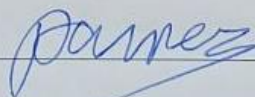


МИНОБРНАУКИ РОССИИ  
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ  
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ


## ОТЧЕТ ПО УЧЕБНОЙ ТЕХНОЛОГИЧЕСКОЙ (ПРОЕКТНО-ТЕХНОЛОГИЧЕСКОЙ) ПРАКТИКЕ

Студент  
гр. БИН-21-01



П.С. Резниченко

Руководитель,  
Доцент,  
канд. физ. мат. наук



А.В. Тюевев

Владивосток 2024

МИНОБРНАУКИ РОССИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВВГУ»)  
Институт информационных технологий и анализа данных  
Кафедра информационных технологий и систем

Индивидуальное задание  
на учебную технологическую (проектно-технологическую) практику

Студенту гр. БИН-21 Резниченко Павлу Сергеевичу

**1 Характеристика предприятия (организации), являющегося объектом дальнейшей автоматизации** (полное наименование, краткая информация, основные виды деятельности компании, миссия и основные бизнес-цели компании, номенклатура продукции или услуг, описание организационной структуры предприятия с описанием основных подразделений.)

**2 Состояние и стратегия развития информационных технологий в организации** (степень автоматизации процессов в подразделении компании, покрытие функциональных областей, ИТ-архитектура, определение уровня зрелости ИТ процессов по стандарту СММІ. Наличие в компании программно-аппаратных комплексов, технических устройств, корпоративных информационных систем и других ИС для эффективного управления предприятием.)

**3 Обоснование выбора технологии проектирования на основе анализа существующих разработок**


**4 Техническое задание на разработку информационной системы по следующей структуре:**

- 1) Общие сведения об информационной системе
- 2) Цели и задачи проекта автоматизации
- 3) Объект автоматизации
- 4) Требования к информационной системе
- 5) Этапы, сроки и результаты выполнения
- 6) Порядок контроля и приемки информационной системы
- 7) Требования к документированию

**5 Срок сдачи отчета на кафедру: 11.07.2024**

Руководитель,  
Доцент, канд. физ. мат. наук

Задание получил:


Тювсеев А.В.

Резниченко П.С.

## Содержание

|   |    |
|---|----|
| Введение.....   | 4  |
| 1 Описание и анализ деятельности компании ООО «ДНС» ..... | 5  |
| 1.1 Описание ООО «ДНС» .....                              | 5  |
| 1.2 Анализ деятельности компании.....                     | 5  |
| 2 Состояние информационных технологий в ООО «ДНС» .....   | 8  |
| 2.1 Программное обеспечение отдела.....                   | 8  |
| 2.2 Используемые подходы при создании ПО .....            | 9  |
| 2.3 Оценка зрелости процессов.....                        | 10 |
| 4 Процесс разработки программного обеспечения .....       | 11 |
| 4.1 Техническое задание.....                              | 11 |
| 4.2 Проектирование решения для автоматизации.....         | 12 |
| 4.3 Написание кода .....                                  | 16 |
| Заключение .....  | 17 |
| Список использованных источников .....                    | 18 |

## Введение

В рамках учебной технологической практики требуется выполнить следующее:

- привести характеристику ООО «ДНС», в котором будет проводится последующая автоматизация;
- описать состояние и стратегия развития информационных технологий в организации на момент прохождения практики;
- создать прототип решения для автоматизации деятельности на предприятии.

## 1 Описание и анализ деятельности компании ООО «ДНС»

### 1.1 Описание ООО «ДНС»

ООО «ДНС Ритейл», являющаяся частью ООО «ДНС Групп», в основном занимается торговлей розничной телекоммуникационным оборудованием, включая розничную торговлю мобильными телефонами, в специализированных магазинах [1]. Также, данная компания занимается автоматизацией бизнес-процессов [2].

Основными ценностями компании являются:

– Для клиентов: «Работаем ради клиентов, строим долгосрочные отношения и ценим доверие. Экспертиза — наше всё. Стремимся к постоянному развитию. Открыты для предложений.»

– Для партнеров: «Честны с партнёрами, соблюдаем этику бизнеса. Уважаем чужие мнения и интересы. Выполняем обязательства и берём ответственность за свои решения. Нетерпимы к коррупции.»

– Для сотрудников: «Мы — одна команда. DNS — территория личной и коллективной самореализации. Здесь уважают мнения и интересы сотрудников, ценят свободу, смелость и ответственность.»

