**Введение**

На учебной практике была поставлена задача, разработать Web-приложение «Task-A».

Цель учебной практики заключается в помощи пользователям с распределением времени, путём планирования задач.

Создаваемое приложение будет рассчитано на любого рода пользователей.

Далее приведем краткое описание разделов пояснительной записки.

Первый раздел носит название «Анализ задачи». В нем вы сможете ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе «Инструменты разработки» будет рассмотрена среда, в которой создается данное игровое приложение. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачей.

В разделе «Проектирование задачи» будут рассмотрены основные аспекты разработки игрового приложения. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

«Реализация задачи» – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полная аннотация файлов используемых в данном проекте.

Четвертый раздел – «Тестирование». В нем будет описано полное и функциональное тестирование данной программы, т.е. будет оттестирован каждый пункт меню, каждая операция, которая выполняется приложением. Будут смоделированы все возможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе «Применение» будет описано назначение, область применения, среда функционирования игрового приложения. Также в нем будет описано использование справочной системы.

«Заключение» будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В разделе «Список используемых источников» будет приведен список используемых при разработке источников.

В приложениях к пояснительной записке будет приведен листинг программы с необходимыми комментариями и диаграммы.

**1 Анализ задачи**

* 1. **Постановка задачи**

**1.1.1 Организационно-экономическая сущность задачи**

Темой данного проекта является разработка разработка Web-приложение «Task-A».

Данное Web-приложение предназначено для людей любого возраста.

Данный программный продукт может быть использован в любое время, не зависимо от того нуждается ли пользователь в данной продукции или он просто хочет ознакомиться с данной предметной областью.

Источником получения данных является СУБД H2, с которой пользователь взаимодействует посредством приложения.

Web-приложение имеет цифровой аналоги, как: Microsoft ToDo, Todoist, TickTick и др.

**1.1.2 Функциональные требования**

На странице сайта будут представлены две таблицы. В первой таблице будут отображаться задачи, поставленные пользователем. Во второй таблице будут отображаться задачи пользователя, помеченные значком “Выполнено”, выполнение задачи происходит посредством нажатия кнопки “Выполнено” в первой таблице соответствующей задаче. Пользователь будет иметь возможность добавлять задачи, редактировать имеющиеся задачи, а так же удалять имеющиеся задачи.

К поставленной задаче были заявлены следующие функциональные требования, который может выполнять пользователь:

* Просмотр задач: Пользователь должен иметь возможность просматривать установленные, а так же выполненные задачи;
* Создание задач: Пользователь должен иметь возможность создавать новые задачи и добавлять их в свой список дел;
* Удаление задач: Пользователь должен иметь возможность удалять задачи из списка дел;
* Редактирование задач: Пользователь должен иметь возможность редактировать задачи в списке дел;
* Отметка выполненных задач: Пользователь должен иметь возможность отмечать задачи в списке дел.

**1.1.3 Описание входной, выходной и условно постоянной информации**

Таблица 1 – Функции программы с описание с входной, выходной и условно-постоянной информации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Категория пользоват  елей | Наименова  ние процесса | Краткое описание алгоритма выполнения процесса | Входная информация | Выходная информация | Условно–постоянная информация |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | Пользователь | Создание задачи | Пользователь может создать новую задачу | Заголовок задачи, описание задачи | Отображение на главной таблице | Отсутствует |
| 2 | Пользователь | Редакти- рование задачи | Пользователь может редактировать задачи | Заголовок задачи, описание задачи | Заголовок задачи, описание задачи | Отсутствует |
| 3 | Пользователь | Удаление заачи | Пользователь может просмотреть рекорд в главном меню | Отсутствует | Отсутствует | Отсутствует |
| 4 | Пользователь | просмотр информации о разработчике | Пользователь может узнать информацию о разработчике | Отсутствует | Перечень информации о игре | Отсутствует |

**1.1.4 Нефункциональные требования**

**Требования к применению:**

Данный программный продукт поможет пользователям с организацией своего времени, управлением задачами и личными делами, а так же планированием проектов, встреч и новых задач.

**Требования к интерфейсу:** Пользовательский интерфейс будет прост и понятен. Будут использоваться приятные, насыщенные тона цветов. Текст сайта будет разборчив и понятен.

**Требования к производительности:**

Требования к производительности заключаются в использовании структурированного и оптимизированного кода, для обеспечения наибольшей производительности.

**Требования к реализации:**

Для реализации будет использоваться язык Java, с фреймворком Spring Boot, для создания интерфейса будут использоваться языки HTML и CSS.

**Требования к надежности:**

Для данного продукта будет реализована система создания отдельной базы данных для каждого пользователя, для обеспечения надёжности.

* 1. **Диаграмма вариантов использования**

Проектирование – один из важных шагов при разработке программы, который очень часто игнорируется начинающими разработчиками. Как результат, у них нет чёткого плана дальнейших действий, и проект может быть отложен.

Обычно при проектировании разработчики изображают систему графически, поскольку человеку легко разобраться в таком представлении. Именно поэтому вместо написания громоздких текстов про каждую возможность будущей программы разработчики строят различные диаграммы для описания своих систем. Это помогает им не забывать, что нужно реализовать в программе, и быстро вводить в курс дела своих коллег.

На диаграмме вариантов использования изображаются:

- акторы — группы лиц или систем, взаимодействующих с системой;

- варианты использования — сервисы, которые наша система предоставляет акторам;

- комментарии;

- отношения между элементами диаграммы.

С диаграммой вариантов использования для Web-приложения «Task-A» можно ознакомиться в Приложении А.

* 1. **Модель данных**

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных.

Модели данных служат для проектирования структуры постоянных хранилищ данных, используемых системой. Профайл на языке UML для проектирования базы данных предоставляет разработчикам базы данных набор элементов моделирования, позволяющих разрабатывать подробный макет таблиц в базе данных и моделировать макет физической памяти базы данных. Профайл базы данных на языке UML также предоставляет конструкции для моделирования целостности по ссылкам (ограничений и триггеров), а также хранимых процедур, предназначенных для управления доступом к базе данных.

