# СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

АНАЛИТИЧЕСКАЯ ЧАСТЬ 4

1.1 Постановка проблемы 4

1.2 Описание функционала системы 4

ПРОЕКТНАЯ ЧАСТЬ 6

2.1 Структура базы данных 6

2.2 Описание backend-системы 7

2.3 Описание frontend-системы 8

ЗАКЛЮЧЕНИЕ 10

# ВВЕДЕНИЕ

В современном мире люди постоянно занимаются множеством различных дел, требующих внимание к деталям. Из-за большого количества информации, которые человек постоянно получает, некоторые идеи забываются с течением времени.

Таким образом, актуальность выбранной темы обусловлена тем, что в связи с необходимостью многих людей систематизировать свои планы, необходимо разработать веб-приложение, позволяющее создавать проекты и подзадачи, а также отслеживать прогресс по ним.

В рамках курсового проекта данное приложение было создано, поэтому в данном документе будет описан его функционал. Также в данном документе будут описаны системы, из которых состоит данное приложение, а также принципы, используемые при разработке.

Работа разделена на 2 раздела – аналитической части, состоящей из постановки проблемы и описания функционала системы, и проектной части, в которой описаны frontend, backend и база данных данной системы.

# АНАЛИТИЧЕСКАЯ ЧАСТЬ

## Постановка проблемы

В современном мире для людей абсолютно обычно заниматься в течение жизни несколькими проектами одновременно – школьники и студенты получают домашние задания по нескольким предметам одновременно, а сотрудники получают на своих работах несколько задач.

Поскольку большинство задач людей являются важными, а некоторые из них пересекаются между собой, людям приходиться запоминать большое количество информации. Очевидно, что данный подход является неудобным, поскольку цена ошибки и забытой задачи является очень высокой. Именно поэтому на работах создаются или внедряются системы, отслеживающие задачи, а в школах вводятся электронные дневники.

Данный подход в некоторой степени решает проблему – люди систематизируют свои задачи по определенной сфере. Однако сфер у каждого человека несколько, и внедрение данных об одной сфере в другую является проблематичной.

На телефонах существуют приложения, в которые можно внести информацию о своих делах, но по своему функционалу они ушли недалеко от записной книжки, на их заполнение требуется множество времени. Кроме того, необходимо не забывать постоянно вносить обновления.

Таким образом, на данный момент не существует приложения, позволяющего отслеживать задачи из разных сфер жизни человека, что обуславливает необходимость создания системы, решающей данную проблему.

Разработанное приложение обладает минимальным функционалом для того, чтобы с ним можно будет работать, а методы, используемые при разработке, позволяют легко его изменять и с течением времени внедрять микросервисную архитектуру, а также интегрировать её в другие системы, что позволит решить проблему и получить систему, в которой хранятся задачи из разных сфер жизни человека. Текущий функционал подробнее описан в следующем пункте данного раздела.

## Описание функционала системы

Весь функционал системы условно можно разделить на 2 части – работа с проектами и дополнительные функции.

Дополнительные функции включают в себя авторизацию, регистрацию, изменение пароля, фильтрация данных и выход из системы.

К функциям, связанным с проектом, относятся:

* Создание проекта.
* Изменение описания проекта.
* Удаление проекта.
* Создание задачи в проекте.
* Просмотр задач проекта.
* Изменение описания задачи.
* Удаление задачи.
* Просмотр доски активных задач в проекте.
* Создания комментария к задаче.
* Изменение комментария к задаче.
* Удаление комментария к задаче.
* Изменение статуса задачи.
* Просмотр своих проектов.
* Просмотр доступных проектов.
* Вступление в проект.

Описанные выше функции разработаны таким образом, что у них есть ограничения, которые позволяют вносить только корректные изменения, однако их легко изменять в зависимости от необходимости, что делает проект удобным и гибким в плане требований. Также стоит отметить, что сущности «проект – задача – комментарий» связаны между собой, нельзя создать задачу у несуществующего проекта, написать комментарий к удаленной задаче и так далее. Также наложены ограничения, связанные с пользователем – менять можно описание только у своих проектов, а удалять или изменять чужие задачи запрещено. Таким образом, в данном программном продукте все функции не только реализованы, но и защищены от некорректных действий пользователя – причем проверки идут не только на стороне клиента, но и на сервере, что положительно влияет на защищенность проекта. Более подробно о реализации данных функций описано в проектной части.

Таким образом, в рамках аналитической части была поставлена проблема и описан функционал программы, которая решает данную проблему.

# ПРОЕКТНАЯ ЧАСТЬ

## 2.1 Структура базы данных

В качестве СУБД была выбрана PostgreSQL в связи с тем, что данная СУБД обладает широким функционалом и множеством возможностей, которые позволят расширять функционал разрабатываемого продукта быстрее и качественнее.

На данный момент база данных состоит из 7 таблиц – одной таблицы, отражающей истории миграций, а также 6 таблиц, непосредственно задействованных в системе. Все данные таблицы были созданы из сущностей в backend-системе. Для id во всех сущностях используется Guid.

Далее приведено краткое описание всех таблиц, не отражающих системную информацию.

