

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Web-дизайн и мультимедиа
(направленность (профиль) / специализация)

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)**

на тему «Фронтенд разработка компонентов формирования и генерации формуляров документов»

Обучающийся

А.Н. Шукурбеков
(Инициалы Фамилия)

(личная подпись)

Руководитель

канд.пед.наук, доцент, Т.А. Агошкова

(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант

канд.филол.наук, доцент, М.М. Бажутина

(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема бакалаврской работы – «Фронтенд разработка компонентов формирования и генерации формуляров документов».

Объектом исследования бакалаврской работы является фронтенд разработка веб-приложений.

Предметом исследования бакалаврской работы являются компоненты формирования формуляров и генерации документов.

Цель выпускной квалификационной работы – фронтенд-разработка компонентов формирования формуляров и генерации документов.

Методы исследования – технологии фронтенд разработки веб-приложений, методы и технологии веб-дизайна.

Практическая значимость бакалаврской работы заключается в разработке компонентов формирования и генерации формуляров документов, которые могут быть использованы при реализации соответствующих функций веб-приложений.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы.

Результаты бакалаврской работы представляют научно-практический интерес и могут быть рекомендованы программистам, занимающимся фронтенд-разработкой компонентов для формирования и генерации формуляров документов.

Бакалаврская работа состоит из 41 страницы текста, 19 рисунков, 6 таблиц и 25 источников.

Abstract

The title of the bachelor's thesis is "Front-end development of components for the formation and generation of document forms."

The object of study of the bachelor's thesis is the front-end development of web applications.

The subject of the study of the bachelor's work is the components of the formation of forms and the generation of documents.

The purpose of the final qualification work is the front-end development of the components for the formation of forms and the generation of documents.

Research methods - front-end web application development technologies, web design methods and technologies.

The practical significance of the bachelor's work lies in the development of components for the formation and generation of forms of documents that can be used in the implementation of the relevant functions of web applications.

This work consists of an introduction, three chapters, a conclusion and a list of references.

The results of the bachelor's work are of scientific and practical interest and can be recommended to programmers involved in front-end development of components for the formation and generation of document forms.

The work consists of an introduction, 3 sections, a conclusion, and a list of 25 references. The total volume of the work, without its appendices, is 41 pages of a typewritten text, including 6 tables, 19 figures.

Оглавление

Введение.....	5
Глава 1 Постановка задачи на фронтенд разработку компонентов формирования и генерации формуляров документов	7
1.1 Анализ процесса формирования и генерации формуляра-шаблона документа.....	7
1.2 Разработка требований к клиентской части веб-приложения для формирования и генерации формуляров документов.....	9
1.3 Обзор и анализ программ-клиентов для формирования и генерации формуляров документов	11
Глава 2 Проектирование программы-клиента для формирования и генерации формуляров документов	16
2.1 Логическое проектирование программы-клиента.....	16
2.2 Выбор технологии для фронтенд-разработки.....	22
2.3 Разработка архитектуры программы-клиента.....	26
Глава 3 Реализация и тестирование фронтенд-разработки компонентов для формирования формуляров и генерации документов	28
3.1 Разработка программы-клиента для формирования формуляров и генерации учетных документов.....	28
3.2 Тестирование прототипа программы-клиента.....	32
Заключение	37
Список используемой литературы	39

Введение

Формирование и печать документов являются обязательными функциями современного программного обеспечения.

Следует отметить, что в программах, обеспечивающих информационную поддержку деятельности, связанной с формированием регламентированной отчетности или обработкой строго структурированной документации, одной из ключевых задач является генерация формуляров документов.

Согласно ГОСТ Р 51141–98 формуляр документа – это «модель построения документа, устанавливающая область применения, форматы, размеры полей, требования к построению конструкционной сетки и основные реквизиты» [4].

Как показывает практика разработки веб-приложений, в которых предусмотрена функция формирования и генерации интерактивных формуляров документов, программная реализация данной функции осуществляется на стадии разработки фронтенда веб-приложения.

Фронтенд разработка компонентов формирования и генерации формуляров документов представляет актуальность и научно-практический интерес.

Объектом исследования бакалаврской работы является фронтенд разработка веб-приложений.

Предметом исследования бакалаврской работы являются компоненты формирования формуляров и генерации документов.

Цель выпускной квалификационной работы – фронтенд-разработка компонентов формирования формуляров и генерации документов.

Для достижения данной цели необходимо выполнить следующие задачи:

- проанализировать и выбрать технологии фронтенд разработки компонентов формирования и генерации формуляров документов;

- разработать компоненты формирования и генерации формуляров документов;

- выполнить реализацию и тестирование компонентов формирования и генерации формуляров документов.

Методы исследования – технологии фронтенд разработки веб-приложений, методы и технологии веб-дизайна.

Практическая значимость бакалаврской работы заключается в разработке компонентов формирования и генерации формуляров документов, которые могут быть использованы при реализации соответствующих функций веб-приложений.

Данная работа состоит из введения, трех глав, заключения, списка используемой литературы.

Первая глава посвящена анализу и выбору технологий фронтенд разработки компонентов формирования и генерации формуляров документов.

Вторая глава разработке компонентов формирования и генерации формуляров документов.

В третьей главе представлены реализация и тестирование компонентов формирования и генерации формуляров документов.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 41 страницы текста, 19 рисунков, 6 таблиц и 25 источников.

Глава 1 Постановка задачи на фронтенд разработку компонентов формирования и генерации формуляров документов

1.1 Анализ процесса формирования и генерации формуляра-шаблона документа

Формуляр документа — это образец документа, определяющий состав и расположение его реквизитов.

Формуляр-шаблон документа представляет собой схему расположения на стандартном листе реквизитов, которые нужны для подтверждения юридической значимости документа в данной системе.

Процесс формирования и генерации формуляра-шаблона документа в веб-приложении представлен на рисунке 1.

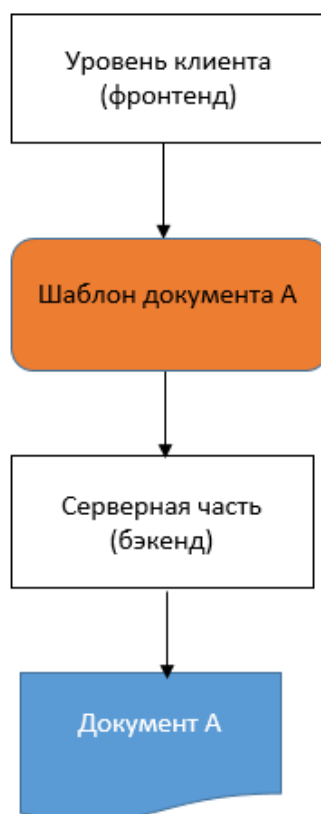


Рисунок 1 - Процесс формирования и генерации формуляра-шаблона документа

Фронтенд — это уровень представления или та часть веб-приложения, которую видит пользователь. Например, графический интерфейс пользователя (GUI) [22].

Фронтенд состоит из частей, представленных на рисунке 2 [24].

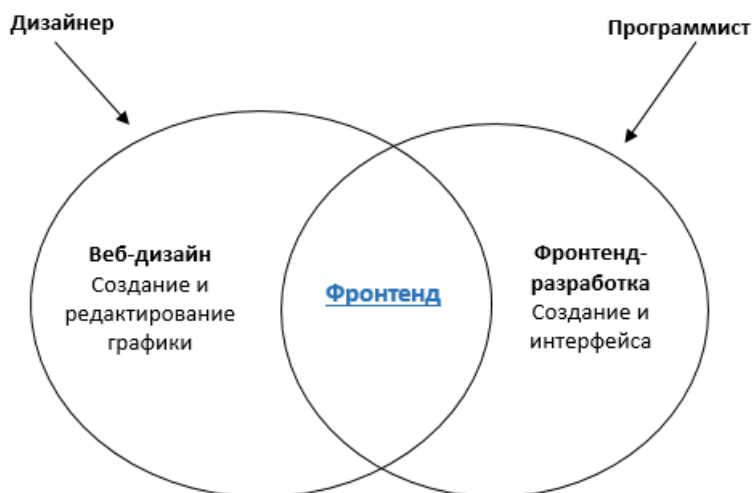


Рисунок 2 – Структура фронтенда

Благодаря такому сочетанию кодирования и проектирования у разработчика есть широкий спектр возможностей для индивидуального дизайна веб-приложения.