Модели данных могут создаваться на уровне предприятия, отдела или отдельного приложения. Модели данных на уровне предприятия или отдела могут использоваться для предоставления стандартных определений для ключевых бизнес-сущностей (таких как клиент и сотрудник), которые будут применяться всеми приложениями всего бизнес-процесса или его части. С помощью этих типов Моделей данных можно также определить, какая система предприятия будет "владельцем" данных для конкретной бизнес-сущности и какие другие системы будут пользователями данных.

С моделью данных созданную для игрового приложения «Task-A» можно ознакомиться в Приложение Б.

**2 Проектирование**

**2.1 Выбор стратегии разработки и модели жизненного цикла**

Для разработки игрового приложения следует выбрать стратегию разработки и модель жизненного цикла. Осуществляем выбор посредством составления таблиц:

Таблица 2 – Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории требований | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Да | Да | Да | Нет | Нет | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да | Да | Да | Да | Нет | Нет |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет | Нет | Нет | Нет | Да | Да |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Нет | Нет | Да | Нет | Да | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет | Нет | Да | Нет | Да | Да |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Нет | Нет | Да | Да | Да | Да |

**Вычисления: 4 за каскадную, 4 за V- образную, 5 за RAD, 5 за инкрементную, 3 за быстрого прототипирования и 3 за эволюционную.**

**Итог: На основе результатов заполнения табл. 2 подходящей является RAD модель и инкрементная модель.**

Таблица 3 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта? | Да | Да | Да | Да | Нет | Нет |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |

Вычисления: 5 за каскадную, 5 за V-образную, 4 за RAD, 5 за инкрементную, 2 за быстрого прототипирования и 3 за эволюционную.

Итог: На основе результатов заполнения табл. 3 подходящими являются каскадная, V-образная и инкрементная модели.

Таблица 4 – Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории коллектива пользователей | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки? | Нет | Нет | Нет | Да | Да | Да |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет | Нет | Да | Нет | Да | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 0 за каскадную, 0 за V-образную, 2 за RAD, 1 за инкрементную, 4 за быстрого прототипирования и 2 за эволюционную.

Итог: На основе результатов заполнения табл. 4 подходящей является модель быстрого проектирования.

Таблица 5 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории типов проекта и рисков | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1. | Разрабатывается ли в проекте продукт нового для организации направления? | Нет | Нет | Нет | Да | Да | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| 3. | Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
|  | | | | | | | |
|  |  |  |  |  |  |  |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| 8. | Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| 9. | Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

**Вычисления: 3 за каскадную, 4 за V-образную, 4 за RAD, 8 за инкрементную, 6 за быстрого прототипирования и 8 за эволюционную.**

**Итог: На основе результатов заполнения табл. 5 подходящей является инкрементная и эволюционная модели.**

**2.2 Инструменты разработки**

**Для разработки данного проекта будет выбрана среда разработки InteliJ IDEA, предоставляющая возможность работать с многими языками программирования, как Python, Kotlin, Java и д.р., так же она является одной из самых актуальных сред разработки и имеет широкий спектр использования от обучающих курсов до использования в больших компаниях для разработки больших проектов.**

**Разработка будет проводится на таких языках программирования, как:**

* **HTML (Hyper Text Markup Language) - будет использован для разметки сайта и создания его front-end части, то есть стиля.**
* **Java - объектно-ориентированный язык программирования, используемый для создания различных приложений, включая веб-приложения, мобильные приложения и настольные приложения. Этот язык программирования будет использован для обеспечения работоспособности сайта, то есть back-end часть.**
* **Spring Boot - фреймворк, дающий возможность создания микросервисов и веб-приложений на языке Java. Spring Boot предоставляет широкий спектр функций, таких как управление зависимостями, автоматическая конфигурация, встроенные серверы, управление ошибками, предоставляет множество библиотек и плагинов, которые помогают в разработке высококачественного приложения. Данный фреймворк будет использован в качестве основной платформы, на которой, с помощью языка программирования Java, будет построен сайт.**
* **Thymeleaf - современный шаблонизатор для веб-приложений на языке Java, предоставляющий возможность создавать элегантные и естественные шаблоны, которые могут быть быть корректно отображены в браузерах и использоваться в качестве статических прототипов. Thymeleaf предоставляет естественные шаблоны HTML, которые выглядят и работают, как обычный HTML, что позволяет использовать их в качестве полезных дизайн-артефактов. Данный шаблонизатор будет использован для разработки дизайна веб-приложения с использованием HTML.**
* **Java Persistence API (JPA) - стандартный интерфейс для управления объектно-реляционным отображением (ORM) в Java. Он предоставляет возможность работать с базами данных, используя объекты Java, вместо написания SQL-запросов. JPA предоставляет абстракцию уровня хранения данных, которая позволяет разработчикам работать с базами данных, он предоставляет возможность создавать, читать, обновлять и удалять данные из базы данных, используя объекты Java, а не SQL-запросы. Так же JPA предоставляет возможность управлять отношениями между объектами Java и таблицами базы данных. Данный интерфейс будет использован для хранения данных о задачах, создаваемых пользователем, и о конечном выводе их в список задач на сайте.**

**Иные инструменты, используемые при разработке и написании сопутствующей документации:**

* **Web-ресурс DRAW.IO - будет использоваться для создания графической части и разработки UML-диаграмм;**
* **Microsoft Documents - будет использоваться для написания всей документации;**

**Разработка проекта будет происходить на компьютере со следующими параметрами:**

* **Процессор 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 Ghz**
* **Объём оперативной памяти 8 Гб**
* **Объём места на жёстком диске 512 Гб**
* **Видеокарта Intel(R) Iris(R) Xe Graphics**
* **ОС Windows 10 Pro**

**2.3 Разработка UML-диаграмм**

**2.3.1 Диаграмма вариантов использования**

Проектирование – один из важных шагов при разработке программы, который очень часто игнорируется начинающими разработчиками. Как результат, у них нет чёткого плана дальнейших действий, и проект может быть отложен.