### **2.1.1 Таблица Users**

Состоит из id, логина, который должен быть уникальным, и пароля пользователя. Пароль хранится в зашифрованном виде. Все поля являются обязательными к заполнению.

### **2.1.2 Таблица Projects**

Состоит из id, названия проекта, описания проекта, автора проекта и пометки, является ли проект личным. Данная пометка влияет на то, будет ли проект отображаться у других пользователей в качестве доступной к вступлению или нет. Все поля являются обязательными к заполнению.

### **2.1.3 Таблица ProjectUser**

Смежная таблица, отражающая связь между пользователями и проектами.

### **2.1.4 Таблица TaskStatuses**

Таблица-справочник, отражающая статусы задачи. Состоит из id и названия статуса.

### **2.1.5 Таблица Tasks**

Состоит из id, названия задачи, её описания, ссылки на проект, ссылки на исполнителя, ссылки на статус, даты начала и конца. Все поля, кроме дат, являются обязательными.

### **2.1.6 Таблица Comments**

Состоит из id, текста комментария, ссылки на задачу, ссылки на автора комментария и даты, когда был оставлен комментарий. Все поля являются обязательными.

## 2.2 Описание backend-системы

В качестве backend-системы был выбран язык программирования C#, фреймворк .NET. Данный фреймворк обладает множеством возможностей, а также легок в освоении и обладает удобочитаемым кодом, что позволит при необходимости быстро расширить команду, если это потребуется при разработке.

В данную систему внедрено dependency injection и интерфейс IDisposable, что позволило оптимизировать процессы, протекающие в системе, с точки зрения памяти.

Решение состоит из 4 проектов – «Основной», «Данные», «Модели» и «Сервисы».

### **2.2.1 Проект «Данные»**

В данном проекте хранятся сущности базы данных, а также история миграций. Данный проект необходим потому, что при разработке базы данных использовался метод «Code-First». Благодаря данному проекту взаимодействие с базой данных может осуществляться не только с помощью SQL запросов, но и при помощи LINQ.

### **2.2.2 Проект «Сервисы»**

В данном проекте хранятся интерфейсы, описывающие сервисы, и их реализации. Данный проект ответственен за получение данных из базы данных. Кроме того, в данном проекте хранятся классы, отвечающие за шифрование и дешифровку данных, и класс, ответственный за преобразование dao в dto. Стоит отметить, что помимо запрашивающих действий, в сервисах осуществляется проверка возможности их выполнения.

### **2.2.3 Проект «Модели»**

В данном проекте хранятся dto объекты, используемые в проекте.

### **2.2.3 Проект «Основной»**

В данном проекте хранятся контроллеры, к которым обращается frontend, а также объекты, описывающие виды request, response, класс, валидирующий входящее данные, а также класс, описывающий конфигурацию системы.

Контроллеры помимо действий, направленных напрямую на выполнение поставленной задачи, содержит методы, проверяющие, имеет ли запрашивающий доступ к действию. Кроме того, контроллеры выводят информацию об ошибке в случае, если запрашиваемое действие не было совершено, причем в ошибке содержится текстовое объяснение, описывающее причины того, что запрос не был удовлетворен.

## 2.3 Описание frontend-системы

Для frontend-системы был выбран фреймворк Vue, поскольку он является более удобным, чем React, и позволяет реализовывать функции быстро и качественно.

На данный момент приложение рассматривается только для устройств с широкими экранами, однако разделение оформления по файлам позволяет достаточно легко расширить минимальную ширину экрана, на которой сайт будет корректно отображаться.

Frontend можно разделить на несколько условных блоков – API, Assets, Components, Pages, Router, Store.

### **2.3.3 Блок API**

Данный блок содержит в себе методы, запрашивающие информацию у backend системы.

### **2.3.2 Блок Assets**

Данный блок содержит в себе .css файлы, описывающие внешний вид системы, а также применяемые на сайте изображения.

### **2.3.3 Блок Components**

Данный блок содержит в себе описание внешнего вида компонентов, а также логику, представленную в них.

### **2.3.4 Блок Pages**

Данный блок содержит в себе описание внешнего вида страниц, а также логику, представленную в них. Ключевым отличием от блока компонентов является то, что каждый элемент блока страниц – это страница, на которую пользователь может зайти. Страницы содержат в себе элементы блока компоненты, но компоненты не содержат в себе страницы.

### **2.3.5 Блок Router**

Данный блок отвечает за маршрутизацию при работе с проектом, проверяет возможность зайти на страницу, определяет ссылки, по которым расположены страницы.

### **2.3.6 Блок Store**

Данный блок содержит в себе хранилища для данных, используемых страницами, методы для изменения и получения данных, а также методы, вызывающие методы api. Методы из этого блока применяются страницами и компонентами.

# ЗАКЛЮЧЕНИЕ

Результатом данной курсовой работы является веб-приложение, обеспечивающее возможность работать с личными и групповыми проектами, способное к удобному расширению и интеграции с другими системами.

В ходе выполнения курсового проекта был создан продукт, обладающий достаточным для работы с ним функционалом, обеспечивающий корректную работу системы, защищающий её от некорректных действий пользователя.

Таким образом, все цели, поставленные при выполнении курсового проекта, были выполнены.