Фронтенд закрывает разрыв между интерфейсом и действиями, которые выполняются в фоновом режиме.

Это делает возможным взаимодействие пользователя с серверной частью, которое в противном случае было бы затруднено или требовало высокого уровня специализированных ноу-хау.

Фронтенд-разработчик отвечает за реализацию визуальных элементов, которые пользователи видят и с которыми взаимодействуют в веб-приложении. Обычно они поддерживаются внутренними веб-разработчиками, которые отвечают за логику серверных приложений и интеграцию работы, выполняемой внешними разработчиками [17].

Фронтенд-разработчики создают следующие элементы [16]:

- кнопки;
- макеты;
- навигация;
- картинки;
- графика;
- анимации;
- организация контента.

Таким образом, фронтенд разработка – это процесс разработки клиентской части веб-приложения.

1.2 Разработка требований к клиентской части веб-приложения для формирования и генерации формуляров документов

Для разработки требований к клиентской части веб-приложения для формирования и генерации формуляров документов использована методология FURPS+.

«FURPS+ - это метод проверки приоритетных требований к ПО после понимания потребностей клиента.

Аббревиатура FURPS означает «функциональность, удобство использования, надежность, производительность и возможность поддержки».

В методологии FURPS+ требования к ПО разделены на функциональные и нефункциональные требования.

Функциональные требования объясняют, что должно быть сделано путем определения необходимой задачи, действия или деятельности, которые должны быть выполнены.

Функциональный анализ требований будет использоваться в качестве функций верхнего уровня.

Нефункциональные требования – это требования, которые определяют критерии, которые могут использоваться для оценки работы системы, а не

конкретного поведения» [23].

Нефункциональные требования к качеству ПО – это удобство использования, надежность, производительность и возможность поддержки.

В таблице 1 представлены основные требования к клиентской части веб-приложения с учетом особенностей методологий FURPS+.

Таблица 1– Требования к клиентской части веб-приложения

Требование	Статус	Полезность	Риск	Стабильность
Создание формуляров документов	Одобренное	Критическая	Средний	Низкая
Генерация формуляров документов	Одобренное	Критическая	Средний	Низкая
Современный веб-дизайн	Одобренное	Критическая	Средний	Низкая
Отсутствие функциональной избыточности	Одобренное	Критическая	Средний	Низкая
Допустимое время простоя – до 40 часов в год, но не более 8 часов в месяц	Одобренное	Важная	Средний	Средняя
Среднее время сбоев: 1 раб. день	Одобренное	Важная	Средний	Средняя
Возможность восстановления системы после сбоев: 1 раб. день	Одобренное	Важная	Средний	Средняя
Режим работы: 24x5	Одобренное	Важная	Средний	Средняя
Допустимое количество одновременно работающих пользователей: 10	Предложенное	Важная	Средний	Средняя
Время реакции на возникновение аварийной ситуации: 10 с	Предложенное	Важная	Средний	Средняя
Время устранения критических проблем: в течение рабочего дня	Предложенное	Важная	Средний	Средняя
Фронтенд-разработка на CMS	Предложенное	Критическая	Средний	Низкая
Простота адаптации	Предложенное	Критическая	Средний	Низкая

Разработанный перечень требований является основой для проектирования клиентской части веб-приложения для формирования и

генерации формуляров документов.

1.3 Обзор и анализ программ-клиентов для формирования и генерации формуляров документов

1.3.1 Программа «МойСклад»

Простая программа, которая подходит для оптовой и розничной торговли, интернет-магазина, производства или строительства — для любого бизнеса, где есть товары, сырье или материалы [8].

На рисунке 3 показан фрагмент страницы форм документов.

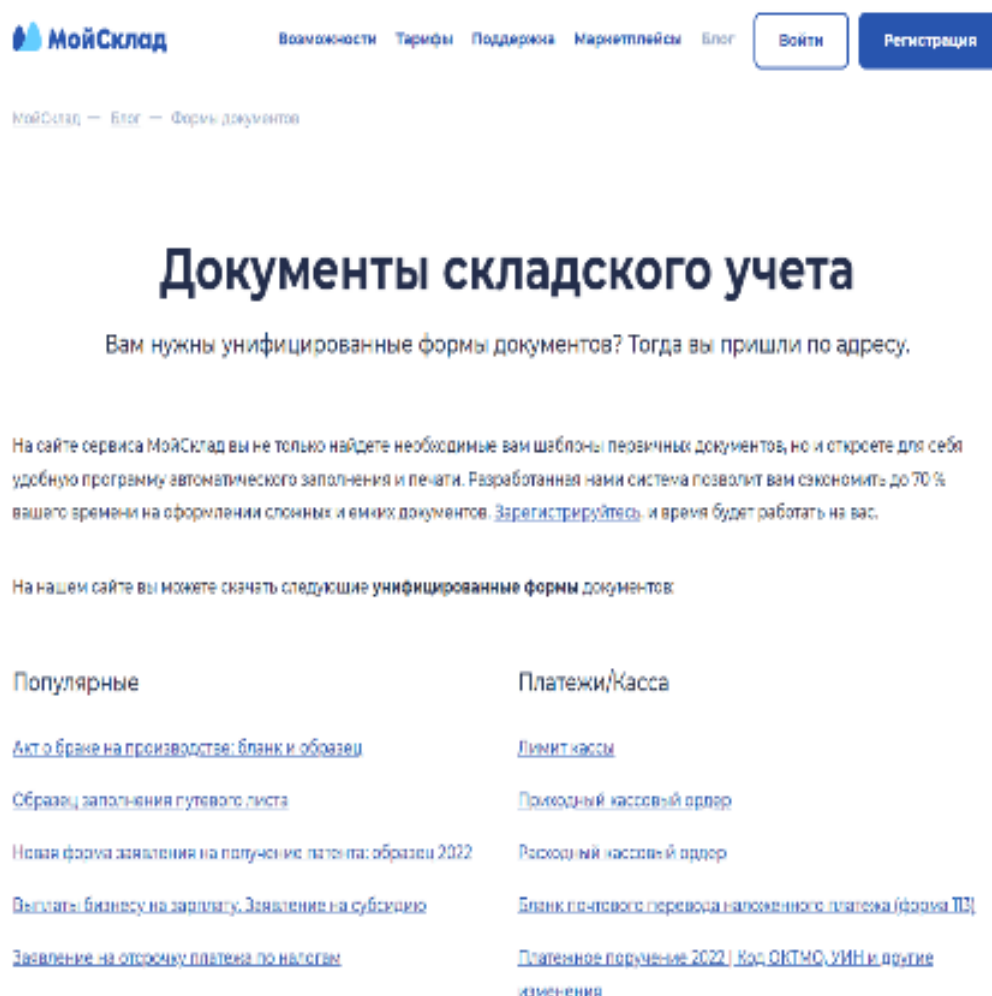


Рисунок 3 – Страница «Документы складского учета» программы «МойСклад»

На уровне клиента программа позволяет добавлять поля в справочники и документы, загружать свои шаблоны печатных форм и отчетов и создавать каталоги.

Основные функции:

- приемка, отгрузка, инвентаризация;
- работа с размерами, цветами, упаковками, серийными номерами, весовым товаром;
- печать документов и аналитика.

Программа разработана на платформе CMS 1С: Битрикс.

К недостаткам программы в контексте рассматриваемой задачи можно отнести избыточную функциональность.

1.3.2 Программа CloudShop

Программа для учёта товара в магазине «CloudShop» предназначена для оптимизации продаж розничного магазина и позволяет вести складской и финансовый учёт, контролировать персонал, анализировать данные, настраивать скидки и подключать онлайн-кассы [9] (рисунок 4).

Документ DocNumber от DocDate

Магазин: DocStore

Покупатель: DocCustomerName

Поз.	Товар	Цена	Кол-во	Сумма со скидкой
1	Товар №1	118	1	118
2	Товар №2	118	2	236
Итого:			☒ 3	☒ 354

DocDate

Товар получил:

Рисунок 4 – Шаблон документа программы CloudShop

В программе имеется возможность настройки шаблонов документов.

Программа разработана без использования CMS.

