МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Специальность 1-40 05 01 03 «Информационные системы и технологии»

Специализация «Издательско-полиграфический комплекс»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ**

по дисциплине «Программирование в Internet»

Тема: Web-приложение для учета рабочего времени

Исполнитель

|  |  |  |
| --- | --- | --- |
| cтудент 4 курса группы 1    Руководитель |  | \_\_\_\_\_\_\_\_\_\_\_\_ И.А. Парибок  подпись, дата |
| ассистент |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.С. Кантарович  подпись, дата |
| Курсовой проект защищен с оценкой |  | \_\_\_\_\_\_\_\_\_\_\_\_ |
| Руководитель \_\_\_\_\_\_\_\_\_\_ |  | В.С. Кантарович |

подпись

Минск 2023

Оглавление

[Оглавление 5](#_Toc149554929)

[Введение 6](#_Toc149554930)

[1. Постановка задачи 7](#_Toc149554931)

[1.1 Обзор аналогов 7](#_Toc149554932)

[1.1.1. Веб приложение «Toggl» 7](#_Toc149554933)

[1.1.2. Веб приложение «Harvest» 8](#_Toc149554934)

[1.1.3. Веб приложение «Hubstaff» 9](#_Toc149554935)

[1.2. Актуальность задачи 11](#_Toc149554936)

[1.3. Постановка целей 12](#_Toc149554937)

[2. Разработка 13](#_Toc149554938)

[2.1. Разработка модели базы данных 13](#_Toc149554939)

[2.2. Диаграмма вариантов использования 16](#_Toc149554940)

[2.3. Проектирование основных алгоритмов 18](#_Toc149554941)

Введение

Веб-приложение для учета рабочего времени разрабатывается с целью упростить и автоматизировать процесс отслеживания рабочего времени. Оно предоставляет возможности регистрации пользователей, управления пользователями и проектами, отслеживания времени работы и анализа статистики. Это приложение позволит организациям более эффективно управлять рабочим процессом и оптимизировать использование ресурсов.

Для хранения данных веб-приложения будет использоваться реляционная модель базы данных. Реляционная модель основана на использовании таблиц, столбцов и связей между ними. Каждая таблица представляет собой совокупность записей с однотипной структурой. Записи в таблицах связаны между собой с помощью ключей и отношений.

Использование реляционной модели базы данных позволяет эффективно хранить и обрабатывать данные. Предоставляет возможность выполнять запросы для получения информации из нескольких таблиц одновременно, выполнять фильтрацию, сортировку и агрегацию данных. Это обеспечивает гибкость и мощные возможности анализа данных, необходимые для учета рабочего времени и отслеживания проектов.

Для создания веб-приложения выбраны JavaScript и платформа Node.js. JavaScript является одним из самых популярных языков программирования, который широко используется для разработки клиентской и серверной частей веб-приложений. Он обладает большим сообществом разработчиков, обширной экосистемой инструментов и библиотек, что облегчает разработку и поддержку приложения.

Платформа Node.js предоставляет возможность запускать JavaScript на сервере, что позволяет разрабатывать эффективные и масштабируемые веб-приложения. Она основана на событийно-ориентированной и неблокирующей архитектуре, что способствует высокой производительности и отзывчивости приложения при обработке большого количества запросов.

Выбор JavaScript и Node.js обеспечивает единообразие языка программирования на клиентской и серверной сторонах, что упрощает разработку и поддержку кода. Кроме того, наличие богатой экосистемы инструментов и библиотек для JavaScript и Node.js позволит эффективно реализовать требуемую функциональность и обеспечить безопасность и стабильность приложения.

В качестве инструмента для разработки интерфейса выбран фреймворк React. React — это популярная JavaScript-библиотека, которая позволяет разрабатывать эффективные и масштабируемые пользовательские интерфейсы. Она основана на компонентной модели, что позволяет создавать переиспользуемые компоненты и эффективно управлять состоянием интерфейса.

Постановка задачи

## Обзор аналогов

* + 1. Веб приложение «Toggl»

Первым аналогом для обзора было выбрано веб-приложение «Toggl» (рис. 1.1).

Стоимость: бесплатно до 5 пользователей (ограниченные возможности); $10 в месяц за пользователя (стартовый); $20 в месяц за пользователя (премиум).

Платформа: веб-сервис.

Язык: английский.

