Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Технологии разработки программного обеспечения»

**ОТЧЕТ**

к лабораторной работе

на тему:

**«Тестирование программного продукта и экономическое обоснование его реализации»**

|  |  |
| --- | --- |
|  | Выполнил:  Слуцкий Никита Сергеевич,  студент группы 053505 |
|  | Проверил: Гриценко Никита Юрьевич, ассистент каф. Информатики |

Минск 2023

СОДЕРЖАНИЕ

[Введение 3](#_Toc149419812)

[1 Проектирование и моделирование бизнес-процессов с bpmn 4](#_Toc149419813)

[1.1 Модель BPMN 4](#_Toc149419814)

[1.2 Создание модели BPMN 4](#_Toc149419815)

[1.3 Определение ролей 5](#_Toc149419816)

[1.4 Моделирование потоков данных 5](#_Toc149419817)

[1.5 Анализ и оптимизация процесса 5](#_Toc149419818)

[1.6 Сравнение моделирований BPMN и IDEF 6](#_Toc149419819)

[Заключение 7](#_Toc149419820)

[Список используемых источников 8](#_Toc149419821)

# Введение

Целью данной лабораторной работы является:

– разработка плана тестирования, определение тестовых случаев и тестовых данных;

– реализация тестов;

– проведение тестирования, анализа тестирования и документирование найденных ошибок с последующим исправлением их;

– составление экономического обоснования разработки программного продукта.

Также необходимо оформить отчёт о проделанной работе в соответствии со стандартом предприятия БГУИР.

# 1 ПЛан тестирования

План тестирования – это специализированный технический документ, внутри которого описывается весь будущий фронт работ по тестированию разрабатываемого продукта, начиная с документирования самого объекта, использованного в начале стратегии разработки, и критериев нахождения ошибок. План тестирования проиллюстрирован в таблице 1.1.

Таблица 1.1 – План тестирования

|  |  |  |
| --- | --- | --- |
| Тестовый случай | Тестовые данные | Ожидаемый результат |
| Регистрация пользователя | Почта, пароль, никнейм | Пользователь успешно зарегистрирован |
| Вход пользователя в систему | Логин и пароль | Пользователь успешно вошел в систему |
| Загрузка конфигурации по ID | ID конфигурации | Конфигурация загружена и отрисована |
| Отрисовка начального состояния стен на холсте | Массив с элементами PositionedLayoutItem | Стены отрисованы корректно |
| Переход в трёхмерное отображение | Текущее состояние конфигурации | WebGL сцена отрисовалась корректно |
| Внесение изменений в конфигурацию | Новые стены или объекты | Конфигурация сохранена в базе данных |
| Вращение сцены | Координаты мыши на холсте | Корректная смена положения объектов |
| Получение summary по конфигурации | Данные текущей конфигурации | Отображение предварительного заказа с ценами и другими данными |
| Генерация PDF-файла | Данные текущей конфигурации | Скачивание PDF-файла с предварительным предложением клиенту |
| Удаление лэйаута | ID конфигурации | Лэйаут удалён из базы данных |
| Создание новой конфигурации | Ассоциированное имя лэйаута | Конфигурация создана и записана в базу данных |
| Привязка в режиме рисования стен | Координаты мыши | Рисовка может производиться с привязкой к углам или другим стенам |

План тестирования включает определение целей, аудитории, тестовых случаев, приоритетов, рисков, методологии, инструментов, ресурсов и сроков. Затем проводятся тесты, а результаты анализируются с созданием отчета. План тестирования помогает обеспечить качество программного продукта и удостовериться, что оно соответствует ожиданиям пользователей.

Тест план является важной составляющей процесса тестирования. Он содержит в себе всю необходимую информацию, описывающую данный процесс. Иногда он играет формальную роль.

При больших объемах работы и наличии команды тестировщиков сложно представить себе, как осуществлять тестирование без этого документа.

В повседневной жизни на проекте может быть один мастер тест план и несколько детальных тест планов, описывающих отдельные модули одного приложения.

# 2 Реализация тестов

## 2.1 Тесты на приём и обработку входных данных

Тесты на прием и обработку входных данных являются одним из видов модульного тестирования, предназначенного для проверки правильности обработки входных данных программой. Эти тесты сосредотачиваются на том, как программа взаимодействует с входными данными, какие данные она принимает, как их обрабатывает и какие результаты выдает в ответ.

Некоторые ключевые аспекты тестов на прием и обработку входных данных представлены ниже:

1 Проверка корректности форматов данных: тесты проверяют, что программа правильно обрабатывает данные в соответствии с их форматом, например, числовые данные, строки, даты, времена и так далее. Если формат данных неверен, программа должна либо отклонить их, либо преобразовать в корректный формат.