К недостаткам программы в контексте рассматриваемой задачи можно отнести невозможность доработки под специфику задач конкретного пользователя.

1.3.3 Единый портал «Электронный бюджет»

В справочнике терминов портала формуляр рассматривается как «метаописание структуры данных, алгоритмов контроля данных и визуального представления одного информационного объекта предметной области (электронный документ, справочник, реестр, отчет)» [5].

Соответственно документ рассматривается как экземпляр формуляра.

Визуальная форма экземпляра формуляра «Шаблон листа согласования» показана на рисунке 5.

Рисунок 5 – Визуальная форма экземпляра формуляра «Шаблон листа согласования»

Для сравнения аналогов фронтенда веб-приложений складского учета используем таблицу 2.

Таблица 2 – Описание полей формуляра «Шаблон листа согласования»

Наименование поля	Описание поля
Блок реквизитов в табличной части «Общая информация»	
Дата начала действия	Выбор из календаря или автоматическое заполнение текущей датой.
Дата окончания действия	Выбор из календаря
Статус записи	Поле будет заполнено при сохранении
Блок реквизитов в табличной части «Данные об организации»	
Тип документа	Выбор значения из справочника.
Код ТОФК	Выбор значения из справочника.
Наименование ТОФК	Заполнится автоматически после выбора из справочника Кода ТОФК
Блок реквизитов в табличной части «Утверждающий»	
Должность утверждающего	Заполнится автоматически.
ФИО утверждающего	Заполнится автоматически.
Дата направления	Выбор из календаря
Дата утверждения	Заполнится автоматически
Логин пользователя	Выбор значения из справочника.
Примечание	Ввод вручную.
Блок реквизитов в табличной части «Согласование»	
Логин согласующего	Выбор значения из справочника.
ФИО согласующего	Заполнится автоматически
Должность согласующего	Заполнится автоматически
Порядок согласования	Ввод вручную.
Решение согласующего	Заполняется автоматически.
Примечание	Ввод вручную.
Дата поступления на согласование	Заполняется автоматически.
Дата согласования	Заполняется автоматически.
Структура подразделения согласующего	Ввод вручную.
Статус согласования	Ввод вручную.

Для разработки сайта не использована какая-либо CMS.

К недостаткам программы в контексте рассматриваемой задачи можно отнести невозможность доработки под специфику задач конкретного пользователя.

Для сравнения аналогов клиентской части веб-приложения составлена таблицу 3.

Таблица 3 – Сравнительный анализ аналогов клиентской части веб-приложения формирования формуляров и генерации документов

Характеристика/балл (0-3)	МойСклад	CloudShop	Единый портал «Электронный бюджет»
Создание и генерация формуляров документов	3	3	3
Современный веб-дизайн	3	3	3
Функциональная избыточность	1	1	1
Фронтенд-разработка на CMS	3	3	0
Простота адаптации	2	2	1
Итого	12	12	8

Как показал сравнительный анализ существующих аналогов, ни один из них не соответствует всем установленным требованиям.

Поэтому целесообразно разработать новый фронтенд для программы-клиента формирования формуляров и генерации документов.

Выводы по главе 1

Результаты проделанной работы позволили сделать нижеследующие выводы:

- фронтенд — это уровень представления или та часть веб-приложения, которую видит пользователь;
- как показал сравнительный анализ существующих аналогов клиентской части веб-приложения формирования формуляров и генерации документов, ни один из них не соответствует всем установленным требованиям.

Поэтому целесообразно разработать новый фронтенд для программы-клиента формирования формуляров и генерации документов.

Глава 2 Проектирование программы-клиента для формирования и генерации формуляров документов

2.1 Логическое проектирование программы-клиента

Для разработки программы используем методологию RUP (Rational Unified Process) [20].

Методология RUP – это процесс разработки программного обеспечения для объектно-ориентированных моделей. Она также известна как унифицированная модель процесса.

RUP снижает вероятность непредвиденных обстоятельств, затраты на разработку и предотвращение нерационального использования ресурсов.

RUP предлагает итеративный подход к проектированию и разработке ПО, основанный на спиральном жизненном цикле.

Жизненный цикл ПО в методологии RUP представлен на рисунке 6.

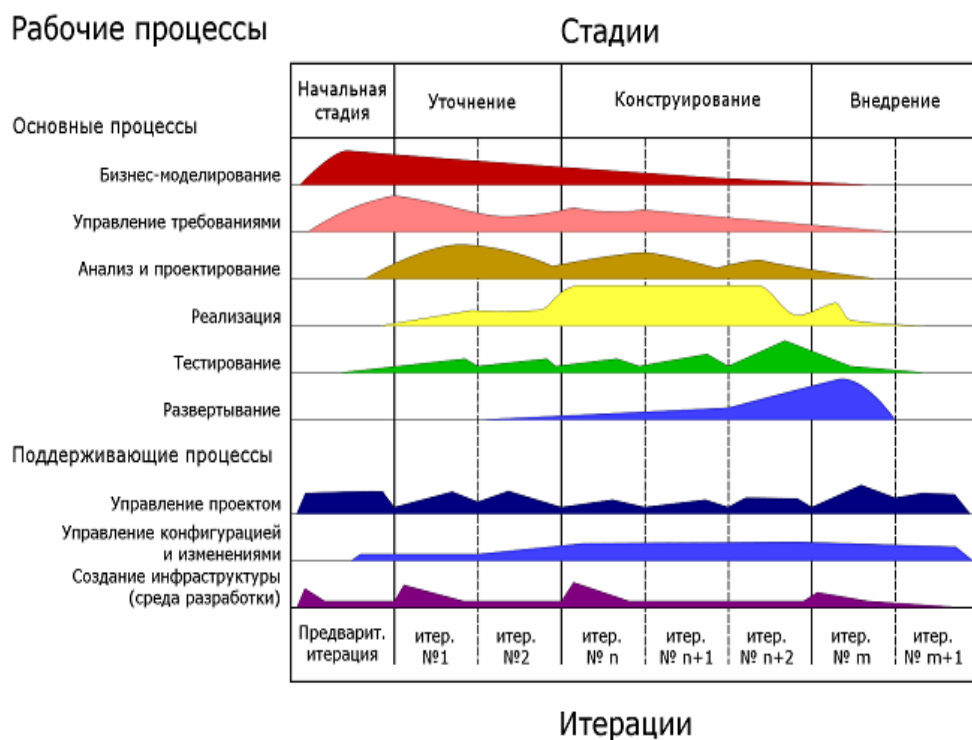


Рисунок 6 – Жизненный цикл ПО в методологии RUP

Жизненный цикл ПО в методологии RUP включает четыре фазы [21]:

- начальная стадия (Inception);
- уточнение (Elaboration);
- конструирование (Construction);
- тестирование (Transition).

Каждая фаза складывается из последовательности итераций, число которых может быть любым.

Рабочий процесс фронтенд разработчика строится на тесном взаимодействии с веб-дизайнером (проектировщиком логической структуры и удобных решений взаимодействия с пользователями) и бэкэнд программистом (разработчиком серверной части проекта). В результате это взаимодействие дает высокий показатель качества программного обеспечения.

Наиболее важным типом артефактов в RUP является логическая модель программы, разработанная на основе диаграмм языка UML и отражающая ее различные аспекты.

Преимущества методологии RUP:

- ограничивает финансовые риски затратами на одну итерацию;
- снижает риск непоставки продукта;
- ускоряет темпы процесса разработки ПО в целом;
- облегчает адаптацию к неизбежным изменениям требований.

В методологии RUP логическая модель программы представляет собой комплекс диаграмм UML, образующих ядро языка UML.

«Диаграммы вариантов использования применяются для выявления функциональных требований к системе» [1].

Диаграмма вариантов использования содержит следующие элементы:

- актер – участник, обладающий поведением или ролью, например, человек, другая система, организация;
- сценарий – определенная последовательность действий и взаимодействий между субъектами и системой, также известная как экземпляр варианта использования;

– вариант использования – набор связанных успешных и неудачных сценариев, описывающие акторов, использующих систему для достижения цели:

– связи между акторами и вариантами использования.

В результате анализа были выделены следующие акторы: Пользователь и Браузер.

Варианты вариантом использования программы-клиента описаны в таблицах 4-5.