### 1.2 Анализ деятельности компании

Технологическая практика происходила в отделе по разработке средств автоматизации бизнес-процессов (рис 1.2.1).

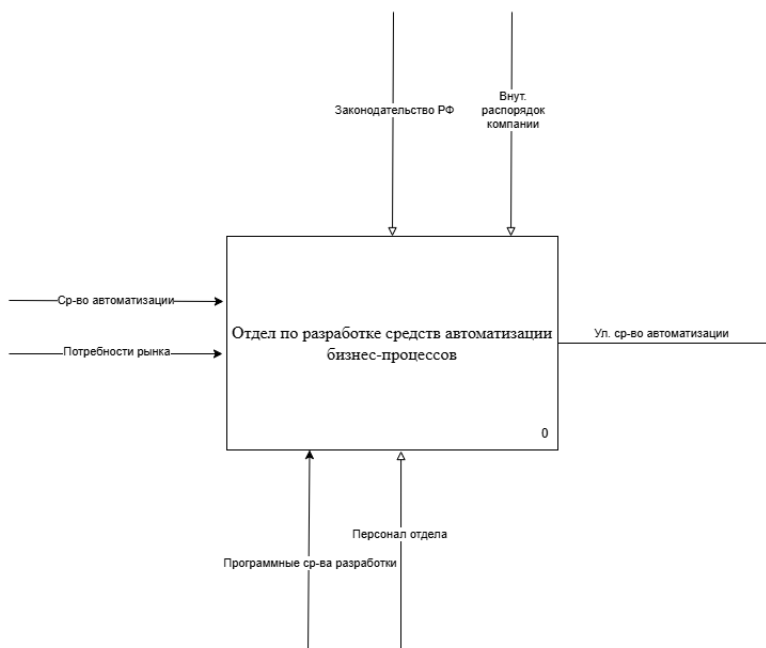


Рисунок 1.2.1 – Контекстный уровень отдела

В отделе существуют множество процессов (рис. 1.2.2).

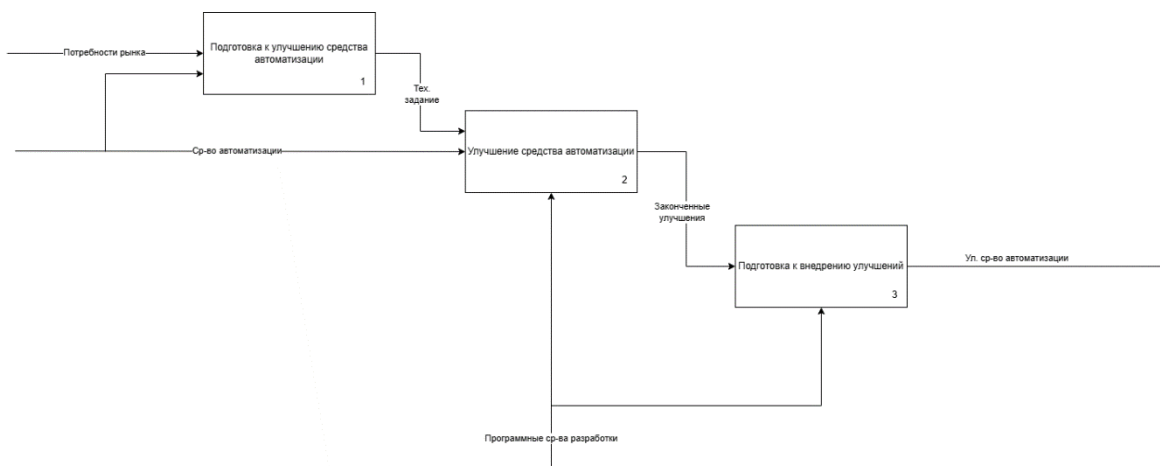


Рисунок 1.2.2 – Процессы верхнего уровня отдела

Всего в отделе существуют следующие процессы:

- подготовка к улучшению средства автоматизации;
- улучшение средства автоматизации;
- подготовка к внедрению улучшений.

Процесс «подготовка к улучшению средства автоматизации» можно разделить на подпроцессы «выявление будущих улучшений» и «формирование технического задания» (рис. 1.2.3).