2 Проверка обработки исключительных ситуаций: тесты должны включать в себя ситуации, когда входные данные не соответствуют ожиданиям, и программа должна правильно обрабатывать исключения или ошибки.

3 Проверка обработки нулевых и пустых значений: тесты могут включать в себя проверку того, как программа обрабатывает нулевые значения и пустые строки.

4 Проверка диапазона данных: если программа должна обрабатывать данные в определенном диапазоне, тесты должны убедиться, что она это делает корректно.

5 Проверка взаимодействия с внешними данными: в случаях, когда программа взаимодействует с внешними источниками данных тесты могут проверять корректность запросов и обработку возвращаемых данных.

6 Тестирование ввода через интерфейс пользователя: если программа имеет пользовательский интерфейс, тесты могут включать в себя сценарии взаимодействия пользователя с приложением, чтобы убедиться, что ввод обрабатывается правильно.

7 Автоматизация и изоляция: тесты на прием и обработку входных данных легко автоматизировать и должны выполняться в изолированной среде, чтобы исключить внешние факторы.

8 Создание реалистичных тестовых данных: важно создавать тестовые данные, которые максимально приближены к реальным сценариям использования приложения.

Тестирование на прием и обработку входных данных помогает обнаружить ошибки, связанные с некорректными или некорректно обработанными данными, и убедиться в корректности работы программы в различных сценариях ввода.

В таблице 2.1 показаны наименования тестов ввода и вывода с рассматриваемым тестовым случаем.

Таблица 2.1 – Именования тестов

|  |  |
| --- | --- |
| Название теста | Тестовый случай |
| TestSignIn | Проверка входа пользователя в систему с корректными данными. |
| TestDataFetching | Проверка успешного запроса из базы Supabase на получение данных. |
| TestSignUp | Проверка регистрации пользователя с корректными данными |

Необходимые функции или окружения могут быть замоканы – таким образом удаётся избежать ситуации нежелательного вмешательства в реальную базу данных. Также можно клонировать базу данных и иметь свежую копию, предназначенную для тестовую.

## 2.2 Тесты на правильность выполнения основных функций

Тесты на корректность выполнения основных функций программного продукта предназначены для проверки того, что ключевые функции и алгоритмы программы работают правильно и выполняют свою основную задачу в соответствии с заявленными требованиями. Эти тесты фокусируются на проверке правильности работы отдельных частей кода и их взаимодействия друг с другом. Некоторые аспекты тестов на корректность выполнения основных функций программы приведены ниже:

1 Тестирование основных алгоритмов: тесты должны проверять корректность выполнения ключевых алгоритмов, которые обеспечивают основную функциональность программы. Это может включать в себя математические расчеты, обработку данных, фильтрацию, сортировку и другие алгоритмы.

2 Проверка обработки разных случаев: тесты должны включать в себя разнообразные сценарии использования программы, включая различные вари-анты входных данных, краевые случаи и исключительные ситуации. Это позволяет убедиться, что программа правильно обрабатывает разнообразные сценарии.

В таблице 2.2 показаны наименования тестов с рассматриваемым тестовым случаем.

Таблица 2.2 – Тесты на правильность выполнения общих функций

|  |  |
| --- | --- |
| Название теста | Тестовый случай |
| CanvasDrawingBinding | Проверка рабочей привязки при рисовке новых стен. Привязка может осуществляться к другим стенам или к базовым углам рисовки: 0, 45, 90, 270 градусов. |
| LayoutSavedAfterChange | Лэйаут сохраняется локально и в базу данных после рисовок пользователя. |
| LayoutDeletedCreated | Лэйаут создаётся или удаляется успешно, если пользователь залогинен в приложении. |

## 2.3 Тесты на вывод результатов

Тесты на обработку и вывод результатов предназначены для проверки правильности обработки данных программой и корректного вывода результата или информации пользователю. Эти тесты оценивают, насколько хорошо программа взаимодействует с внешним миром и предоставляет пользователю ожидаемые результаты.

Некоторые ключевые аспекты тестов на обработку и вывод результатов:

1 Тестирование вывода результатов: этот тип тестов проверяет, что программа корректно форматирует и выводит результат своей работы. Это может включать в себя проверку правильности вывода текстовых сообщений, генерации отчетов, создания файлов и других способов предоставления ин-формации пользователю.

2 Тестирование интерфейса пользователя: если программа имеет графический интерфейс (GUI), тесты проверяют правильность отображения элементов интерфейса, их доступность и функциональность. Это также включает в себя тестирование навигации, взаимодействия с элементами управления и общего пользовательского опыта.