Таблица 4 – Создание формуляра документа

Прецедент: Формирование формуляра документа
ID: 1
Краткое описание: Формирование формуляра документа
Главный актер: Пользователь
Второстепенный актер: Браузер
Предусловие: нет
Постусловие: нет
Основной поток: Пользователь формирует документ
Альтернативные потоки: нет

Таблица 5 – Генерация документа

Прецедент: Генерация формуляра документа
ID: 2
Краткое описание: Генерация формуляра документа
Главный актер: Пользователь
Второстепенный актер: Браузер
Предусловие: Создание формуляра
Постусловие: нет
Основной поток: Пользователь выполняет генерацию документа средствами фронтенда
Альтернативные потоки: нет

Разработанная диаграмма вариантов использования представлена на рисунке 7.

«Для отражения статического и элементного аспекта программы-клиента разработана диаграмма классов UML» [1]. Диаграмма классов

программы-клиента представлена на рисунке 8.

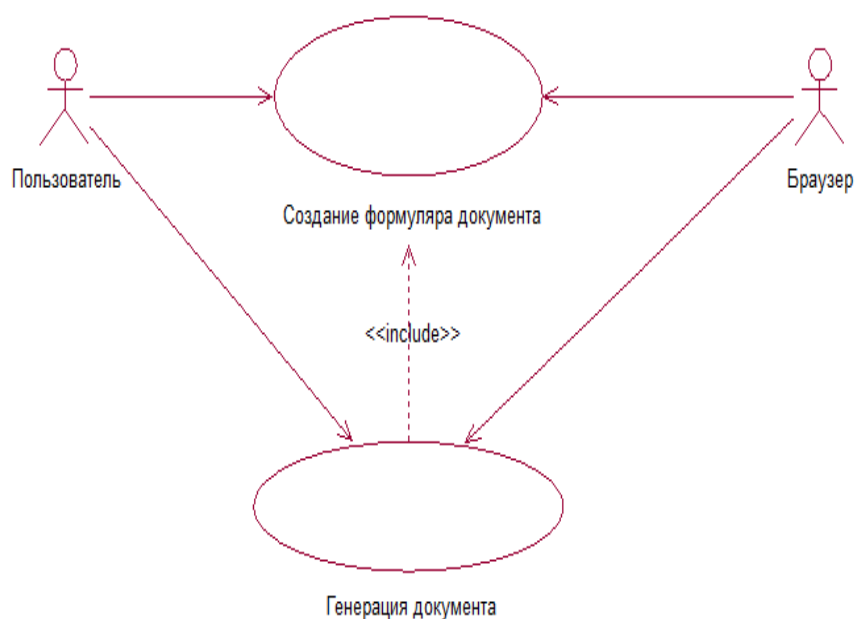


Рисунок 7 – Диаграмма вариантов использования программы-клиента для формирования формуляров и генерации учетных документов

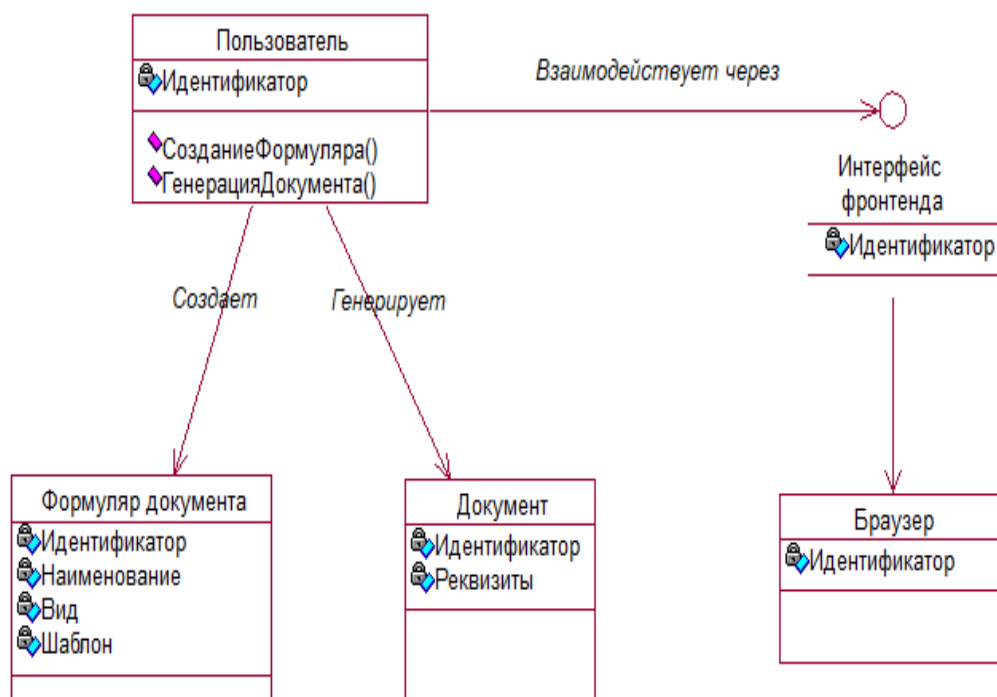


Рисунок 8 – Диаграмма классов программы-клиента для формирования формуляров и генерации учетных документов

Диаграммы классов используются как на этапах анализа, так и на этапах проектирования. На этапе анализа создается концептуальный проект очень высокого уровня. В это время может быть создана диаграмма классов, в которой показаны только имена классов, или, возможно, могут быть добавлены некоторые фразы, подобные псевдокоду, для описания обязанностей класса.

В таблице 6 представлена спецификация классов программы-клиента.

Таблица 6 – Спецификация классов программы-клиента

Класс	Описание
«Пользователь	Класс объектов, моделирующих на логическом уровне пользователей веб-приложения
Браузер	Класс объектов, моделирующих на логическом уровне браузеры, в которых функционирует фронтенд
Формуляр	Класс объектов, моделирующих на логическом уровне формуляры документов
Интерфейс	Класс объектов, моделирующих на логическом уровне интерфейс фронтенда» [6]

«Для отражения динамического аспекта информационной системы используется диаграмма последовательности UML.

Диаграммы последовательности используются на этапах анализа и проектирования» [1].

Диаграммы последовательности часто используются для отображения хронологически структурированного потока событий в прецеденте. При создании диаграммы последовательности идентифицируются объекты, участвующие в варианте использования.

Процесс создания диаграммы последовательности часто приводит к уточнению варианта использования, потенциально определяя недостающее, но желаемое поведение.

Диаграммы последовательности представляют собой поведение системы, основанное на необходимых взаимодействиях между набором

объектов с точки зрения сообщений, которыми они обмениваются для получения желаемого результата.

Диаграммы последовательности показывают последовательность сообщений во времени [18].

На рисунке 9 изображена диаграмма последовательности сценария создания формуляра и генерации документа [2].



Рисунок 9 – Диаграмма последовательности сценария создания формуляра и генерации документа

Сценарий создания формуляра и генерации документа организован следующим образом:

Объект Пользователь через интерфейс активизирует опцию создания формуляра документа.

Объект Браузер открывает страницу создания формуляра и сообщает об это объекту Пользователь.

Объект Пользователь создает формуляр документа.

Объект Пользователь через интерфейс активизирует опцию генерации документа.

Объект Браузер открывает страницу генерации документа и сообщает об это объекту Пользователь.

Объект Пользователь выполняет генерацию документа.

Процесс формирования и генерации формуляра документа завершен.

2.2 Выбор технологии для фронтенд-разработки

«В качестве платформы для разработки коммерческих сайтов используются CMS – системы управления контентом. Это программное обеспечение, которое помогает пользователям создавать и изменять контент на веб-сайте, а также управлять им без специальных технических знаний.

Для выбора системы управления контентом рассмотрим характеристики CMS-платформ Wordpress, Drupal и October» [14].

2.2.1 Система управления контентом Wordpress

«Wordpress - это инструмент для создания веб-сайтов с открытым исходным кодом, написанный на PHP в комплекте с СУБД MySQL или MariaDB.

По мнению разработчиков, это, вероятно, самая простая и мощная из существующих сегодня CMS [24].

Возможности включают в себя архитектуру плагинов и систему шаблонов, называемых в WordPress-темами.