Ссылка: https://track.toggl.com/

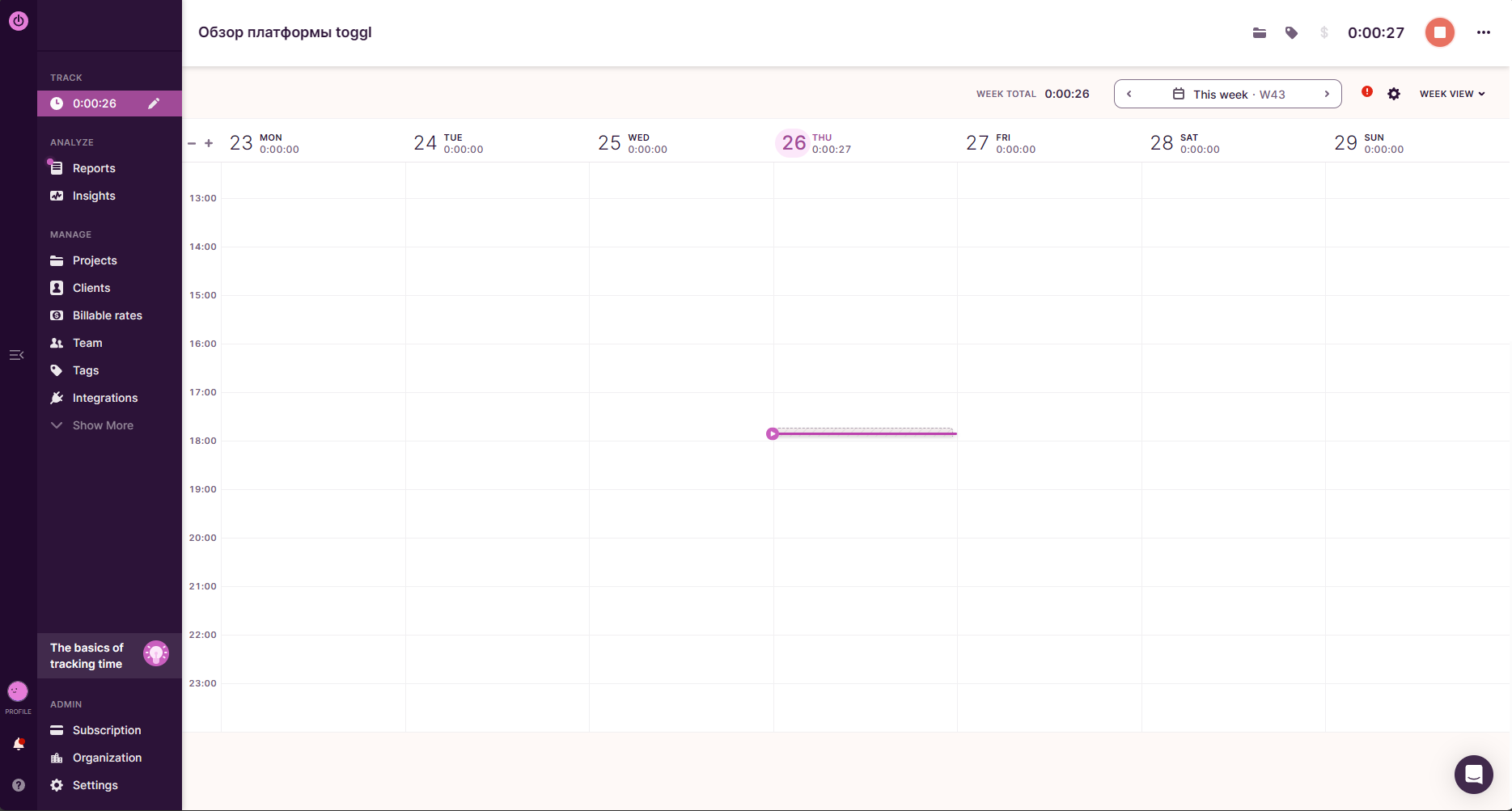


Рисунок 1.1 – Веб-сервис «Toggl»

Toggl — это популярное веб-приложение и инструмент учета времени, который позволяет пользователям отслеживать время, затраченное на различные задачи и проекты. Оно предоставляет функциональность для регистрации времени работы, анализа статистики и генерации отчетов.

Пользователи Toggl могут создавать таймеры для отслеживания времени, начинать и останавливать таймеры при переключении между задачами или проектами. Они также могут добавлять описания, теги и категории к каждому таймеру для более подробного анализа и организации данных.

Toggl предоставляет пользовательский интерфейс, который обеспечивает удобство использования и навигации. Он также предлагает функциональность для группировки задач и проектов, создания отчетов о затраченном времени, анализа производительности и мониторинга временных трендов.

Преимущества:

* Красивый UX/UI дизайн, интуитивный интерфейс.
* Соотношение цена-качество.
* Неограниченное количество клиентов, проектов и отчетов (даже для бесплатных пользователей).
* Подходит для управления личными и командными проектами.
* Управление клиентами.
* Создание ставок оплаты.
* Интеграция с большим количеством сервисов (100+).
* Кроссплатформенность, есть клиенты для iOS, Android, Mac, Windows, Linux.

Недостатки:

* Нет русского языка.

Вывод:

Доступный инструмент с современным дизайном для отслеживания времени с большим количеством интеграций и возможностей в бесплатной версии.