Обычно при проектировании разработчики изображают систему графически, поскольку человеку легко разобраться в таком представлении. Именно поэтому вместо написания громоздких текстов про каждую возможность будущей программы разработчики строят различные диаграммы для описания своих систем. Это помогает им не забывать, что нужно реализовать в программе, и быстро вводить в курс дела своих коллег.

На диаграмме вариантов использования изображаются:

- акторы — группы лиц или систем, взаимодействующих с системой;

- варианты использования — сервисы, которые наша система предоставляет акторам;

- комментарии;

- отношения между элементами диаграммы.

С диаграммой вариантов использования для Web-приложения «Task-A» можно ознакомиться в Приложении А.

**2.3.2 Диаграмма классов**

Диаграмма классов — структурная [диаграмма](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_(UML)) языка моделирования [UML](https://ru.wikipedia.org/wiki/UML), демонстрирующая общую структуру иерархии [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) системы, их коопераций, [атрибутов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B5_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0) (полей), [методов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_(%D1%8F%D0%B7%D1%8B%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)), интерфейсов и взаимосвязей (отношений) между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

При разработке программного продукта была поставлена задача разработать диаграмму классов, которая показывает общую структуру иерархии [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) системы. С данной диаграммой можно ознакомиться в Приложении В.

**2.3.3 Диаграмма объектов**

Диаграмма объектов в языке моделирования [UML](https://ru.wikipedia.org/wiki/UML) предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.

Диаграмма объектов описывает конкретные экземпляры объектов и напрямую соотносится с [диаграммой классов](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%BE%D0%B2), которая даёт общее представление о конфигурации системы. Она используется для документирования структур данных и создания статических снимков состояний объектов принимая во внимание реальные экземпляры или прототипы. Динамику поведения объектов обычно изображают в виде последовательности таких диаграмм.

При разработке программного продукта была поставлена задача разработать диаграмму объектов, которая описывает процесс связей между ними в фиксированный момент времени. С данной диаграммой можно ознакомиться в Приложении Г.

**2.4 Разработка пользовательского интерфейса**

Важным элементом проектирования данного программного продукта является описание внешнего интерфейса разрабатываемого Web-приложения.

Для разработки визуального дизайн использовались строгие, сдержанные, насыщенные цвета для удобства использования программного продукта.

В ходе разработки был спроектирован дизайн главной страницы Web-приложения «Task-A». Ранее разработанная структура Web-приложения расположена в Приложении А.

Для организации эффективной работы пользователя нужно создать целостный программный продукт данной предметной области, в котором все компоненты будут сгруппированы по функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс пользователя. Web-приложение должно позволить пользователю решать задачи, затрачивая значительно меньше усилий, чем при работе с разрозненными объектами. Все исходные данные будут разделены на несколько групп.

Прототип – это наглядная модель пользовательского интерфейса. В сущности, это «черновик» созданный на основе представления разработчика о потребностях пользователя. Итоговое отображение программы может отличаться от прототипа. С прототипом главной страницы вы можете ознакомиться в Приложении Д.

Так же поставленной задачей на практику была реализация UX/UI интерфейсов. При создании UX/UI интерфейсов были использованы модульные сетки с целью создания пропорционального, понятного интерфейса. Использовались преимущественно оттенки холодных цветов, для определения строго стиля. Основные разделы доступны с первой страницы. Таким образом был реализован понятный пользовательский интерфейс, созданы макеты под такие устройства как: компьютер, мобильный телефон. Целью проекта была реализация интерфейса для frontend части.

**2.5 Разработка плана работы над проектом**

Для разработки плана над проектом был использован web-ресурс Wrike. Данное программное обеспечение позволяет отслеживать прогресс и анализировать объём работы. Показывает в какие строки может быть выполнена та или иная задача.

В данном случае Excel позволил быстро составить план разработки проекта, с возможностью отмечать степень его готовности. Таким образом это должно увеличить скорость разработки. План разработки составленный в Excel представлен в Приложении Е.

**3 Тестирование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения этапа написания программы было проведено тщательно функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономном режиме. Отчёт о результатах тестирования представлен в таблице 6.

Таблица 6 – Отчёт о результатах тестирования функций для пользователя

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название теста | Действие | Ожидаемый результат | Фактический результат | Результат тестирования |
| 1 | 2 | 3 | 4 | 5 |
| Переход на страницу /new | Нажатие на кнопку «+» | Откроется страница с адресом /new | Открылась страница с адресом /new | Выполнено |
| Создание задачи | Ввод информации задачи, нажатие кнопки “Добавить задачу” | Откроется страница с задачами | Открылась страница с задачами | Выполнено |
| Переход на страницу /info | Нажатие на кнопку «Детальнее» | Откроется страница /info | Открылась страница /info | Выполнено |
| Удаление задачи | Нажатие на кнопку «Удалить» на странице /info | Задача удалиться из списка | Задача удалилась из списка | Выполнено |
| Переход на страницу /edit | Нажать на кнопку «Редактировать» на странице /info | Откроется страница /edit | Открылась страница /edit | Выполнено |
| Редактирование задачи | Изменение текста задачи на странице /edit и нажатие кнопки «Сохранить изменения» | Изменится текст задачи | Текст задачи изменился | Выполнено |
| Переход со страницы /edit на страницу /info | Нажатие кнопки «Назад» | Переход на страницу /info | Переход был осуществлен | Выполнено |
| Переход со страницы /edit на главную | Нажатие кнопки «К списку» | Переход на главную страницу | Переход был осуществлен | Выполнено |
| Переполнение таблицы с задачами | Создание большого количества задач | Появление возможности прокручивать задачи | Задачи можно прокрутить, в случае переполнения таблицы | Выполнено |
| Переполнение таблицы с выполненными задачами | Создание большого количества задач | Появление возможности прокручивать задачи | Задачи можно прокрутить, в случае переполнения таблицы | Выполнено |

В результате проведения тестирования выяснилось, что все ранее оговоренные функции и требования, были разработаны, а также протестированы. Тесты показали, что все функции работают правильно. Следовательно разработанное Web-приложение можно передать заказчику.