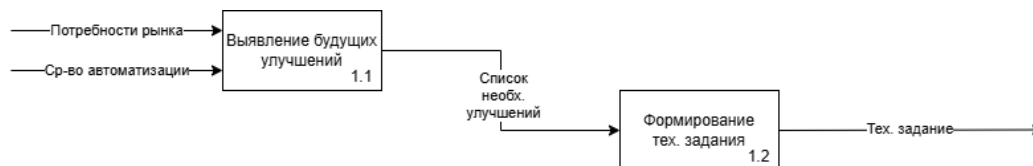


Рисунок 1.2.3 – Подпроцессы процесса «подготовка к улучшению средства автоматизации»

Сначала, исходя из потребностей на рынке, рассматривается текущее состояние средства автоматизации, после чего составляется список улучшений, которые необходимо внести в приложение. Исходя из данного списка, формируется техническое задание, которое направляется разработчикам.

Процесс «улучшение средства автоматизации» можно разделить на под процессы «оценка объема работ», «реализация улучшений» и «тестирование улучшений» (рис. 1.2.4).

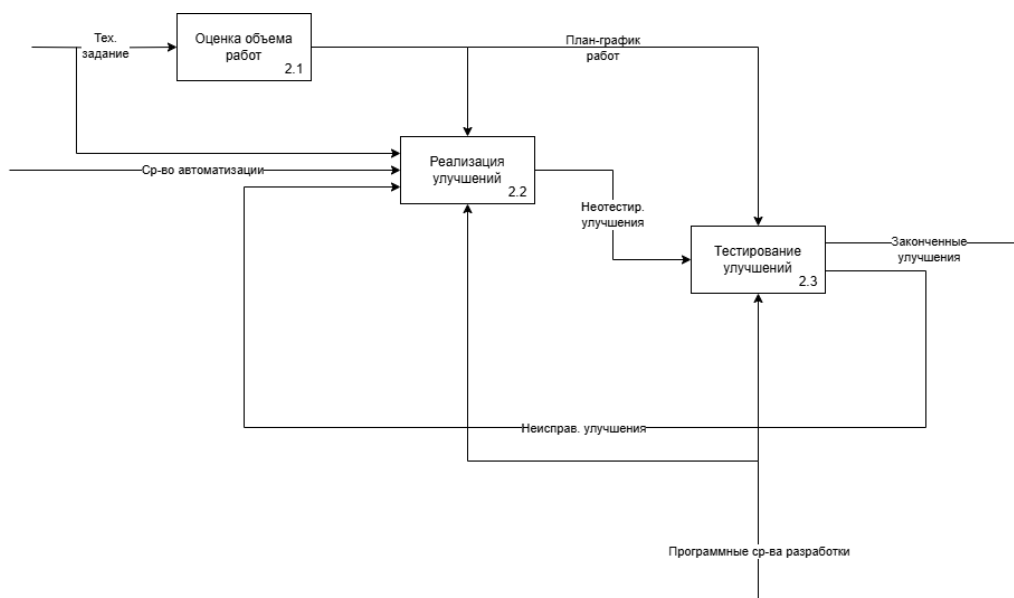


Рисунок 1.2.4 – Подпроцессы процесса «улучшение средства автоматизации»

Сначала, глава разработки (он же «тимлид»), исходя из ТЗ, составляет план-график работ, на который будут опираться разработчики при создании улучшений для средства автоматизации. Далее разработчики при помощи программных средств разработки создают необходимые улучшения, после чего отправляют тестировщикам на тестирование. Наконец, прошедшие улучшения отправляются на подготовку к внедрению, а непрошедшие – на доработку.

Процесс «подготовка к внедрению улучшений» представляет собой следующее: разработчики внедряют прошедшие тестирование улучшения в средство автоматизации, после чего получается улучшенное средство автоматизации, соответствующее ранее составленному ТЗ.

## 2 Состояние информационных технологий в ООО «ДНС»

### 2.1 Программное обеспечение отдела

На данный момент в отделе по разработке средств автоматизации используется следующее программное обеспечение:

– Git - это система контроля версий, позволяющая следить за изменениями в файлах проекта. Она предлагает функции для работы с ветвями, отслеживания истории изменений, отката к предыдущим версиям и поддерживает распределенную работу. [3] Git является основным инструментом для многих разработчиков программного обеспечения.

– ASP.NET - это серверный фреймворк для разработки веб-приложений, предназначенный для создания динамических веб-страниц [4]. Он был разработан Microsoft, чтобы позволить программистам строить динамические веб-сайты, приложения и сервисы. Название ASP.NET расшифровывается как Active Server Pages Network Enabled Technologies.