* + 1. Веб приложение «Harvest»

Вторым аналогом было выбрано веб-приложение «Harvest» (рис. 1.2).

Стоимость: бесплатно для одного пользователя, до двух проектов; платная 12 в месяц за пользователя.

Платформа: веб-сервис.

Язык: английский.

Ссылка: https://ikla.harvestapp.com/time

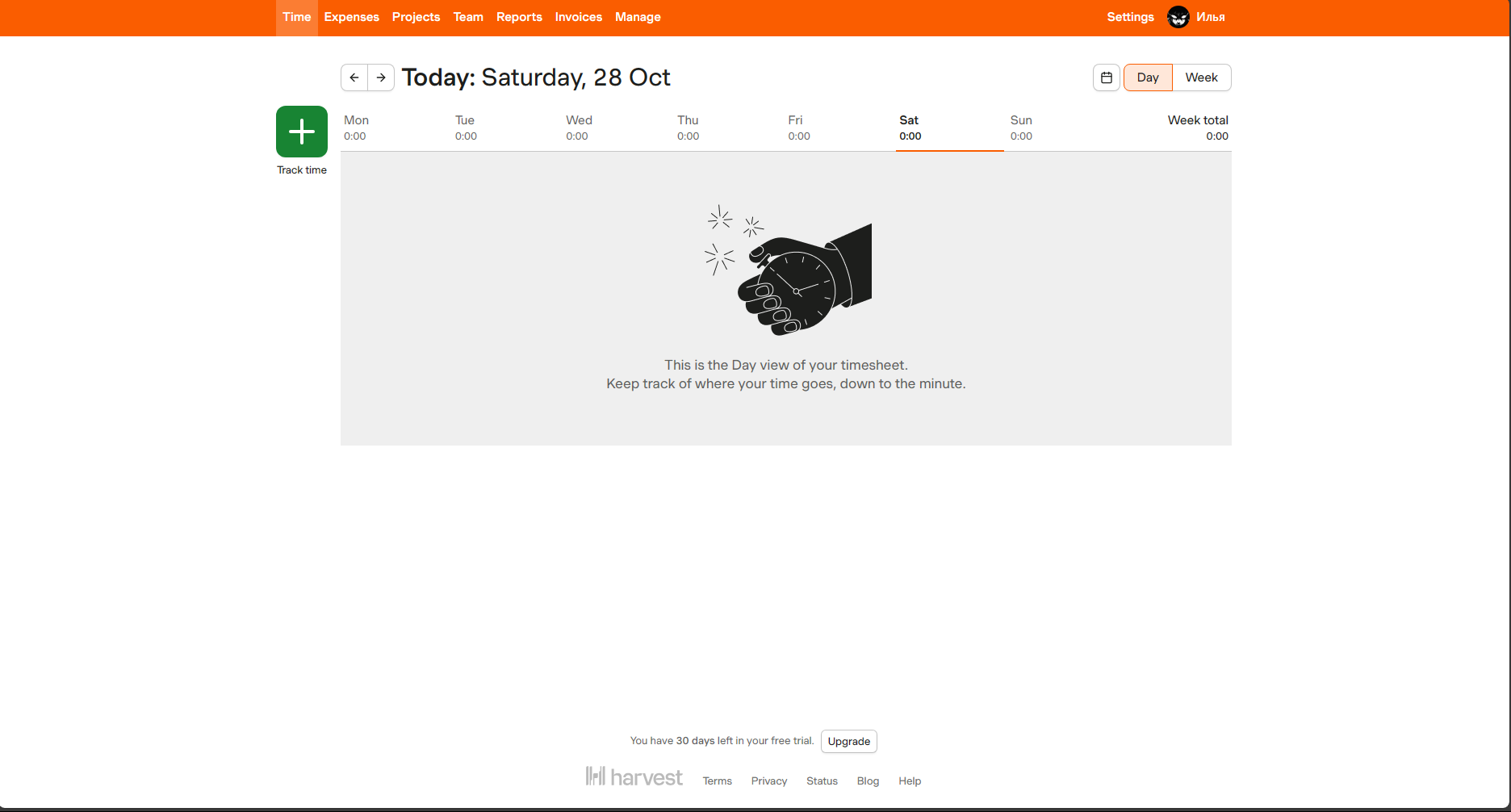


Рисунок 1.2– Веб-сервис «Harvest»

Harvest - еще один лидер в области учета рабочего времени. Как и в Toggl, здесь есть бесплатный тарифный план, но он ограничивается одним пользователем и двумя проектами. С другой стороны, вы получаете массу возможностей интеграции с системами управления проектами, бухгалтерского учета, CRM и другими инструментами.

Кроме того, есть мобильные приложения для учета времени для iOS и Android, а также расширения для Chrome и Safari. Можно также ограничиться настольной версией.