CMS WordPress изначально создавалась как система публикации блогов, но развивалась для поддержки других типов веб-контента, включая более традиционные списки рассылки и форумы, медиа-галереи, сайты участников и онлайн-магазины» [13].

На рисунке 10 представлена главная страница CMS WordPress.

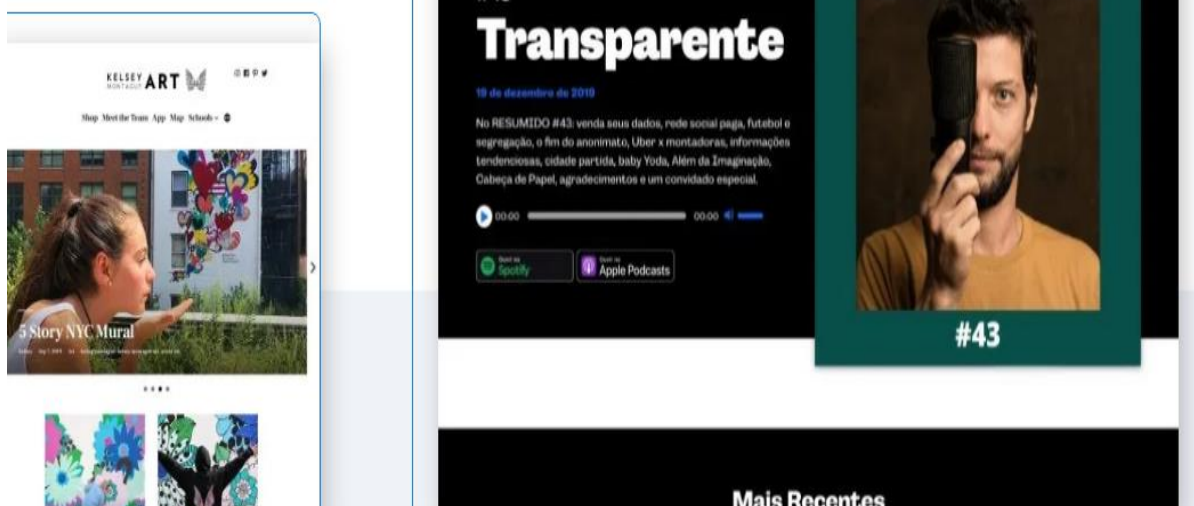


Рисунок 10 – Главная страница сайта wordpress.com

«CMS ориентирована на платформу LAMP (Linux, Apache, MySQL, PHP), но возможны варианты использования другого системного ПО.

Состав программной архитектуры платформы WordPress может изменяться в зависимости от используемой версии данной CMS» [25].

2.2.2 Система управления контентом Drupal

«CMS Drupal – это система управления цифровым интерфейсом для управления веб-контентом и многоканальным персонализированным интерфейсом.

CMS-платформа Drupal - это бесплатное ПО с открытым исходным кодом. Она основан на таких принципах, как сотрудничество, глобализм и инновации. Drupal обладает отличными стандартными функциями, такими как простое создание контента, надежная производительность и отличная безопасность» [11].

На рисунке 11 представлена страница сайта Drupal для веб-

разработчиков.



Рисунок 11 – Страница сайта Drupal для веб-разработчиков

По мнению разработчиков платформы, главным ее достоинства является гибкость.

«Модульность является одним из основных принципов Drupal. Встроенный инструментарий помогает создавать универсальный, структурированный контент, который нужен динамическому веб-интерфейсу. В настоящее время разработчикам предлагается версия Drupal 8.x» [11].

2.2.3 CMS October

October CMS — это система управления сайтом, построенная на компонентах Laravel [12].

Переведена на русский, позволяет гибко проектировать и масштабировать административную часть, без наследования проблем скорости ядра или избыточного, но не нужного проекту функционала.

Мощный фреймворк на php позволит решать сложные функциональные задачи без значительных ограничений.

Использование October CMS даст базовую административную часть и возможность мелких расширений как плагинами из маркетплейса, так и собственными встраиваемыми страницами (рисунок 12).

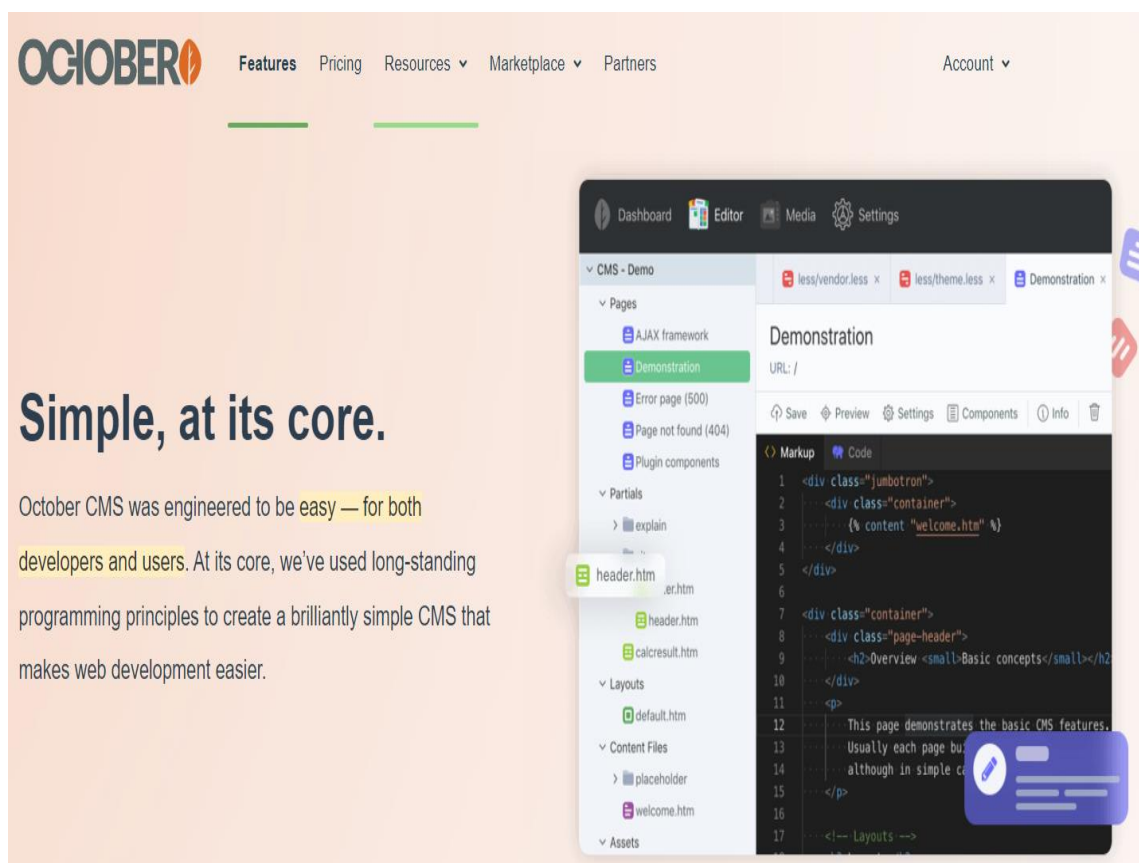


Рисунок 12– Главная страница сайта octobercms.com

Разработчики и агентства любят эту CMS за простоту, как и их клиенты. Тысячи разработчиков и цифровых студий с такими клиентами, как Toyota, KFC и Nestle, уже используют October CMS в качестве своей основной платформы.

Для сравнения характеристик платформ составлена таблица 7.

Таблица 7 – Сравнительный анализ CMS WordPress, Drupal и October

Характеристика/ балл (0-3)	WordPress	Drupal	October
Требования к квалификации разработчика	3	2	2
Применение шаблонов тем и плагинов	3	2	2
Масштабируемость	3	3	3
Предпочтение разработчика	3	1	1
Итого:	12	8	8

Таким образом, на основании сравнительного анализа в качестве платформы для фронтенд-разработки используем CMS WordPress» [14].

2.3 Разработка архитектуры программы-клиента

Для разработки архитектуры программы-клиента для формирования формуляров и генерации учетных документов используем диаграмму пакетов UML.

Если система большая, ее следует разделить на более мелкие подсистемы, каждая из которых имеет свою собственную диаграмму классов.

В нотации UML разделы/подсистемы называются пакетами.