– PostgreSQL - это бесплатная система управления реляционными базами данных с открытым исходным кодом, которая акцентирует внимание на расширяемости и соответствии SQL [5]. Эта СУБД предлагает высокую мощность и широкую функциональность, делая ее одним из наиболее предпочтительных выборов для разработчиков и администраторов баз данных.

– React JS - это бесплатная и открытая JavaScript-библиотека для создания пользовательских интерфейсов на основе компонентов. [6] Он поддерживается Meta и сообществом отдельных разработчиков и компаний. React может использоваться для разработки одностраничных, мобильных или серверных приложений при помощи фреймворков, таких как Next.js.

– Drone CI - это платформа для непрерывной интеграции, предназначенная для загруженных команд разработчиков [7]. Данный инструмент автоматизирует тестирование программного обеспечения и его доставку, упрощает процесс разработки и повышает производительность команды.

– TeamCity - это сервер управления сборкой и система непрерывной интеграции от компании JetBrains [8]. Он предназначен для автоматизации процесса сборки и тестирования программного обеспечения. TeamCity предоставляет быструю обратную связь на каждое изменение кода, снижает проблемы интеграции кода и улучшает процесс разработки.

Код с использованием ПО, описанного выше, создается в Visual Studio Code – кроссплатформенной интегрированной среде разработки (IDE), разработанной компанией Microsoft [9]. Она предназначена для написания и отладки кода, поддерживает множество



языков программирования и обладает разнообразными функциями, способствующие удобству разработки программного обеспечения. Visual Studio Code также позволяет расширять свои возможности с помощью различных плагинов и расширений, что делает его популярным выбором среди разработчиков.

## 2.2 Используемые подходы при создании ПО

На предприятии используются следующие подходы при написании программного обеспечения:

- SOLID - это набор рекомендаций для разработки программного обеспечения, которые вместе обеспечивают более понятный, гибкий и поддерживаемый код [10]. В него входят: Принцип единственной ответственности (у класса должна быть только одна причина для изменения), Принцип открытости/закрытости (программные сущности должны быть открыты для расширения, но закрыты для изменения), принцип подстановки Барбары Лисков (Функции, которые используют базовый тип, должны иметь возможность использовать его подтипы, не зная об этом), Принцип разделения интерфейса (Много специфических интерфейсов лучше, чем один универсальный), Принцип инверсии зависимостей (Зависимости внутри системы строятся на основе абстракций, а не деталей);

- Dependency injection (внедрение зависимостей) - это концепция программирования, используемая для уменьшения уровня связности в программах, что улучшает модульность и эффективность тестирования [11]. В ситуации без DI, компоненты и услуги, зависящие от других классов, создаются внутри класса-пользователя, что приводит к жесткой связности и затрудняет масштабирование, поддержку и тестирование. Это обеспечивает более гибкую архитектуру, упрощает тестирование и позволяет легко вносить изменения в программу.

- KISS (Keep It Simple, Stupid) - это принцип разработки, который подразумевает, что любая система должна быть максимально простой и понятной [12]. Сложность должна быть минимальной, чтобы уменьшить вероятность ошибок, упростить сопровождение и облегчить внесение изменений, что обеспечивает эффективность и удобство использования.

- Непрерывная интеграция/непрерывная доставка (CI/CD) - это практика в сфере разработки программного обеспечения, которая направлена на автоматизацию процессов сборки, тестирования и развертывания приложений [13]. Непрерывная интеграция включает в себя частые автоматические сборки и проверки кода, а непрерывная доставка обеспечивает автоматическое развертывание готового приложения в рабочую среду. Эти практики позволяют команде быстрее и надежнее разрабатывать и доставлять программное обеспечение, улучшая качество и ускоряя процесс разработки.

### 2.3 Оценка зрелости процессов

Для оценки зрелости процессов в отделе по разработке средств автоматизации бизнес-процессов было принято решение использовать модель СММІ [14]. Данная модель является набором методологий совершенствования процессов в организациях разных размеров и видов деятельности. СММІ содержит набор рекомендаций в виде практик, реализация которых, по мнению разработчиков модели, позволяет реализовать цели, необходимые для полной реализации определенных областей деятельности.

Опираясь на данную модель, можно оценить процессы внутри отдела на пятый уровень зрелости - уровень постоянного улучшения (оптимизации) процессов. На данном этапе мы имеем точные характеристики оценки эффективности бизнес процессов, что позволяет нам постоянно и эффективно улучшать бизнес процессы путем развития существующих методов и техник и внедрения новых.