Преимущества:

* Красивый UX/UI дизайн, интуитивный интерфейс.
* Подходит для управления личными и командными проектами.
* Управление клиентами
* Автоматизированное выставление счетов
* Создание ставок оплаты
* Интеграция с большим количеством сервисов ,80+
* Кроссплатформенность, есть клиенты для iOS, Android, Mac, Windows.

Недостатки:

* Нет русского языка
* Бесплатная ограничивается одним пользователем и двумя проектами.

Вывод:

Приложение с современным интерфейсом и высокой функциональностью, имеет большое количество интеграций подробные отчеты.

* + 1. Веб приложение «Hubstaff»

Последним аналогом для обзора было выбрано веб-приложение «Hubstaff» (рис. 1.3).

Стоимость: бесплатно для одного пользователя (ограниченные возможности); 5$. в месяц для одного пользователя (базовые возможности).

Платформа: веб-сервис.

Язык: английский.

Ссылка: https://hubstaff.com/

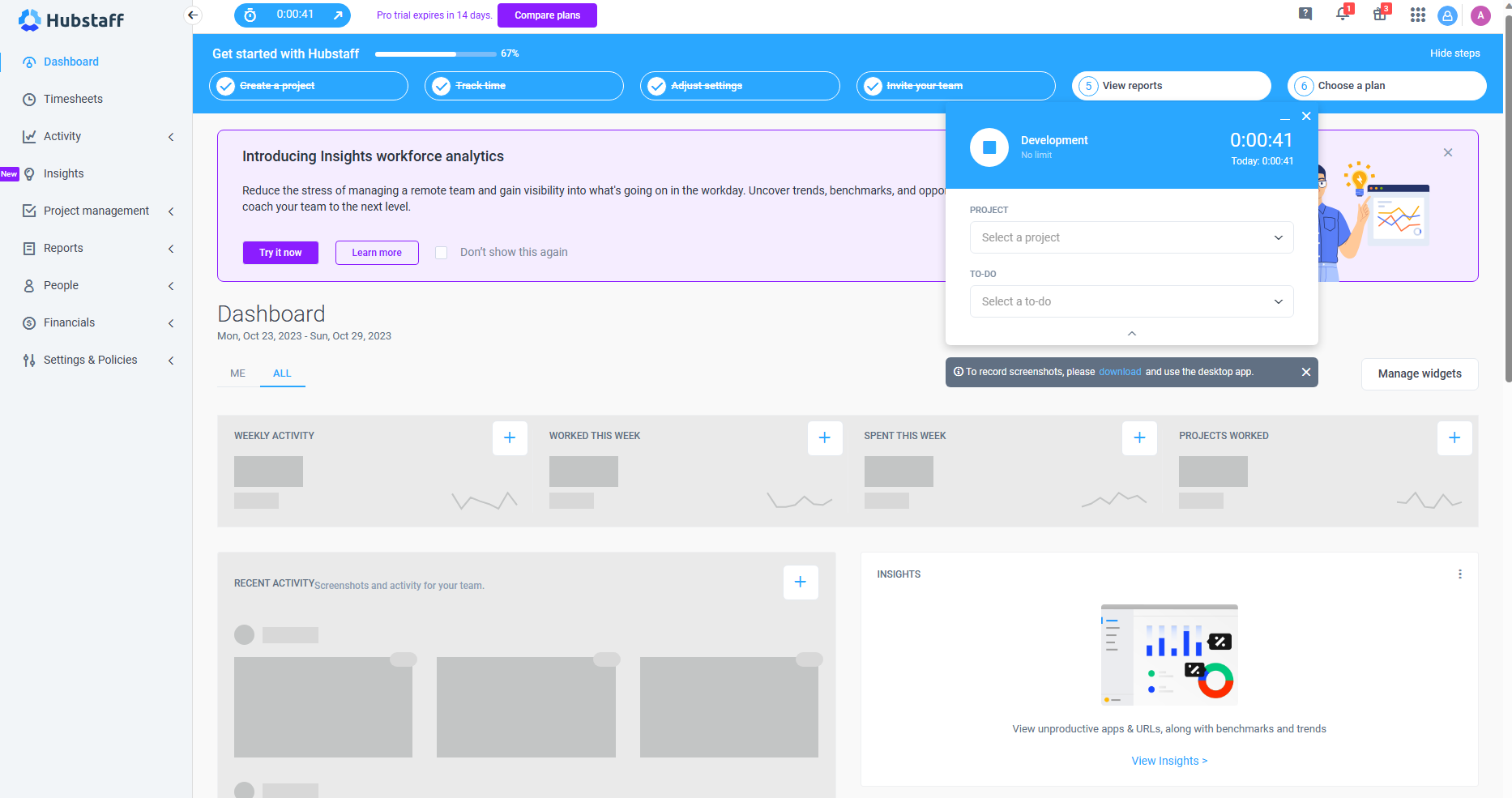


Рисунок 1.3 – Веб сервис «Hubstaff»

Hubstaff - приложение для учета рабочего времени, предназначенное для работодателей, чтобы отслеживать продуктивность работы членов команды.

