 2.333 ГГц, 8 потоков

• Разрядность Windows: 64-разрядная операционная система

• Размер жёсткого диска: 512 Гб

|  |  |  |  |
| --- | --- | --- | --- |
| № | Кол-во записей | Операция | Результат (сек.) |
| 1 | 10 | Вставка | 0,2445 |
| 2 | 50 | Вставка | 0,2636 |
| 3 | 100 | Вставка | 0,3134 |
| 4 | 500 | Вставка | 0,4321 |
| 5 | 1000 | Вставка | 0,7611 |

Таблица 1 – Сравнительная таблица нагрузочного тестирования

При данном тестировании в базу данных загружается сначала 10 записей об объектах, далее 50, 100, 500 и 1000. Далее строится проект, тем самым проверяется скорость работы ПС с различными объёмами данных

После проведения теста мы получили следующие данные:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количество исходных данных | 10 | 50 | 100 | 500 | 1000 |
| Время вычислений | 1870 мс | 1980 мс | 5093 мс | 5516 мс | 9452 мс |

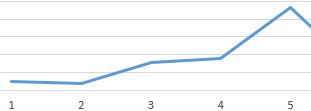


Рисунок 6 – График работы программы

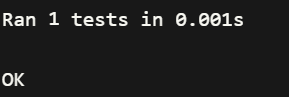
3. Стрессовое тестирование

В этом виде тестирования необходимо проверить работу системы при чрезмерной нагрузке на неё. Стрессовое тестирование ПС будет проведено посредством отправки многократных запросов к БД PostgreSQL. Во время тестирование было обнаружено, что даже в стрессовых ситуациях приложение продолжает функционировать, БД быстро обрабатывает запросы. Могут возникать небольшие задержки в работе интерфейса, но после выполнения череды запросов задержки перестают себя проявлять.

4. Модульное тестирование

Для тестирования работы вставки был написан следующий тест:

def select\_file():  
 file1\_path = filedialog.askopenfilename(initialdir="/", title="Select file",  
 filetypes=(("text files", "\*.docx"), ("all files", "\*.\*")))  
 file1\_entry.delete(0, tk.END)  
 file1\_entry.insert(0, file1\_path)  
  
 file2\_path = filedialog.askopenfilename(initialdir="/", title="Select file",  
 filetypes=(("text files", "\*.docx"), ("all files", "\*.\*")))  
  
 file2\_entry.delete(0, tk.END)  
 file2\_entry.insert(0, file2\_path)



**Раздел 9**

**Баг-репорт №1**

|  |  |
| --- | --- |
| Короткое описание (Summary) | Приложение позволяет регистрироваться пользователям с одинаковой почтой |
| Проект (Project) | Сервис проверки корректности структуры документа |
| Компонент приложения (Component) | main |
| Номер версии (Version) | 1.00 |
| Серьезность  (Severity) | S3 Значительный (Major) |
| Приоритет (Priority) | P2 Средний (Medium) |
| Статус (Status) | Новый |
| Автор (Author) | Арабова Дея Арториксовна |
| Назначен на (Assigned To) | Арабова Дея Арториксовна |
| Окружение | Windows 10 |
| Шаги воспроизведения (steps to Reproduce) | - перейти на стартовую страницу;  - нажать на кнопку «Выбрать файл»;  - выбрать файл с расширением txt; |
| Фактический результат (Result) | программа успешно выбрала файл |
| Ожидаемый результат (Expected Result) | программа не должен высвечиваться файл с расширением txt |
| Прикрепленный файл (Attachment) |  |

Вывод: в рамках данной работы были приобретены навыки выявления дефектов (багов), определения их характеристик и построение баг-репортов.

Руководство пользователя

1. Назначение приложения:

- Название приложения: Проверка корректности структуры документа

- Версия приложения: [указать версию]

- Класс ПС, для которых предназначено приложение: [указать класс ПС]

- Степень универсальности приложения для решения задач: [указать степень универсальности]

- Характеристики пользовательского интерфейса: [описать пользовательский интерфейс]

- Уровень подготовки пользователей: [указать уровень подготовки]

2. Установка приложения:

Скачайте приложение с GitHub: https://github.com/s0163451/main. Распакуйте скачанный архив и установите необходимые зависимости, такие как Python v2.7 и позднее, numpy v1.5 и позднее, scipy v0.13.x и позднее, matplotlib v1.3 и позднее и tkinter v8.5 и позднее.