## 4 Процесс разработки программного обеспечения

### 4.1 Техническое задание

Необходимо разработать инструмент для отслеживания точного срока выполнения задачи.

Инструмент должен помогать пользователю точно отслеживать время, которое было потрачено на выполнение задачи. Процесс работы с инструментом следующий:

- Создание задачи: задача состоит из названия и описания того, что необходимо сделать;

- Внесение плановой оценки выполнения: например, пользователь считает, что для задачи ему необходимо 5 часов, тогда он вносит это в отдельную ячейку. Внесение этой информации не обязательно.

- Старт таймера. После создания задачи, в строке с названием появляется кнопка play, нажатие на которую провоцирует запуск таймера. После нажатия, кнопка play переходит в кнопку pause. Кнопка pause останавливает течение таймера и переводит кнопку с состояние play.

- Выполнение задачи. Каждая строка с задачей содержит кнопку Done, что означает остановку таймера и выполнение задачи. Задача уходит из зоны видимости пользователя, но не удаляется из базы.

- Остановка таймера записывает время, которое прошло от старта таймера. Это необходимо, чтобы оценивать потраченное время не только в общем, но и по промежуткам. Например, пользователь начинает работать над задачей, нажимает кнопку play. Перед обедом он останавливает задачу, нажимая pause. Результат, например, 2ч47м записывается в историю задачи. Так пользователь узнает, что он работает 2ч47м до обеда. Историю нужно вывести в отдельном окне с описанием задачи.

Если таймер отсчитывает времени больше, чем плановая оценка, то строка с задачей демонстрирует это. Например, цифры таймера можно подсвечивать красным цветом. Пример дизайна инструмента можно взять из популярного time and task трекера - clickUp или любого другого популярного трекера.

Реализация UI\UX остается полностью на усмотрение исполнителя.

Технические требования к прототипу следующие:

- Необходимо реализовать клиент, используя стандартные frontEnd технологии. Серверное приложение должно быть реализовано как API.

- Использовать необходимо .net и язык программирования C#.

- СУБД для хранения и управления данными - PostgreSQL. Взаимодействие с серверного приложения с базой на усмотрение исполнителя.

- Публикация на удаленный сервер не обязательна. Достаточно, чтобы приложения запускались на локальной машине исполнителя и проверяющего без лишних манипуляций. Использование docker-compose для серверного приложения и БД будет плюсом. Необходимо подготовить тестовые данные, чтобы проверяющий не заполнял хранилище самостоятельно.

Условия удовлетворенности протипом:

- Сайт должен быть интуитивно понятным и соответствовать бизнес требованиям, которые описаны в разделе “описание”

- Если остановить таймер на одной вкладке\окне, то таймер должен остановиться и в остальных вкладках\окнах, учитывая ring сети. Например, таймер на странице А был остановлен в 15:45:16, но, учитывая ring сети и цикл проверки статуса, информация об остановке дошла до страницы В в 15:45:25. В таком случае, на странице В таймер должен показывать 15:45:16, т.е. проигнорировать 9 сек на синхронизацию

- Тоже самое, что описано в п.4, для старта таймера. Если таймер был запущен на странице А, то на странице В таймер запускается с учетом потерянных секунд на синхронизацию. Например, сразу с 5ой секунды.

- Таймер не перестает тикать, даже если полностью закрыть браузер.

## 4.2 Проектирование решения для автоматизации

Во всех процессах предприятия стоит следующая система учета времени работы сотрудника: работник использует пропуск, чтобы войти в рабочие помещения или выйти из них, тем самым в базе данных предприятия делается запись с указанием даты и времени входа/выхода. Данная система надежна и проста в реализации, однако, не позволяет четко видеть какими задачами занимался сотрудник на работе и сколько времени он потратил.

Именно для решения данного недостатка была начата разработка текущего ПО.

Для ее реализации были использованы все программное обеспечение и подходы, которые используются внутри самого предприятия, а именно:

- Git;
- ASP.NET;
- React JS;
- PostgreSQL;
- Visual Studio Code.

Данное решение обеспечивает максимальную совместимость с инфраструктурой предприятия и позволит обеспечивать работоспособность системы в будущем средствами сотрудников.

Далее была разработана архитектура базы данных, где будут храниться данные (рис 4.2.1).