Hubstaff помогает вам убедиться, что сотрудники рационально используют свое время.

Основная задача Hubstaff - убедиться в том, что команда раскрывают свой потенциал, по крайней мере, с точки зрения управления временем. Система обнаруживает, когда пользователь слишком долго бездействует (т.е. не работает), и выдает соответствующее сообщение, побуждая вернуться к работе или остановить таймер и отредактировать свою запись.

Есть возможность отслеживать по GPS команду. Hubstaff даже делает случайные скриншоты активности пользователей, а также отслеживает нажатия клавиш в приложении, чтобы вы могли видеть, кто активно использует (или не использует) инструмент.

Преимущества:

* Красивый UX/UI дизайн.
* Подходит для управления личными и командными проектами.
* GPS трекинг.
* Автоматизированное выставление счетов
* Создание ставок оплаты
* Подробные отчеты
* Кроссплатформенность, есть клиенты для iOS, Android, Mac, Windows, Linux

Недостатки

* Нет русского языка
* Сложный интерфейс

Вывод:приложение имеет широкий функционал и глубокую настройку. Отлично подходит для наблюдения и оптимизации работы команды.

* 1. Актуальность задачи

Разработка веб-приложения по отслеживанию времени остается актуальной и полезной, поскольку:

1. Позволяет учет времени и оплату труда: Веб-приложение позволяет пользователям точно отслеживать время, затраченное на задачи и проекты, что важно для учета рабочего времени и оплаты труда.

2. Увеличивает производительность и эффективность: Отслеживание времени помогает выявить, какие задачи занимают больше времени, что позволяет оптимизировать рабочий процесс и повысить производительность.

3. Управление проектами и задачами: Веб-приложение предоставляет инструменты для управления проектами, установления сроков, определения приоритетов и контроля выполнения задач.

4. Создание аналитики и отчетности: Веб-приложение предоставляет возможность анализировать данные о времени, создавать отчеты и диаграммы, которые помогают оценить эффективность работы и принять информированные решения.

В целом, разработка веб-приложения по отслеживанию времени актуальна, поскольку она помогает учету рабочего времени, повышает производительность и эффективность, облегчает управление проектами и создание аналитики для принятия информированных решений.

Выбор инструментов разработки

Для решения задачи для хранения данных и создания процедур было выбрано MongoDB.

MongoDB – это гибкая и масштабируемая документоориентированная база данных, которая хранит данные в формате документов (например, JSON). Она обеспечивает простоту использования и разработки, поддерживает обработку больших объемов данных и предоставляет мощные возможности запросов. MongoDB имеет широкую экосистему и поддержку со стороны сообщества.

В данном курсовом проекте разрабатывается веб-приложение для учета времени будет с использованием JavaScript, Node.js и React. JavaScript будет использоваться для создания динамического и интерактивного пользовательского интерфейса, а также для обработки данных на клиентской стороне. Node.js будет использоваться для разработки серверной части приложения, обработки запросов и взаимодействия с базой данных. React будет использоваться для построения компонентного пользовательского интерфейса, который будет обеспечивать удобную и эффективную работу с функциональностью приложения.

В процессе разработки данного курсового проекта будет использоваться интегрированная среда разработки (IDE) Visual Studio Code (VSCode). VSCode предоставляет мощные инструменты для разработки веб-приложений, включая поддержку языка JavaScript, интегрированную отладку, автозаполнение кода, систему управления версиями и множество плагинов для расширения возможностей IDE. Благодаря своей популярности и функциональности, VSCode является одним из наиболее предпочитаемых инструментов разработчиков при работе с JavaScript, Node.js и React.

* 1. Постановка целей

Целью данного проекта является разработка веб-приложения "Управление задачами и проектами", которое будет предоставлять следующий набор функциональности. Приложение должно поддерживать три роли: администратор, менеджер и пользователь. Пользователи должны иметь возможность регистрироваться и авторизовываться в системе. Администраторам предоставляется функционал управления пользователями и менеджерами, включая добавление, блокировку и редактирование учетных записей.

Приложение должно также предоставлять возможность отслеживания рабочего времени, позволяя пользователям запускать и останавливать таймер с указанием выполняемой работы и проекта. Менеджерам и пользователям должно быть доступно управление проектами и задачами, включая создание, редактирование и архивацию. Менеджеры должны иметь возможность распределять задачи между пользователями.

Важным функционалом приложения является показ статистики и генерация отчетов. Менеджеры и пользователи должны иметь возможность просматривать статистику и генерировать отчеты о затраченном времени на различные задачи и проекты.