3. Концепция приложения:

Концепция приложения "Проверка корректности структуры документа" заключается в предоставлении пользователю возможности проверить структуру документа на соответствие определенным критериям или правилам. Приложение позволяет автоматизировать процесс проверки, обнаруживая потенциальные ошибки или несоответствия в структуре документа.

Основные аспекты концепции приложения:

* Типы документов: Приложение может работать с различными типами документов, такими как текстовые документы, электронные таблицы, презентации и другие форматы файлов.
* Правила проверки: Пользователь может настраивать правила проверки, определяющие ожидаемую структуру документа. Правила могут включать проверку наличия определенных разделов или элементов, форматирования текста, наличия ссылок и других характеристик документа.
* Визуализация результатов: Приложение предоставляет визуальную обратную связь о результатах проверки. Ошибки или несоответствия в структуре документа могут быть выделены цветом, подсвечены или отображены в виде списка, чтобы пользователь мог легко идентифицировать проблемные места.
* Исправление ошибок: Пользователь может выбрать опцию исправления обнаруженных ошибок. Приложение может предложить автоматические исправления или предоставить рекомендации по внесению изменений в документ.
* Экспорт результатов: Результаты проверки могут быть экспортированы в различные форматы, такие как отчеты в текстовом формате, таблицы Excel или PDF-файлы. Это позволяет пользователям сохранять и передавать результаты проверки другим пользователям или системам.

Концепция приложения стремится упростить и ускорить процесс проверки структуры документа, помогая пользователям обнаружить и исправить потенциальные ошибки или несоответствия. Оно может быть полезно для профессионалов, работающих с различными типами документов, таких как письма, отчеты, презентации и другие, где точность и соответствие структуре играют важную роль.

4. Функциональный состав приложения:

Функциональный состав приложения "Проверка корректности структуры документа" включает следующие основные функции:

1. Загрузка документа:

- Пользователь может загрузить документ для проверки из локального хранилища или указать путь к файлу.

- Поддерживаются различные форматы документов, такие как текстовые файлы (например, .txt, .docx), электронные таблицы (например, .xlsx) и презентации (например, .pptx).

2. Настройка правил проверки:

- Пользователь может настроить правила проверки, определяющие ожидаемую структуру документа.

- Правила могут включать проверку наличия определенных разделов или элементов, форматирования текста, наличия ссылок и других характеристик документа.

- Пользователь может добавлять, редактировать и удалять правила проверки в соответствии с требованиями документа.

3. Проверка структуры документа:

- Приложение анализирует загруженный документ и сравнивает его с настроенными правилами проверки.

- Проверка включает анализ разделов, подразделов, заголовков, списков, форматирования текста и других элементов структуры документа.

- Приложение выявляет ошибки или несоответствия в структуре документа и генерирует соответствующий отчет.

4. Визуализация результатов:

- Результаты проверки отображаются в пользовательском интерфейсе приложения.

- Ошибки, несоответствия или потенциально проблемные места в структуре документа могут быть выделены цветом, подсвечены или отображены в виде списка.

- Это позволяет пользователю легко идентифицировать проблемные области и сконцентрироваться на их исправлении.

5. Исправление ошибок:

- Приложение предлагает пользователю опции исправления обнаруженных ошибок.

- Это может включать автоматические исправления, предложения по исправлению или рекомендации по внесению изменений в документ.

- Пользователь может выбрать опцию исправления или внести изменения вручную.

6. Экспорт результатов:

- Результаты проверки могут быть экспортированы в различные форматы, такие как отчеты в текстовом формате, таблицы Excel или PDF-файлы.

- Это позволяет пользователям сохранять и передавать результаты проверки другим пользователям или системам.

7. Управление настройками:

- Пользователь может настроить параметры приложения, такие как язык интерфейса, внешний вид, предпочтения проверки и другие параметры.

- Приложение сохраняет настройки пользователя для последующего использования.

Это основной функциональный состав приложения "Проверка корректности структуры документа". Он предоставляет средства для загрузки, анализа и проверки структуры документа, а также возможности визуализации результатов, исправления ошибок и экспорта отчетов.