В ходе тестирования программного обеспечения продукта на разных устройствах не было выявлено каких-либо ошибок, так как адаптивность **Web**-приложения была проведена на всех стадиях разработки.

**5 Руководство пользователя**

**5.1 Общие сведения о программном продукте**

Разрабатываемый web-ресурс будет носить название «Task-A».

Данный программный продукт является ToDo приложением, служащим пользователю своеобразным списком задач.

Данный программный продукт преимущественно предназначен для студентов, работающих людей, а так же полезен для компаний и производств, старающихся повысить продуктивность своих работников, структурировать производство и выполнение работы.

Данный программный продукт может быть использован на любом устройстве, с любого браузера и в любое время суток.

Быстродействие любой программы во многом зависит от характеристик выбранного

персонального компьютера: рабочей частоты процессора, объема оперативной памяти и т.д.

Несмотря на все реализованные в ней задачи, она легко запускается и функционирует на

любых машинах.

Тестирование проводилось на разных классах ЭВМ и работать с данной программой

было комфортно. Программа разработана на ПК со следующими характеристиками:

* **Процессор 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 Ghz**
* **Объём оперативной памяти 8 Гб**
* **Объём места на жёстком диске 512 Гб**
* **Видеокарта Intel(R) Iris(R) Xe Graphics**
* **ОС Windows 10 Pro**

**5.2 Инсталляция**

**Данный программный продукт не требуется в установки на свой персональный компьютер. Для полноценной работы web-ресурса вам необходимо иметь на своём компьютере установленный интернет-браузер и стабильное подключение к сети интернет.**

**В случаи имеющих пунктов вам потребуется ссылка на web-ресурс (http://taska.com), которую вам требуется вставить в любую поисковую строку, или можно воспользоваться теми же поисковыми системами, введя в них название web-ресурса («Task-A»).**