Пакет представляет собой группу элементов модели и, как таковую, является конструкцией UML, используемой также в других диаграммах UML.

Сами пакеты могут быть вложены в другие пакеты.

Пакет может содержать как подчиненные пакеты, так и обычные элементы диаграммы классов, хотя обычно не рекомендуется смешивать в одной и той же диаграмме пакеты и классы.

Пакет зависит от другого пакета, если хотя бы один из его классов зависит от классов последнего пакета.

На рисунке 13 представлена архитектура программы-клиента веб-приложения.

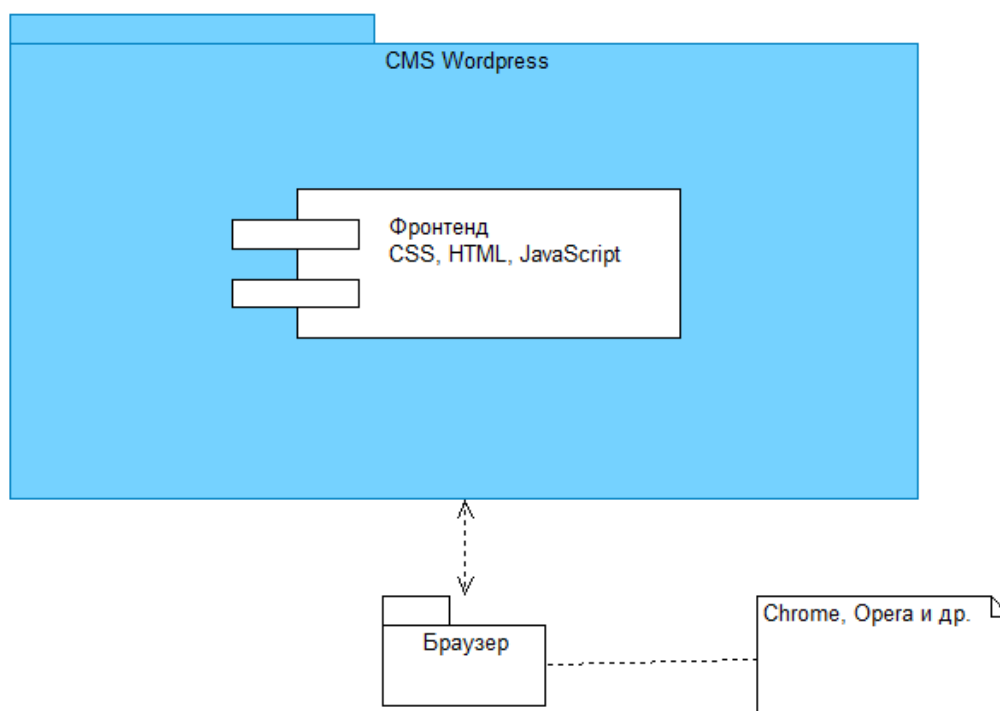


Рисунок 13 – Архитектура программы-клиента

ИТ-инфраструктура компании-заказчика фронтенда должна соответствует требованиям по аппаратно-программному обеспечению программы-клиента для CMS WordPress.

Выводы по главе 2

На основании результатов проделанной работы были сделаны следующие выводы:

- для фронтенд-разработки используются системы управления контентом (CMS-системы);
- в качестве платформы для фронтенд-разработки компонентов программы-клиент выбрана CMS WordPress.

ИТ-инфраструктура компании-заказчика фронтенда должна соответствует требованиям по аппаратно-программному обеспечению программы-клиента для CMS WordPress.

Глава 3 Реализация и тестирование фронтенд-разработки компонентов для формирования формуляров и генерации документов

3.1 Разработка программы-клиента для формирования формуляров и генерации учетных документов

Разработан прототип программы-клиента.

В процессе фронтенд-разработки компонентов программы клиента использованы такие популярные технологии, как HTML, CSS и JavaScript [10, 16].

Также использована утилита ONLYOFFICE Document Builder [19].

ONLYOFFICE Document Builder — это один из самых полных инструментов, который позволяет приложению пользователя изменять или создавать документы в форматах DOCX, XLSX, PPTX или PDF, фактически не запуская Microsoft Office.

В качестве веб-фреймворка использована среда Express-Node.

Express — самая популярная веб-инфраструктура Node.js, а также базовая библиотека для ряда других популярных веб-фреймворков Node [15].

Рассмотрим компоненты прототипа программы-клиента для формирования формуляра и генерации документа «Резюме».

На рисунке 14 показана модель компонента «Шаблон документа».

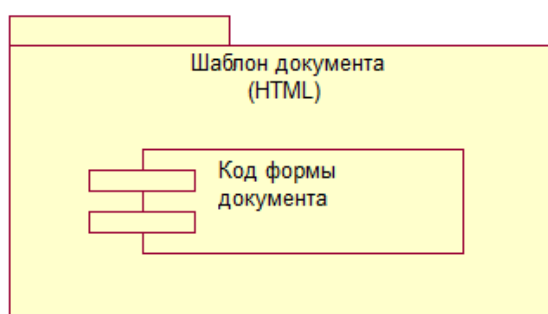


Рисунок 14 – Модель компонента «Шаблон документа»

В листинге 1 представлен HTML-код формы документа «Резюме».

Листинг 1 – HTML-код формы документа

```
« <html>
<div class="fill-name">
  <input type="text" id="fill-name" placeholder="full name">
</div>
<div class="phone-number">
  <input type="number" id="phone-number" placeholder="phone number">
</div>
<div class="email">
  <input type="text" id="email" placeholder="email">
</div>
<div class="profile">
  <textarea id="profile" placeholder="Insert a brief description of yourself"></textarea>
</div>

<div class="education">
  <input type="text" id="degree" placeholder="degree">
  <input type="text" id="university" placeholder="university">
  <input type="text" id="location" placeholder="location">
  <input type="date" id="year" placeholder="year">
</div>
<div class="skills">
  <div class="skill">
    <input type="text" id="new-skill" placeholder="skill"
onkeyup="add_skill_by_enter(event)">
    <button onclick="add_skill()">+</button>
  </div>
</div>
<button onclick="sendForm()">Send</button>
</body>
```

</html>» [3]

На рисунке 15 показана модель компонента генерации формуляра.

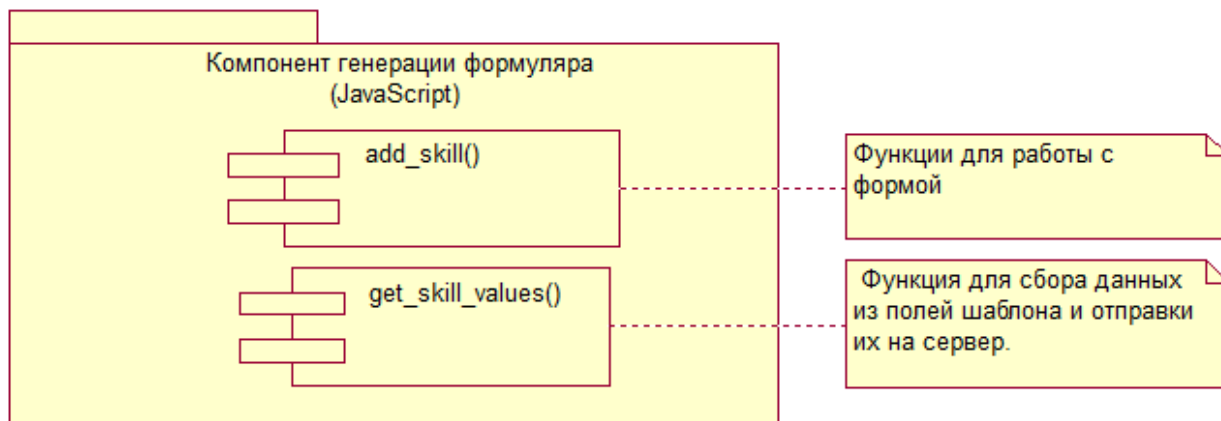


Рисунок 15 – Модель компонента генерации формуляра

В листинге 2 представлен код JavaScript генерации формуляра документа «Резюме».