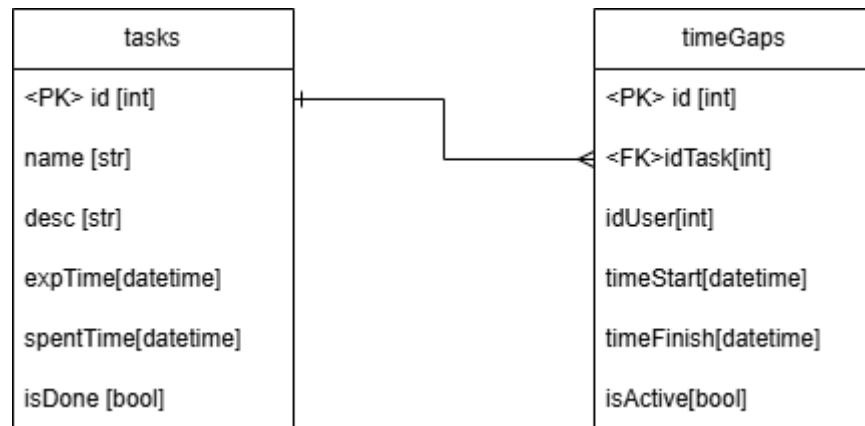


Рисунок 4.2.1 – Архитектура базы данных

В базе данных были следующие таблицы:

- «tasks» - таблица, где перечислены задачи.
- «timeGaps» - таблица, где указаны промежутки времени выполнения заданий.

В таблице «tasks» были следующие значения:

- id - номер задания
- name - название задание
- desc - описание задания
- expTime - ожидаемое время выполнения
- spentTime - потраченное время выполнения
- isDone - выполнено ли задание

В таблице «timeGaps» были следующие значения:

- id - номер записи
- idTask - номер выполняемого задания
- idUser - номер исполнителя задания
- timeStart - время и дата начала промежутка
- timeFinish - время и дата конца промежутка
- isActive - активирован ли на этом промежутке трекер

Разрабатываемое ПО будет создано в формате web-приложения [12], то есть программа работает на сервере и взаимодействует с конечными пользователями через сеть

(обычно — Интернет) с помощью браузера. Веб-приложения могут иметь различный функционал — от простых страниц с информацией до сложных интернет-магазинов или сервисов для работы с документами. Одними из главных преимуществ веб-приложений являются их доступность с любого устройства с браузером и подключением к сети, а также отсутствие необходимости устанавливать дополнительное программное обеспечение.

Далее был разработан дизайн приложения (рис 4.2.2). За основу был взят онный с популярного аналога «ClickUp» [13].