Для достижения поставленных целей необходимо разработать централизованную базу данных и заполнить ее данными. Требуется создать веб-приложение с клиент-серверной архитектурой, обеспечивающее взаимодействие с базой данных. Необходимо реализовать алгоритмы авторизации и регистрации пользователей, настроить параметры работы страниц и разграничить права доступа. Также требуется добавить функционал, соответствующий целям каждой веб-страницы, разработать алгоритмы обработки ошибок и сбоев, создать привлекательный интерфейс и провести тестирование с последующим исправлением ошибок.

В результате выполнения поставленных целей и задач планируется достичь разработки функционального веб-приложения "Управление задачами и проектами".

Разработка

* 1. Разработка модели базы данных

Целью данной базы данных является управление проектами и задачами в организации. База данных позволяет хранить информацию о компаниях, пользователях, проектах, задачах, командах, а также отслеживать время выполнения задач и историю действий пользователей. Схема таблицы представлена на рисунке 2.1.

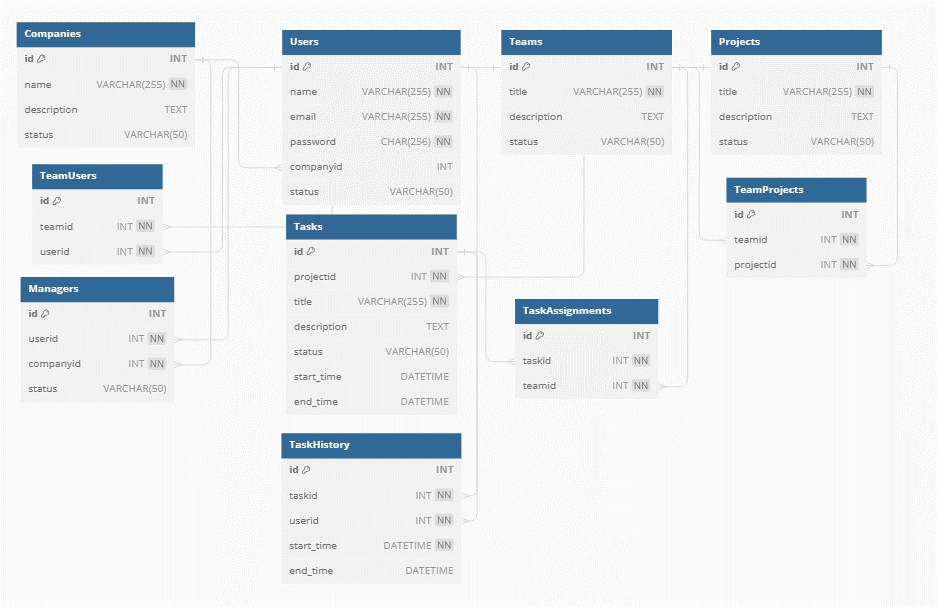


Рисунок 2.1 – Схема базы данных

Приведенная схема базы данных имеет следующую структуру:

Таблица "Companies" представляет информацию о компаниях:

- id: Уникальный идентификатор компании.

- name: Название компании.

- description: Описание компании.

- status: Статус компании.

Таблица "Users" содержит информацию о пользователе:

- id: Уникальный идентификатор пользователя.

- name: Имя пользователя.

- email: Адрес электронной почты пользователя.

- password: Хэш пароля пользователя.

- companyid: Идентификатор компании, к которой принадлежит пользователь.

- status: Статус пользователя.

Таблица "Teams" хранит информацию о командах:

- id: Уникальный идентификатор команды.

- title: Название команды.

- description: Описание команды.

- status: Статус команды.

Таблица "Projects" содержит информацию о проектах:

- id: Уникальный идентификатор проекта.

- title: Название проекта.

- description: Описание проекта.

- status: Статус проекта.

Таблица "TeamUsers" устанавливает связь между командами и пользователями:

- id: Уникальный идентификатор записи.

- teamid: Идентификатор команды.

- userid: Идентификатор пользователя.

Таблица "Tasks" хранит информацию о задачах:

- id: Уникальный идентификатор задачи.

- projectid: Идентификатор проекта, к которому относится задача.

- title: Название задачи.

- description: Описание задачи.

- status: Статус задачи.

- start\_time: Время начала выполнения задачи.

- end\_time: Время окончания выполнения задачи.

Таблица "TaskAssignments" устанавливает связь между задачами и командами:

- id: Уникальный идентификатор записи.

- taskid: Идентификатор задачи.

- teamid: Идентификатор команды.

Таблица "TeamProjects" устанавливает связь между командами и проектами:

- id: Уникальный идентификатор записи.

- teamid: Идентификатор команды.

- projectid: Идентификатор проекта.

Таблица "Managers" содержит информацию о менеджерах:

- id: Уникальный идентификатор менеджера.