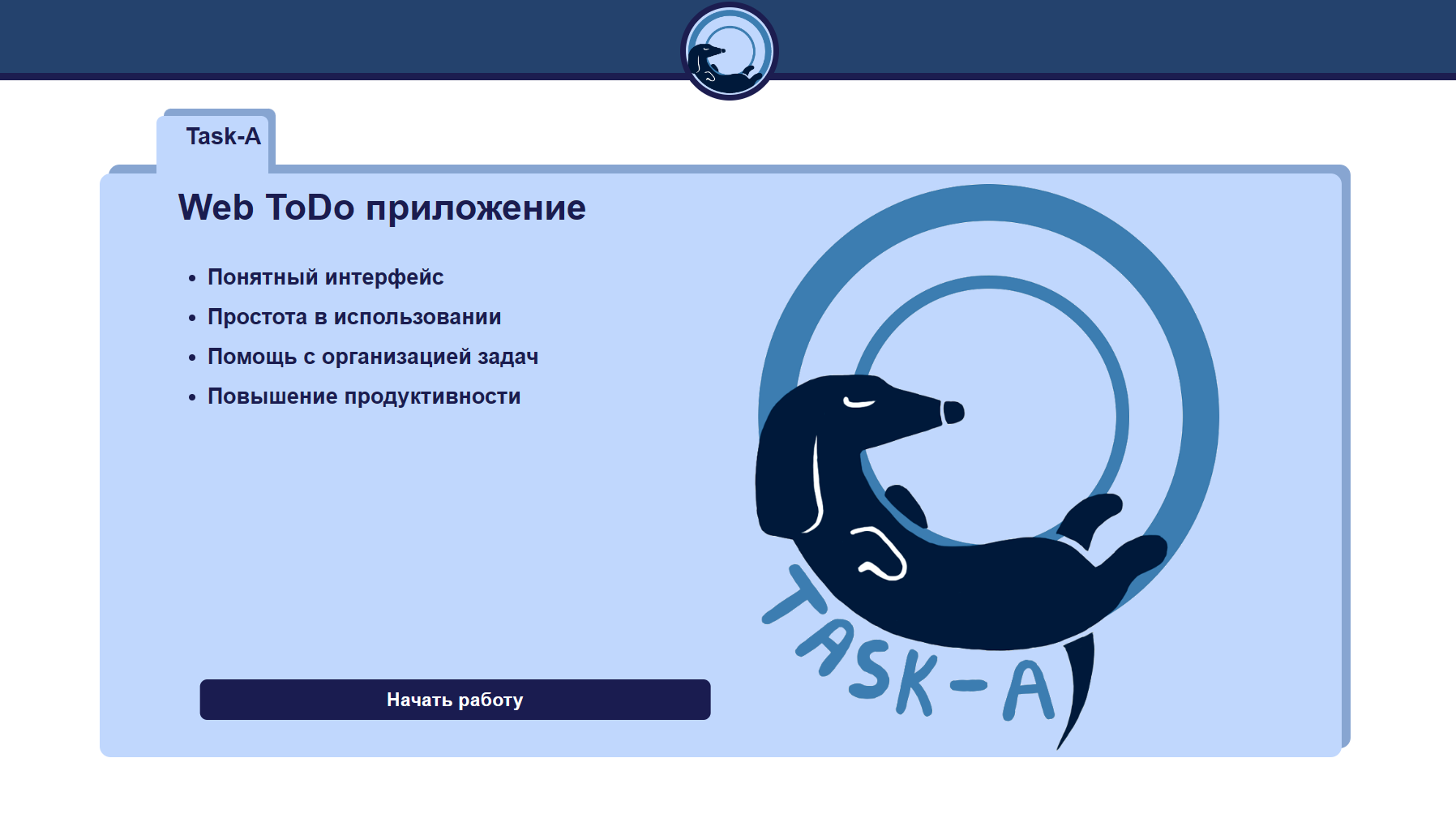
**5.3 Выполнение программы**

**5.3.1 Запуск программы**

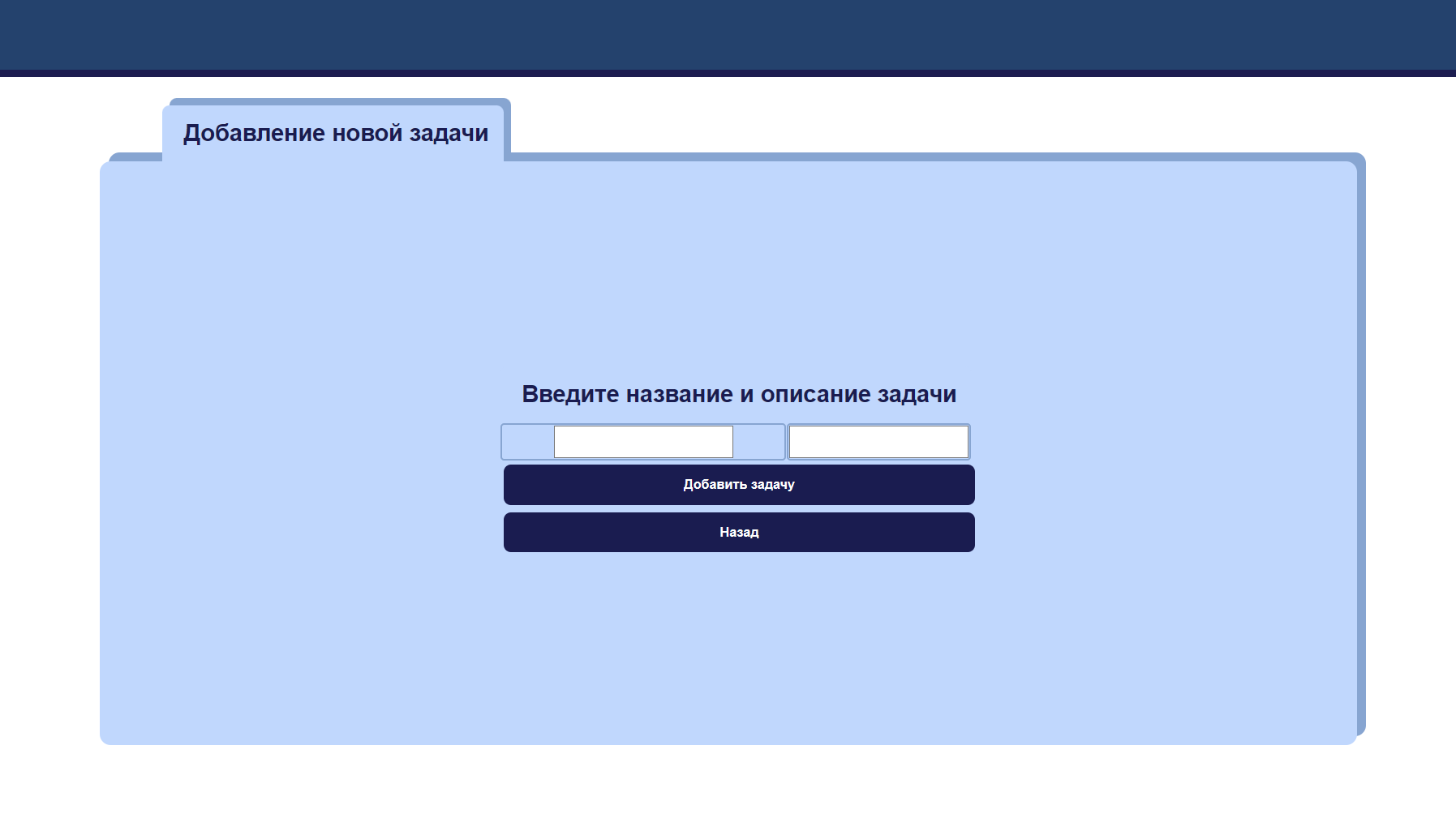
Для запуска web-ресурса «Task-A» вам необходимо иметь стабильное интернетподключение и любой интернет-браузер. После вам потребуется ссылка на web-ресурс (http://taska.com), которую вам требуется вставить в любую поисковую строку, или можно воспользоваться теми же поисковыми системами, введя в них название web-ресурса («Task-A»).

**5.3.2 Инструкция по работе с программой**

**После того, как вы перешли по ссылке, которую вы ввели в поисковую систему ли нашли в поисковике браузера по названию web-ресурса, у вас в браузере откроется главная страница сайта (рисунок 1).**