Листинг 2 – Программный код генерации формуляра документа

```
«<script>
```

```
add_skill = () => {  
    const newSkill = document.getElementById("new-skill");  
    if (newSkill.value === "") {return; } // ничего не делаем, если в поле не введены данные  
    const div = document.createElement("div"); //внешний div  
    const span = document.createElement("span"); // название навыка  
    const button = document.createElement("button"); // кнопка удаления навыка  
    span.innerText += newSkill.value; // добавляем в span введенный текст  
    newSkill.value = ""; // обнуляет поле с названием способности  
    newSkill.focus(); // возвращаем фокус в поле с названием навыка  
    button.innerText += "-";  
    button.onclick = () => { // добавляем действие на кнопку удаления
```

```

    div.remove();
  },»[3]
  «div.appendChild(span); // добавляем span в div
  div.appendChild(button); //добавляем кнопку удаления
  document.getElementsByClassName('skills')[0].appendChild(div); // добавляем объект на
  страницу
  };
  add_skill_by_enter()
  add_skill_by_enter = (event) => {
  if (event.code === "Enter") { // добавляем элемент, только если нажали на кнопку enter
  add_skill();
  }
  };
  get_skill_values = () => {
  const skills = [];
  if (document.getElementById('new-skill').value !== "") {
  skills.push(document.getElementById('new-skill').value);
  }
  Array.from(document.getElementsByClassName('skillfield')).forEach(current_element => {
  skills.push(current_element.innerHTML);
  });
  return skills;
  };
  sendForm()
  sendForm = () => {
  fetch('/', {
  method: 'POST',
  headers: {
  'Content-Type': 'application/json'
  },»[3]
  «body: JSON.stringify({
  userData: {
  fillName: document.getElementById('fill-name').value,

```

```

phoneNumber: document.getElementById('phone-number').value,
email: document.getElementById('email').value,
profile: document.getElementById('profile').value,
education: {
  degree: document.getElementById('degree').value,
  university: document.getElementById('university').value,
  location: document.getElementById('location').value,
  year: document.getElementById('year').value,
},
skills: get_skill_values()
}
})
}).then(res => res.json())
.then(response => {
  location.replace('/' + response.filename); // скачиваем файл, который будет доступен по этой
  ссылке
})
catch(error => console.error('Error:', error));
};
</script>».

```

Рассмотрим процесс тестирования прототипа программы-клиента.

3.2 Тестирование прототипа программы-клиента

Для проверки работоспособности прототипа клиентской части веб-приложения проведено его функциональное тестирование.

Порядок генерации документа:

- пользователь вводит данные в форму на странице в браузере, форма отправляется на сервер;
- Node.js на сервере создает на основе пользовательских данных скрипт для утилиты Document Builder;

- Node.js отдает скрипт утилите Document Builder;
- Document Builder создает с помощью скрипта документ;
- Node.js возвращает пользователю ссылку на документ.

Запускаем express: node main.js.

Открываем в браузере адрес: http: localhost:3000.

Открываем форму и заполняем ее произвольными данными (рисунок 16).

FE-разработчик			
8987			
fe@list.ru			
Россия			
Тольятти			
Привет!			
Бакалавр	ТГУ	Тольятти	2022-05-02
linux	+		
apple.js -			
vintage.js -			
zerg.js -			
css -			
html -			
Send			

Рисунок 16 – Формуляр документа с заполненными данными

Далее создаем скрипт для Document Builder.

За основу берем шаблон резюме, представленный на рисунке 17.

Используя эти данные и шаблон, нужно сделать скрипт, с помощью которого Document Builder создаст документ.

Ваше имя

УЛ. УЛИЦА, Д. 1
ВАШ ГОРОД, СТРАНА, 123456
(+7) (000) 000-00-00
NO_REPLY@EXAMPLE.COM

Навыки

Введите свой текст здесь Введите свой текст здесь Введите свой текст здесь Введите свой текст здесь
Введите свой текст здесь Введите свой текст здесь.

Опыт работы

МЕСЯЦ 20XX - НАСТОЯЩЕЕ ВРЕМЯ

Компания, Город – *должность*

- Введите свой текст здесь Введите свой текст здесь.
- Введите свой текст здесь.
- Введите свой текст здесь Введите свой текст здесь Введите свой текст здесь.

МЕСЯЦ 20XX - МЕСЯЦ 20XX

Компания, Город – *должность*

- Введите свой текст здесь Введите свой текст здесь.
- Введите свой текст здесь.

МЕСЯЦ 20XX - МЕСЯЦ 20XX

Компания, Город – *должность*

- Введите свой текст здесь Введите свой текст здесь Введите свой текст здесь.
- Введите свой текст здесь Введите свой текст здесь.
- Введите свой текст здесь.

Рисунок 17 – Шаблон резюме

Выполняем генерацию скрипта с помощью функции, представленной в модели компонента генерации документа (рисунок 18).

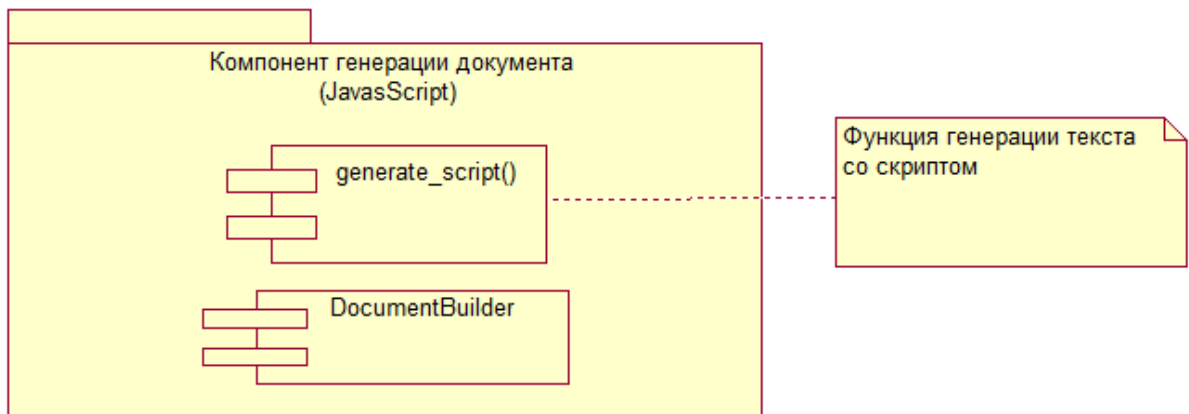


Рисунок 18 – Модель компонента генерации документа

В листинге 3 представлен фрагмент кода JavaScript функции для генерации текста со скриптом.

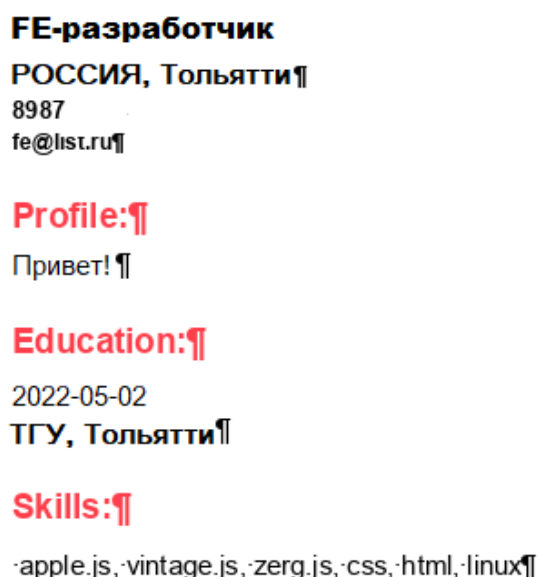
Листинге 3 – Фрагмент кода функции для генерации текста со скриптом

```
«generate_script = (data) => {  
  let first_template = 'builder.CreateFile("docx");\n' +  
    'const Document = Api.GetDocument();\n';  
  first_template += 'const data = ' + JSON.stringify(data) + ';\n';  
  first_template += 'let paragraph = Document.GetElement(0);\n' +  
    'FullName_style = Document.CreateStyle("FullName");\n' + ...  
  return first_template;».
```

Далее записываем скрипт в файл и передать его Document Builder.

Для этого необходимо исполнить команду documentbuilder path/script.js с помощью Node.js.

В результате будет сгенерировано резюме, показанное на рисунке 19.