- userid: Идентификатор пользователя, являющегося менеджером.

- companyid: Идентификатор компании, к которой принадлежит менеджер.

- status: Статус менеджера.

Таблица "TaskHistory" хранит историю изменений задач:

- id: Уникальный идентификатор записи.

- taskid: Идентификатор задачи.

- userid: Идентификатор пользователя, выполнившего действие.

- start\_time: Время начала действия.

- end\_time: Время окончания действия.

Схема базы данных содержит следующие связи между таблицами:

1. "Companies" связана с "Users" через внешний ключ "companyid". Эта связь указывает, что каждый пользователь (запись в таблице "Users") принадлежит определенной компании (запись в таблице "Companies"). Это позволяет связывать пользователей с их компаниями и выполнять операции, такие как поиск пользователей, принадлежащих определенной компании.

2. "Teams" связана с "TeamUsers" через внешний ключ "teamid". Связь указывает, что каждый пользователь может быть частью одной или нескольких команд. Таблица "TeamUsers" содержит записи, которые связывают идентификаторы команд и пользователей, обозначая их принадлежность к определенным командам.

3. "Users" также связана с "TeamUsers" через внешний ключ "userid". Эта связь указывает, какие пользователи принадлежат к каким командам. Таблица "TeamUsers" содержит записи, которые связывают идентификаторы пользователей и команд.

4. "Projects" связана с "Tasks" через внешний ключ "projectid". Эта связь указывает, что каждая задача (запись в таблице "Tasks") относится к определенному проекту (запись в таблице "Projects"). Это позволяет организовать иерархию проектов и задач, а также выполнять операции, связанные с определенными проектами.

5. "Tasks" связана с "TaskAssignments" через внешний ключ "taskid". Связь определяет, какие задачи относятся к каким командам. Таблица "TaskAssignments" содержит записи, которые связывают идентификаторы задач и команд.

6. "Teams" также связана с "TaskAssignments" через внешний ключ "teamid". Связь показывает, какие команды относятся к каким задачам. Таблица "TaskAssignments" содержит записи, которые связывают идентификаторы команд и задач.

7. "Teams" связана с "TeamProjects" через внешний ключ "teamid". Связь указывает, какие команды относятся к каким проектам. Таблица "TeamProjects" содержит записи, которые связывают идентификаторы команд и проектов.

8. "Projects" связана с "TeamProjects" через внешний ключ "projectid". Эта связь определяет, какие проекты относятся к каким командам. Таблица "TeamProjects" содержит записи, которые связывают идентификаторы проектов и команд.

9. "Users" связана с "Managers" через внешний ключ "userid". Связь указывает, какие пользователи являются менеджерами. Таблица "Managers" содержит записи, которые связывают идентификаторы пользователей и компаний, а также указывают их статус в качестве менеджеров.

10. "Companies" связана с "Managers" через внешний ключ "companyid". Связь показывает, какие компании имеют менеджеров. Таблица "Managers" содержит записи, которые связывают идентификаторы компаний и пользователей, являющихся их менеджерами.

11. "Tasks" связана с "TaskHistory" через внешний ключ "taskid". Эта связь используется для отслеживания истории изменений задачи. Таблица "TaskHistory" содержит записи, которые связывают идентификаторы задач и информацию о действиях, выполненных над задачей, включая идентификатор пользователя, дату начала и окончания действия.

Все эти связи позволяют связывать данные из разных таблиц, создавая структуру базы данных, которая отражает взаимосвязи между компаниями, пользователями, командами, проектами и задачами. Например, связи между "Users" и "Companies" позволяют устанавливать принадлежность пользователей к определенной компании, а связи между "Teams" и "TeamUsers" позволяют формировать команды из пользователей. Эти связи обеспечивают целостность данных и позволяют выполнять операции, такие как поиск, фильтрация и агрегация данных в базе данных.

* 1. Диаграмма вариантов использования

Диаграмма вариантов использования позволяет представить все доступные варианты использования приложения и взаимодействия пользователей с различными ролями, такими как администратор, пользователь и менеджер.

Роль "Пользователь" предоставляет функциональность для обычных пользователей, позволяющие выполнять задачи, создавать личные проекты и отчетность.

Роль "Менеджер" предоставляет функциональность по созданию и делегированию проектов, команд и задач.

Роль "Администратор" предоставляет функциональность для администраторов менеджерами и проектами.

На диаграмме вариантов использования, обозначенной на рисунке 2.2, отображается основная логика взаимодействия с системой.