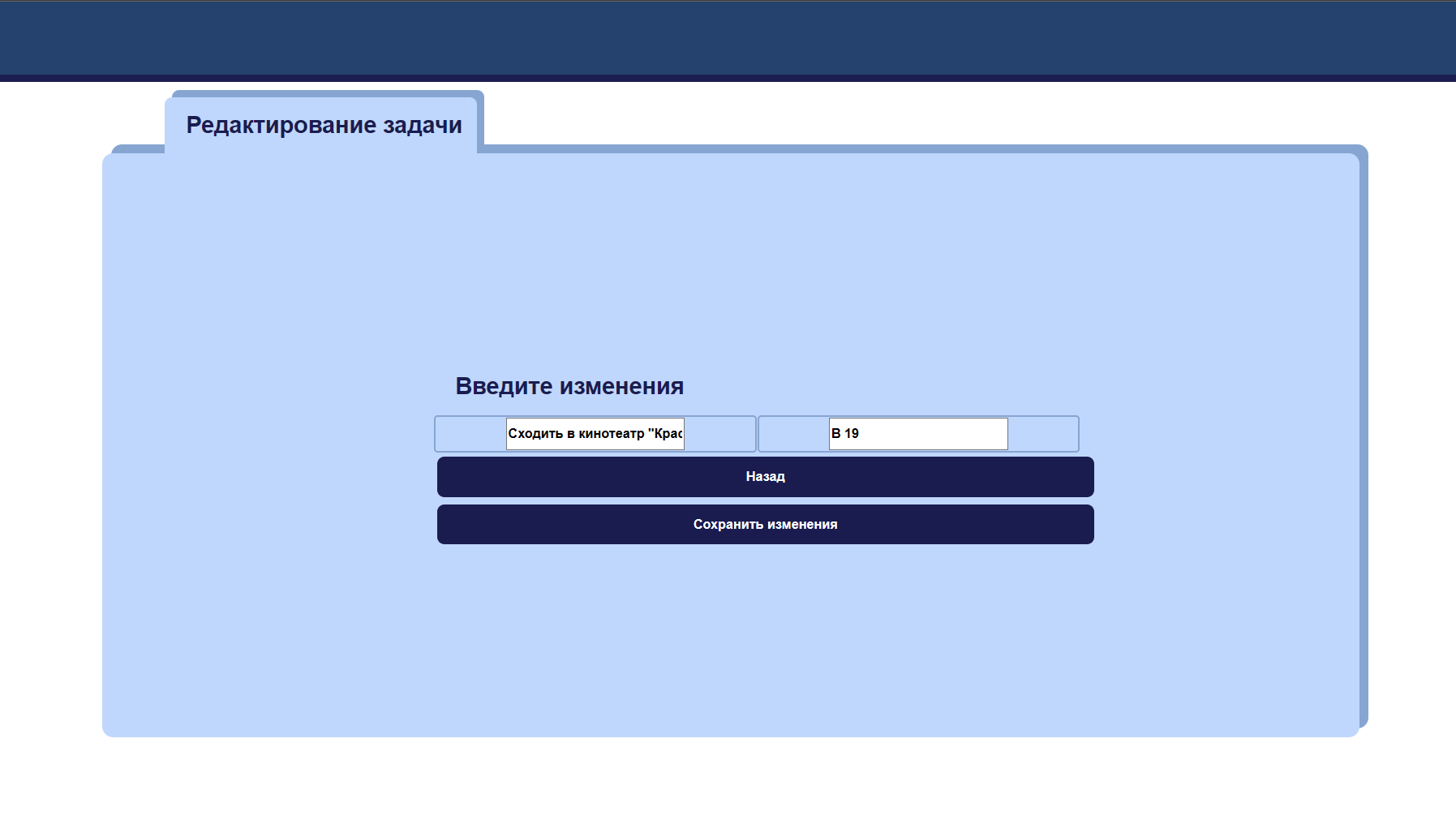
** Для того, чтобы начать работу, пользователю необходимо нажать на кнопку “Начать работу”. После осуществления нажатия кнопки, пользователь переходит на основную страницу, где расположены две таблицы. В первой таблице впоследствии взаимодействия с сайтом будут отображаться поставленные задачи, а во второй, задачи, которые пользователь отметит флажком “Выполнено” (Рисунок 2).**

**Основной задачей данного web-приложения служит создание задач, для того, чтобы это осуществить, пользователю необходимо нажать на круглую кнопку, закрепленную в правом нижнем углу экрана с значком плюса. После нажатия на кнопку, пользователь перейдёт на страницу по созданию новой задачи, где в пустые поля пользователь вводит заголовок задачи и описание. Чтобы добавить задачу, пользователю нужно нажать на кнопку “Добавить задачу”, так же пользователь может вернуться обратно к списку, нажав “Назад” (Рисунок 3).**

** На основной странице, пользователь может посмотреть информацию о задаче, нажав кнопку “Детальнее”. При нажатии кнопки, откроется страница с информацией о задаче, где пользователь может вернуться назад к списку, редактировать задачу или удалить её (Рисунок 4).**

****

**При нажатии кнопки “Редактировать”, пользователь перейдёт на страницу изменения задачи, где может изменить текст и описание, и сохранив изменения, нажав на кнопку “Сохранить изменения”. Так же можно перейти назад, нажав соответствующую кнопку (Рисунок 5).**

** Так же одной из главных функций сайта является выполнение задач, для того, чтобы пометить задачу, как выполненную, пользователю необходимо нажать на кнопку “Выполнено” в первой таблице с задачами, на основной странице сайта (Рисунок 6).**

****

**Заключение**

Целью данного проекта на практику являлась разработка ToDo Web-приложения «Task-A».

Можно заметить, что в web-ресурсе был реализован простой и удобный интерфейс, который позволяет использовать web-приложение тому пользователю, который не обладает отличительными навыками работы с программным продуктом данного типа. При разработке были выполнены требования, поставленные в 1 и 2 разделе документации.

Данное web-приложение предназначено для помощь пользователям структурировать своё время, помочь в распределении рабочего времени. Так же оно будет подходить людям всех возрастов.

После долгого и тщательного тестирования web-приложения были выявлены небольшие ошибки и недоработки, которые в последствии были исправлены на стадии тестирования.

В целом при реализации программного продукта, были выполнены все условия, поставленные на начальном этапе разработки. Из чего мы можем сделать вывод, что web-приложение можно передавать заказчику и вводить в эксплуатацию.

Так же в процессе создания программного продукта была подготовлена программная документация.