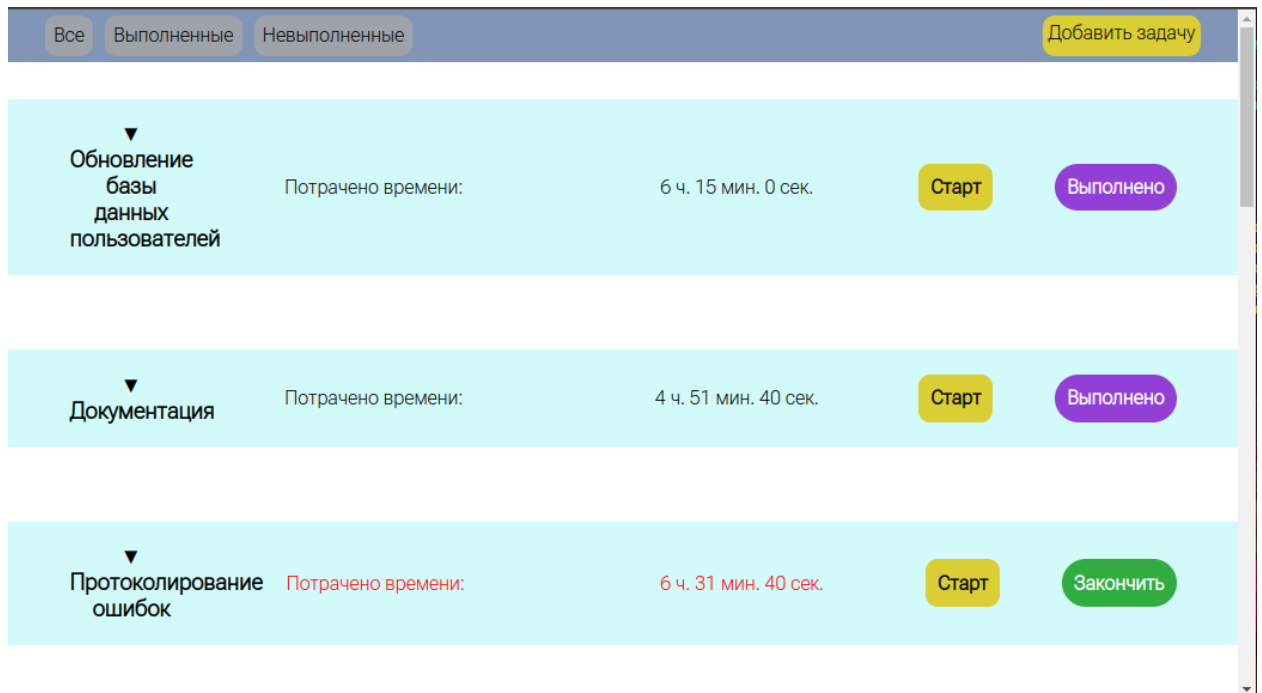


Рисунок 4.2.2 – Главная страница

В верхней части левой части находятся критерии выборки: показывать все задачи или только выполненные и невыполненные. В основной части генерируются карточки заданий, где указано ее название, сколько пользователь потратил времени, а также кнопки «Старт», которые позволяют запустить таймер, и кнопка «Закончить», которая при нажатии меняет статус у задачи, а саму кнопку превращает в пункт «Выполнено». Таймер в карточке меняет свой цвет с черного на красный, если затраченное время превышает ожидаемое.

При нажатии на название, карточка раскрывается, позволяя пользователю взглянуть на полное описание задачи (рис 4.2.3).



Рисунок 4.2.3 – Полная карточка задачи

К карточке добавляется два новых блока. Первый – более подробное описание задачи: описание и предполагаемое время, если они были указаны ранее. Второй – промежутки работы, которые представляют собой две даты со временем, указанные в формате UTC – дата начала и дата конца промежутка, в котором был активирован, а затем деактивирован таймер на данной задаче.

В верхнем правом углу приложения расположилась кнопка «Добавить задачу», которая перемещает пользователя на отдельную страницу (рис. 4.2.4).

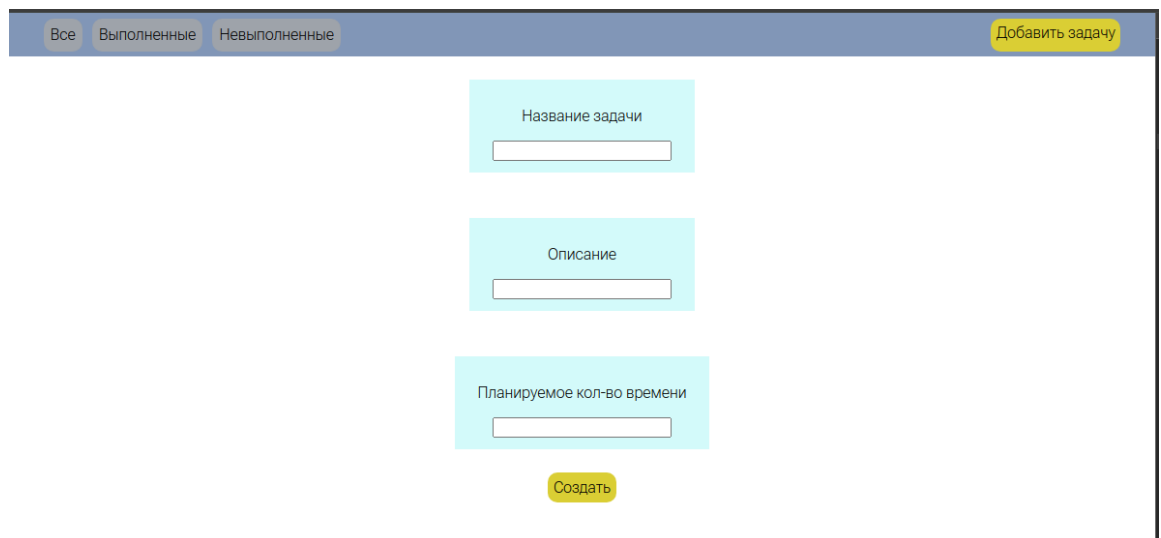


Рисунок 4.2.4 – Страница для создания задач

На данной странице расположена форма, в которой пользователь должен для создания задачи указать ее название, после чего нажать на кнопку «Создать». Опционально, пользователь может указать описание задачи и сколько времени потребуется на ее выполнение в соответствующих полях.