FE-разработчик
РОССИЯ, Тольятти¶
8987
fe@list.ru¶

Profile:¶
Привет! ¶

Education:¶
2022-05-02
ТГУ, Тольятти¶

Skills:¶
·apple.js, ·vintage.js, ·zerg.js, ·css, ·html, ·linux¶

Рисунок 19 – Сгенерированный документ «Резюме»

Функциональное тестирование прототипа программы-клиента подтвердило, что он позволяет создавать и генерировать формуляры документов.

Выводы по главе 3

Результаты проделанной работы позволили сделать следующие выводы:

- разработан прототип программы-клиента;
- в процессе фронтенд-разработки компонентов программы клиента использованы такие технологии, как HTML, CSS, JavaScript и ONLYOFFICE Document Builder;
- утилита ONLYOFFICE Document Builder позволяет приложению пользователя изменять или создавать документы в форматах DOCX, XLSX, PPTX или PDF;
- в качестве веб-фрейморка использована среда Express-Node;
- выполнена реализация компонентов прототипа программы-клиента для формирования формуляра и генерации документа «Резюме»;
- для проверки работоспособности прототипа клиентской части веб-приложения проведено его функциональное тестирование.

Функциональное тестирование прототипа программы-клиента подтвердило, что он позволяет создавать, настраивать и генерировать формуляры документов.

Заключение

Выпускная квалификационная работа посвящена актуальной проблеме фронтенд-разработки компонентов формирования и генерации формуляров документов.

Как показывает практика разработки веб-приложений, в которых предусмотрена функция формирования и генерации интерактивных формуляров документов, программная реализация данной функции осуществляется на стадии разработки фронтенда веб-приложения.

Фронтенд разработка компонентов формирования и генерации формуляров документов представляет актуальность и научно-практический интерес.

Для достижения поставленной в работе цели решены следующие задачи:

- выполнена постановка задачи на фронтенд-разработку компонентов формирования формуляров и генерации документов. Фронтенд — это уровень представления или та часть приложения, которую видит пользователь. Как показал сравнительный анализ существующих аналогов клиентской части веб-приложения формирования формуляров и генерации документов, ни один из них не соответствует всем установленным требованиям. Поэтому целесообразно разработать новый фронтенд для программы-клиента формирования формуляров и генерации документов.

- спроектирована программа-клиент для формирования формуляров и генерации документов. Для разработки программы-клиента использована методология RUP. Наиболее важным типом артефактов в RUP является логическая модель программы, разработанная на основе диаграмм языка UML и отражающая ее различные аспекты. Логическая модель программы-клиента представляет собой комплекс диаграмм UML, образующих ядро языка UML. Для фронтенд-разработки используются системы управления контентом (CMS-системы). Для выбора системы управления контентом рассмотрены характеристики CMS-платформ Wordpress, Drupal и October. В качестве

платформы для фронтенд-разработки компонентов программы-клиент выбрана CMS WordPress. ИТ-инфраструктура компании-заказчика фронтенда должна соответствует требованиям по аппаратно-программному обеспечению программы-клиента для CMS WordPress.

– разработан прототип программы-клиента. В процессе фронтенд-разработки компонентов программы клиента использованы такие технологии, как HTML, CSS, JavaScript и ONLYOFFICE Document Builder. Утилита ONLYOFFICE Document Builder позволяет приложению пользователя изменять или создавать документы в форматах DOCX, XLSX, PPTX или PDF. В качестве веб-фрейморка использована среда Express-Node. Выполнена реализация компонентов прототипа программы-клиента для формирования формуляра и генерации документа «Резюме». Выполнено функциональное тестирование прототипа программы-клиента, которое подтвердило, что он позволяет создавать, настраивать и генерировать документы.

Результаты бакалаврской работы представляют научно-практический интерес и могут быть рекомендованы программистам, занимающимся фронтенд-разработкой компонентов для формирования и генерации формуляров документов.

Список используемой литературы

1. Бабич А. В. Введение в UML [Электронный ресурс]: учебное пособие. М. : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2022. 198 с. URL: <https://www.iprbookshop.ru/120473.html> (дата обращения: 29.04.2022).
2. Генерация документов [Электронный ресурс]. URL: <https://www.amocrm.ru/support/widgets/generaciya-dokumentov> (дата обращения: 22.04.2022).
3. Генерация документов с помощью ONLYOFFICE DocumentBuilder [Электронный ресурс]. URL: <https://habr.com/ru/post/466217/> (дата обращения: 25.04.2022).
4. ГОСТ Р 51141-98 [Электронный ресурс]. URL: <https://docs.cntd.ru/document/1200003829>.
5. Единый портал «Электронный бюджет» [Электронный ресурс]. URL: <http://budget.gov.ru/> (дата обращения: 25.04.2022).
6. Носова Л. С. Case-технологии и язык UML [Электронный ресурс]: учебно-методическое пособие. Челябинск, Саратов : Южно-Уральский институт управления и экономики, Ай Пи Эр Медиа, 2019. 67 с. URL: <https://www.iprbookshop.ru/81479.html> (дата обращения: 29.04.2022).
7. Онлайн-система учета «Большая Птица» [Электронный ресурс]. URL: <https://bigbird.ru/> (дата обращения: 25.04.2022).
8. Программа «МойСклад» [Электронный ресурс]. URL: <https://www.moysklad.ru/vozmozhnosti/> (дата обращения: 25.04.2022).
9. Программа CloudShop (web-версия) [Электронный ресурс]. URL: <https://cloudshop.ru/portal/article-categories/cloudshop-for-web/> (дата обращения: 25.04.2022).
10. Что такое frontend разработка [Электронный ресурс]. URL: <https://php.zone/post/chto-takoe-frontend-razrabotka> (дата обращения: 22.04.2022).

11. CMS Drupal – официальный сайт [Электронный ресурс]. URL: <https://www.drupal.org/> (дата обращения: 29.04.2022).
12. CMS October [Электронный ресурс]. URL: <https://octobercms.com/> (дата обращения: 29.04.2022).
13. CMS WordPress [Электронный ресурс]. URL: <https://ru.wordpress.org/> (дата обращения: 29.04.2022).
14. CMS для коммерческого сайта. Сравнение популярных систем (дата обращения: 29.04.2022). [Электронный ресурс]. URL: <https://www.uplab.ru/blog/cms-for-a-commercial-site/> (дата обращения: 22.04.2022).
15. Express/Node introduction [Электронный ресурс]. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction (дата обращения: 25.04.2022).
16. Front-End Development: Key Technologies and Concepts [Электронный ресурс]. URL: <https://www.altexsoft.com/blog/front-end-development-technologies-concepts/> (дата обращения: 22.04.2022).
17. Front-End Web Developer Job Description Template [Электронный ресурс]. URL: <https://www.toptal.com/front-end/job-description> (дата обращения: 25.04.2022).
18. Interaction, Collaboration and Sequence Diagrams with Examples [Электронный ресурс]. URL: <https://www.guru99.com/interaction-collaboration-sequence-diagrams-examples.html> (дата обращения: 29.04.2022).
19. ONLYOFFICE [Электронный ресурс]. URL: <https://www.onlyoffice.com/> (дата обращения: 25.04.2022).
20. RUP – методология разработки ПО. [Электронный ресурс]. URL: <https://qaevolution.ru/metodologiya-menedzhment/rup> (дата обращения: 25.04.2022).
21. RUP Lifecycle [Электронный ресурс]. URL: https://swi.cs.vsb.cz/RUPLarge/core.base_rup/customcategories/rup_lifecycle_100BF298.html (дата обращения: 25.04.2022).

22. What is Frontend? What is Backend? [Электронный ресурс]. URL: <https://www.frontend-gmbh.de/en/blog/what-is-frontend-what-is-backend/> (дата обращения: 25.04.2022).

23. What is the use of FURPS+ model in classifying requirements? [Электронный ресурс]. URL: <https://findanyanswer.com/what-is-the-use-of-furps-model-in-classifying-requirements> (дата обращения: 25.04.2022).

24. WordPress Development Best Practices: Workflow & Front End [Электронный ресурс]. URL: <https://www.toptal.com/wordpress/modern-wordpress-development-pt-1> (дата обращения: 25.04.2022).

25. WordPress Foundation [Электронный ресурс]. URL: <https://wordpressfoundation.org/> (дата обращения: 25.04.2022).