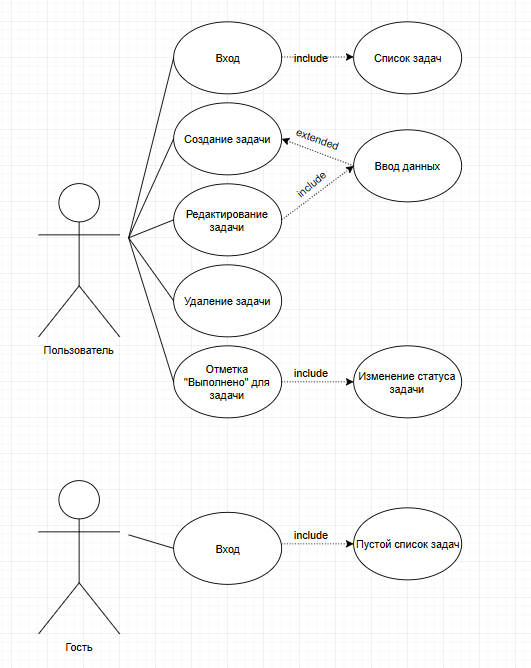
**Список использованных источников**

1 Документация Spring Boot Framework <https://spring.io/guides>

2 Документация СУБД H2 h2database.github.io

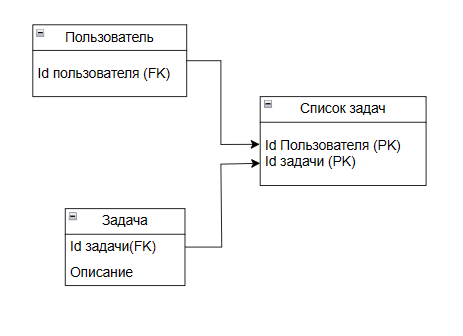
**Приложение А**

**Диаграмма вариантов использования**



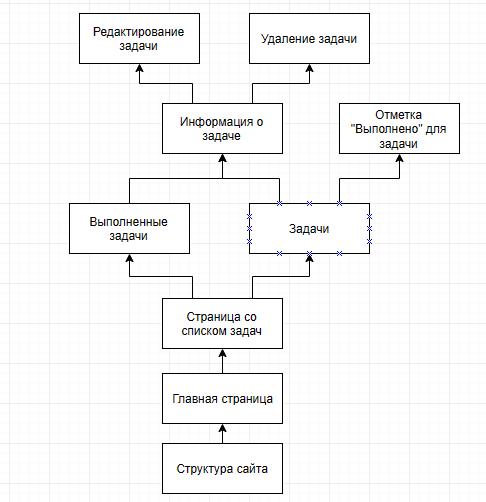
**Приложение Б**

**Модель данных**

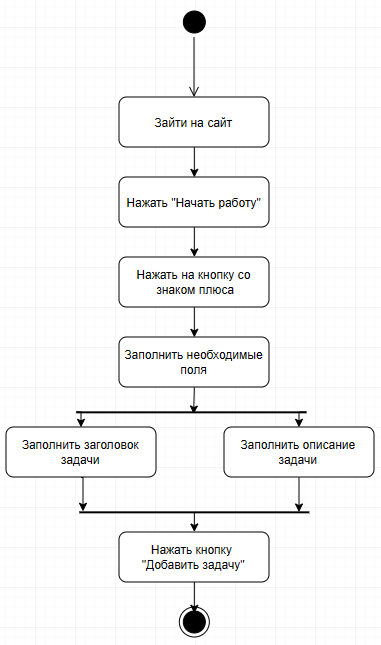


**Приложение В**

**Структура сайта**



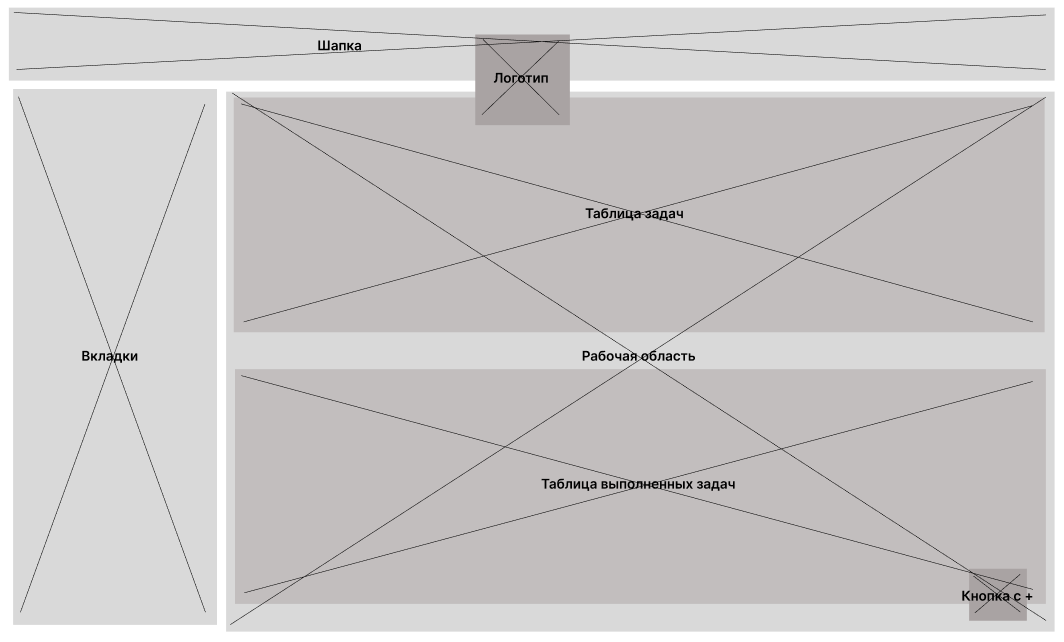
**Приложение Г  
Диаграмма состояния**