3 Тестирование обработки ошибок и исключений: эти тесты оценивают, как программа обрабатывает ситуации ошибок и исключений, включая вывод информативных сообщений об ошибках, корректное завершение про-граммы и предотвращение возможных сбоев.

В таблице 2.3 приведены некоторые тесты вывода данных.

Таблица 2.3 – Тесты вывода данных

|  |  |
| --- | --- |
| Название теста | Тестовый случай |
| PdfGenerated | Проверка успешной генерации PDF из текущей конфигурации. |
| LayoutSaved | Проверка сохранения конфигурации после действий пользователя. |
| SummaryGenerated | Проверка успешной генерации информации о заказе из текущей конфигурации. |

## 2.4 Результаты прохождения тестов

Для разработки тестов выбран самый популярный на данный момент инструмент для тестирования JavaScript/TypeScript приложений – Jest. Фреймворк предоставляет возможности параллельного выполнения тестов, заданий моков и многое другое.

На рисунке 2.1 показан результат работы тестов.

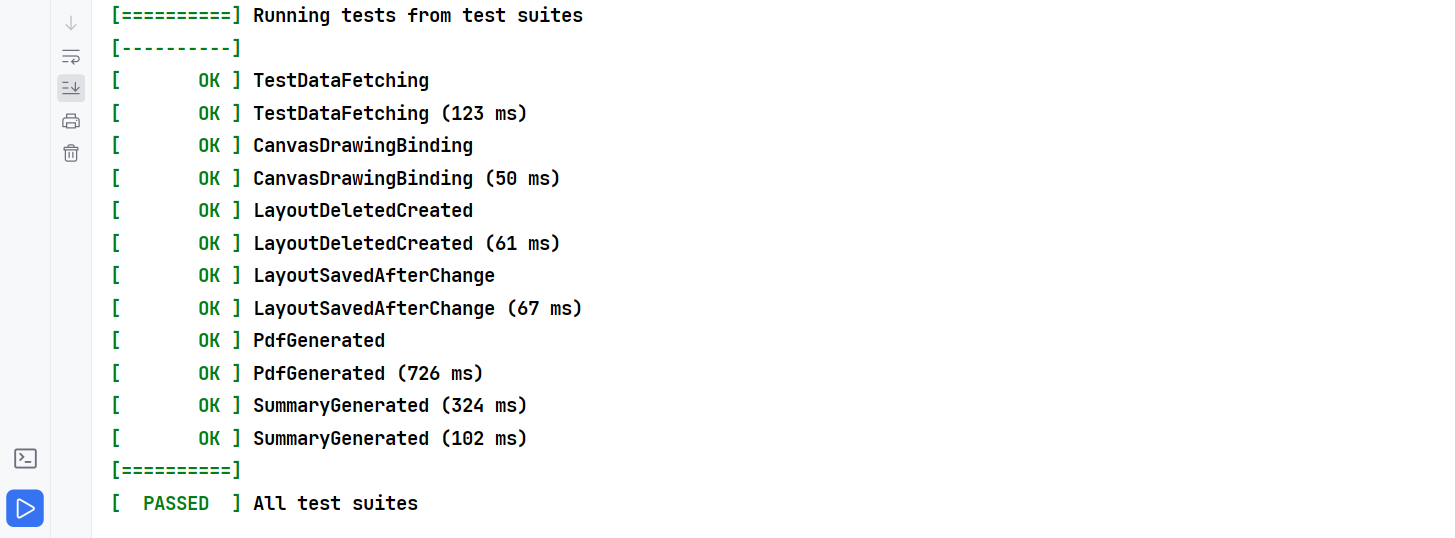


Рисунок 2.1 – Результаты работы некоторых тестов

Ввиду того, что приложение разрабатывается по методологии Test Driven Development, сначала создаются тесты – а после под них пишется функционал. Соответственно, на текущем этапе весь функционал замокан – и тесты по определению проходятся с успешным результатом.

# 3 Технико-экономическое обоснование разработки

## 3.1 Краткая характеристика программного продукта

Целью является создание полноценного веб-решения для конфигурирования и расстановки объектов любого типа в помещении и возможности визуально оценить полученную конфигурацию с помощью 3D-рендера.

Инструмент предоставляет возможность взаимодействовать с моделями помещений в 3D-формате, адаптированными к интерфейсу пользователя. Благодаря такой технологии, каждый человек может визуализировать свое помещение, изменяя параметры и детали, такие как стены, окна, мебель (или иные любые типы объектов) и отделка. Пользователи могут легко и быстро настраивать различные варианты дизайна и экспериментировать с различными комбинациями.