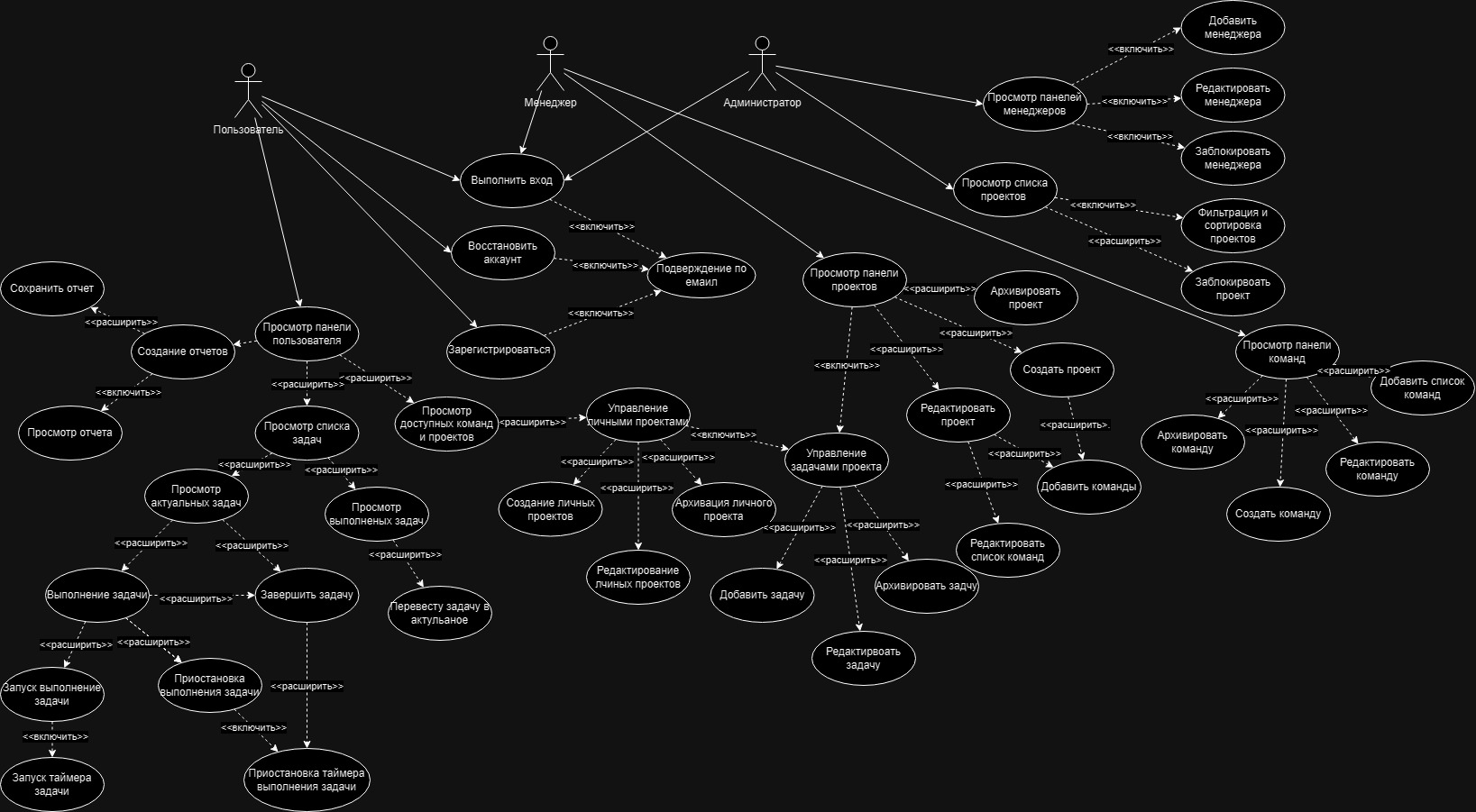


Рисунок 2.2 – Диаграмма использования

* 1. Проектирование основных алгоритмов

Блок-схемы алгоритмов являются визуальным инструментом, который помогает разработчикам понять, проектировать и отлаживать алгоритмы. Они представляют собой графическое представление последовательности шагов, условий и операций, которые выполняются в алгоритме.

Первым алгоритмом является создание команд. Данный алгоритм выполняется менеджером и представлен на рисунке 2.3



Рисунок 2.3 – Схема алгоритма создание команды

Вторым алгоритмом будет выполнение задачи пользователем(рис.2.4).

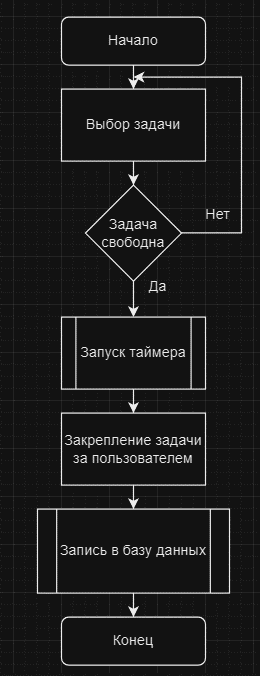


Рисунок 2.4 – Алгоритм выполнение задач