**Приложение Д**

**UX прототип пользовательского интерфейса**

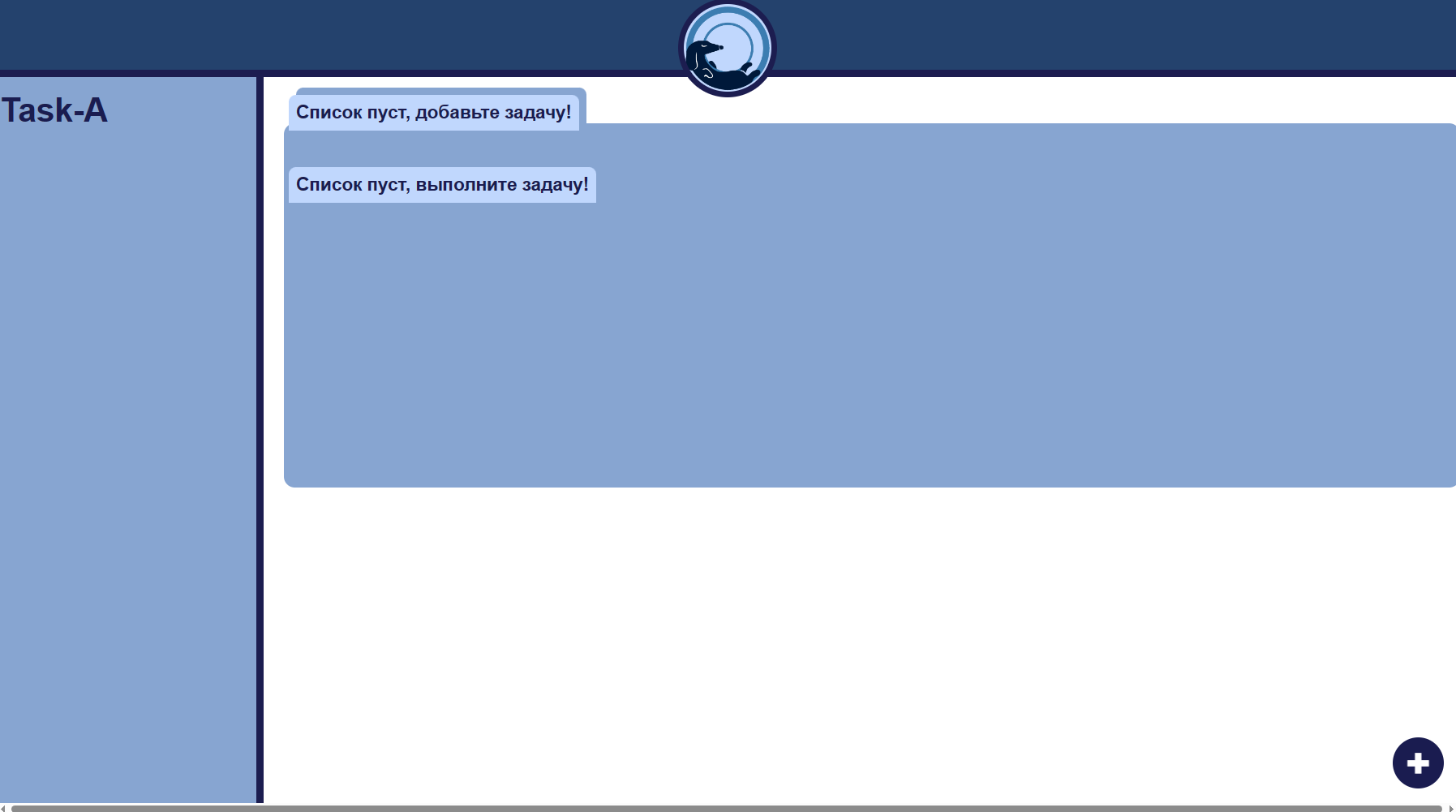
UX прототип пользовательского интерфейса в редакторе Figma - <https://www.figma.com/file/zeLDddjbUORXPqSTmcZFJ6/Figma-basics?type=design&node-id=1669-162202&mode=design&t=FYeootyd7a0y60YJ-0>



***Рисунок 11 - UX прототип интерфейса***

**Приложение Е**

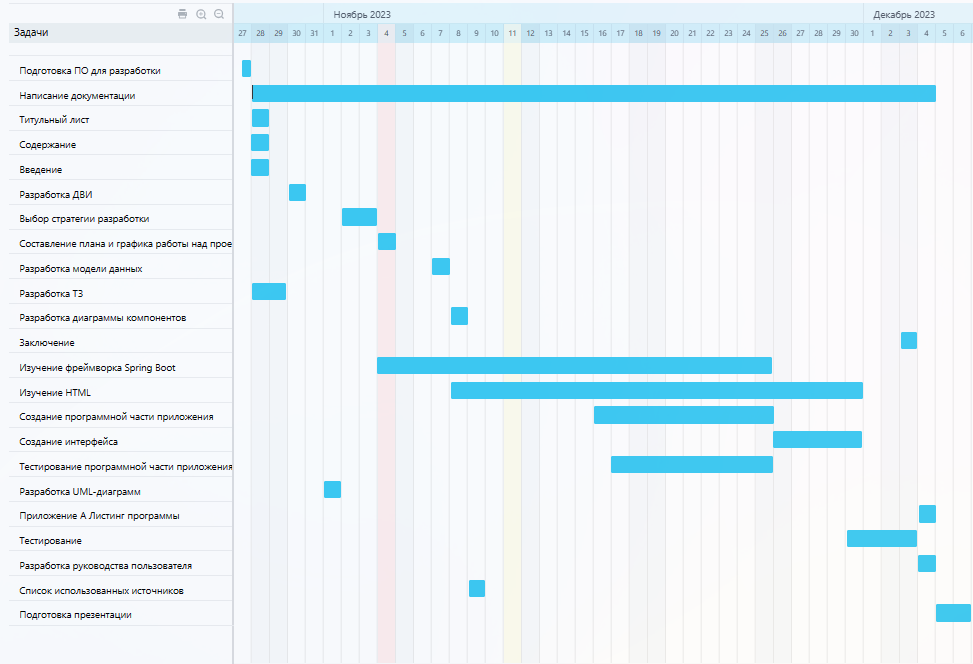
**UI прототип пользовательского интерфейса**

****

***Рисунок 12 - UI прототип основной страницы***

**Приложение Ё**

**Диаграмма Ганта**

****

***Рисунок 13 - Диаграмма разработки проекта***