Это дает возможность оценить, как объекты будут выглядеть и сочетаться друг с другом в реальном мире, помогая принять рациональные решения и избежать неудачных композиций.

Для клиентов, которые заходят на сайт и желают визуально сконфигурировать мебель для своей квартиры, использование онлайн 3D конфигуратора предоставляет несколько значимых преимуществ. Клиенты могут наглядно увидеть, как мебель и другие предметы будут выглядеть в их конкретном интерьере. Это позволяет им представить окончательный результат и принять более обоснованное решение о выборе мебели. Они могут экспериментировать с различными дизайнами и расстановками, визуализируя их в 3D-формате на экране компьютера или мобильного устройства.

Продающей компании предоставляется простой способ:

– конфигурировать цены, каталоги, время доставки и другие параметры объектов;

– загружать модели объектов;

– просматривать информацию о заказах, наиболее популярных единицах товаров, конфигурации пользователей.

Клиенту предоставляется возможным:

– схематичная отрисовка помещения;

– выбор параметров стен: толщина, высота, обои или покрытие, что актуально, если компания продаёт обои, например;

– расстановка продаваемых объектов на сцене;

– получение детальной информации о финальной стоимости такого набора объектов и финального предложения от заказчика;

– просмотр отрендеренной 3D-сцены.

Таким образом продающие компании могут обеспечить то, что клиент, имея такое удобное решение, сразу сконфигурирует нужный набор и сразу в конкретной компании купит все необходимые элементы объектов.

## 3.2 Расчёт затрат на основную заработную плату

Расчет затрат на основную заработную плату команды разработчиков производится по следующей формуле 3.1:

|  |  |
| --- | --- |
|  | (3.1) |

где n – количество исполнителей, занятых разработкой конкретного ПО;

– коэффициент премий;

– часовая заработная плата i-го исполнителя (руб.);

– трудоемкость работ, выполняемых i-м исполнителем (ч).

Количество рабочих часов в месяце принято равным 168 часов.

Для разработки программного обеспечения потребуется команда из четырех специалистов:

* JavaScript / TypeScript / WebGl разработчик с должностным окладом 2040 р. и трудозатратами 72 часов на разработку программного средства и поддержку БД;
* дизайнер с должностным окладом 1600 р. и трудозатратами 36 часов на разработку UI, моделей и ассетов;
* Данные по заработной плате команды разработчиков представлены на дату 26.11.2023 г. Расчет затрат на основную заработную плату () приведен в таблице 3.1.

Таблица 3.1 ‒ Расчет затрат на основную заработную плату команды

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Участник команды | Месячный оклад, руб. | Часовой оклад, руб. | Трудоёмкость работ, ч | Итого, руб. |
| JS/TS разработчик | 2040 | 14 | 72 | 1008 |
| Тестировщик | 960 | 6 | 20 | 120 |
| UX/UI дизайнер | 1600 | 10 | 36 | 360 |
| Итого | | | | 1488 |
| Премия (50%) | | | | 744 |
| Всего затраты на основную заработную плату разработчиков | | | | 2232 |

## 3.3 Расчет затрат на дополнительную заработную плату

Дополнительная заработная плата определяется по формуле 3.2:

|  |  |
| --- | --- |
|  | (3.2) |

где – затраты на основную заработную плату, р.,

‒ норматив дополнительной заработной платы, 15%.

Размер дополнительной заработной платы исполнителей, исходя из выше приведенной формулы, составит:

|  |  |
| --- | --- |
|  |  |

# Заключение

Основное назначение BPMN – визуализация бизнес-процессов, что позволяет бизнес-аналитикам и участникам процесса понимать его логику и взаимосвязи.

Основная цель IDEF – описать различные аспекты бизнес-процесса, включая структуру данных, потоки информации и взаимодействия между участниками и системами.

В зависимости от конкретных потребностей и контекста проекта, можно выбрать подходящую методологию моделирования бизнес-процессов. Обе методологии имеют свои преимущества и гибкость для моделирования и оптимизации процессов организации.

В ходе выполнения лабораторной работы была составлена схемы в соответствии с рассматриваемыми методологиями и оформлен отчёт, а также произведено сравнение методологий. Цели лабораторной работы можно считать достигнутыми.

# Список используемых источников

[1] BPMN диаграмма [Электронный ресурс]. – Режим доступа: https://vc.ru/dev/273096-chto-takoe-bpmn-diagramma-i-zachem-ona-nuzhna-v-razrabotke. – Дата доступа: 29.10.2023

[2] СТП БГУИР 2017 [Электронный ресурс]. – Режим доступа: https://library.bsuir.by/ru/-standarty-po-diplomnomu-proektirovaniyu. – Дата доступа: 29.10.2023