### 4.3 Написание кода

Сначала были созданы страницы для отображения заданий (выполненных, невыполненных и всех) и для добавления новых задач в БД. Далее был написан код для обеспечения работы страниц с базой данных, а именно:

- функция для создания нового задания
- функция для получения всех задач
- функция для получения только выполненных задач
- функция для получения только невыполненных задач
- функция для добавления начала промежутка времени
- функция для добавления конца промежутка времени

Наконец, была налажена связь между веб-страницами и приложением для манипулирования данными из базы данных. Указанные выше функции вызываются при помощи запросов, оформленных с использованием Rest API.

REST API (Representational State Transfer) - это архитектурный стиль для взаимодействия компонентов распределенной системы [17]. Он используется для построения веб-сервисов, которые позволяют управлять и обмениваться данными между различными приложениями через интернет. REST API определяет набор ограничений для архитектуры распределенной системы, включая принципы работы с ресурсами и передачи состояния.



## Заключение

В рамках учебной технологической практики было выполнено следующее:

- была приведена характеристика ООО «ДНС», в котором будет проводится последующая автоматизация;
- было описано состояние и стратегия развития информационных технологий в организации на момент прохождения практики;
- был создан прототип решения для автоматизации деятельности на предприятии.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Виды деятельности ООО «ДНС Ритейл» Текст: электронный // Rusprofile: [сайт] URL: <https://www.rusprofile.ru/okved/4960173#other> (дата обращения 04.07.2024);
2. DNS Group: [сайт] URL: <https://dnsgroup.ru/ru/projects/retail/dns-shop> (дата обращения 04.07.2024);
3. Git: [сайт] URL: <https://git-scm.com/> (дата обращения 04.07.2024);
4. Обзор ASP.NET Текст: электронный // Microsoft Learn [сайт] – URL: <https://learn.microsoft.com/ru-ru/aspnet/overview> (дата обращения 04.07.2024);
5. PostgreSQL: [сайт] URL: <https://www.postgresql.org/> (дата обращения 04.07.2024);
6. React: [сайт] URL: <https://react.dev/> (дата обращения 04.07.2024);
7. DroneCI: [сайт] URL: <https://www.drone.io/> (дата обращения 04.07.2024);
8. TeamCity Текст: электронный // JetBrains [сайт] URL: <https://www.jetbrains.com/ru-ru/teamcity/> (дата обращения 04.07.2024);
9. Visual Studio Code: [сайт] URL: <https://code.visualstudio.com/> (дата обращения 04.07.2024);
10. Принципы SOLID, о которых должен знать каждый разработчик Текст: электронный // Хабр: [сайт] URL: <https://habr.com/ru/companies/ruvds/articles/426413/> (дата обращения 04.07.2024);
11. Dependency injection Текст: электронный // Хабр: [сайт] URL: <https://habr.com/ru/articles/350068/> (дата обращения 04.07.2024);
12. KISS Principle in Software Development Текст: электронный // Geeks for Geeks [сайт] URL: <https://www.geeksforgeeks.org/kiss-principle-in-software-development/?ysclid=lxvb6gxur5905458526> (дата обращения 04.07.2024);
13. What is CI/CD? Текст: электронный // Geeks for Geeks [сайт] URL: <https://www.geeksforgeeks.org/what-is-ci-cd/?ysclid=ly3wiqbf1e551382642> (дата обращения 04.07.2024);
14. Модель СММІ Текст: электронный // Хабр: [сайт] URL: <https://habr.com/ru/articles/79130/> (дата обращения 04.07.2024);
15. What is Web App? ↯ Текст: электронный // Geeks for Geeks [сайт] ↯ URL: <https://www.geeksforgeeks.org/what-is-web-app/?ysclid=ly2f4ri6h3199186792> (дата обращения 04.07.2024);
16. ClickUp: [сайт] URL: [https://clickup.com/?fp\\_ref=c90oc](https://clickup.com/?fp_ref=c90oc) (дата обращения 04.07.2024);

17. REST, что же ты такое? Понятное введение в технологию для ИТ-аналитиков  
Текст: электронный // Хабр: [сайт] URL: <https://habr.com/ru/articles/590679/> (дата обращения 04.07.2024).