**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**на разработку приложения «Time Tracker»**

**Версия:** 1.0  
**Дата составления:** 22.02.2025  
**Разработчик:** Савицкий Степан, Заборовский Максим  
**Заказчик:** Савицкий Степан, Заборовский Максим

# СОДЕРЖАНИЕ

[**СОДЕРЖАНИЕ 2**](#_heading=h.v6s12mtxmxew)

[**1. ОБЩИЕ ПОЛОЖЕНИЯ 5**](#_heading=h.30j0zll)

[1.1. Назначение документа 5](#_heading=h.1fob9te)

[1.2. Цель разработки 5](#_heading=h.3znysh7)

[1.3. Область применения 5](#_heading=h.2et92p0)

[1.4. Нормативные ссылки 5](#_heading=h.tyjcwt)

[**2. ТЕРМИНОЛОГИЯ И ОПРЕДЕЛЕНИЯ 6**](#_heading=h.3dy6vkm)

[2.1. Основные термины 6](#_heading=h.1t3h5sf)

[2.2. Сокращения и условные обозначения 6](#_heading=h.4d34og8)

[**3. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ 7**](#_heading=h.2s8eyo1)

[3.1. Назначение приложения 7](#_heading=h.17dp8vu)

[3.2. Описание бизнес-процессов 7](#_heading=h.3rdcrjn)

[3.3. Сценарии использования 7](#_heading=h.26in1rg)

[3.4 Сравнение с аналогами на рынке 8](#_heading=h.jckyuj13abfo)

[**4. СТРУКТУРА И СОСТАВ СИСТЕМЫ 9**](#_heading=h.lnxbz9)

[4.1. Основные компоненты приложения 9](#_heading=h.35nkun2)

[4.2. Обзор функциональных страниц 9](#_heading=h.1ksv4uv)

[4.3. Взаимодействие модулей и компонентов 9](#_heading=h.44sinio)

[**5. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ 11**](#_heading=h.2jxsxqh)

[5.1. Главная страница 11](#_heading=h.z337ya)

[5.2. Страница истории задач 11](#_heading=h.3j2qqm3)

[5.3. Страница настроек 12](#_heading=h.1y810tw)

[5.4. Страница профиля 12](#_heading=h.4i7ojhp)

[5.5. Таймер (отдельное окно) 13](#_heading=h.2xcytpi)

[5.6. Управление задачами (CRUD-операции) 13](#_heading=h.1ci93xb)

[5.7. Управление аккаунтом и статистикой 14](#_heading=h.3whwml4)

[**6. ТРЕБОВАНИЯ К ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ 15**](#_heading=h.2bn6wsx)

[6.1. Общие принципы дизайна 15](#_heading=h.qsh70q)

[6.2. Описание элементов интерфейса по страницам 15](#_heading=h.3as4poj)

[6.3. Использование QSS для оформления 15](#_heading=h.1pxezwc)

[6.4. Анимационные эффекты 16](#_heading=h.49x2ik5)

[**7. АРХИТЕКТУРА И ТЕХНОЛОГИИ РЕАЛИЗАЦИИ 17**](#_heading=h.2p2csry)

[7.1. Выбор языка и инструментов разработки 17](#_heading=h.147n2zr)

[7.2. Используемые библиотеки и фреймворки 17](#_heading=h.3o7alnk)

[7.3. Интеграция с базой данных PostgreSQL 17](#_heading=h.23ckvvd)

[7.4. Механизмы авторизации и управления учётными записями 17](#_heading=h.ihv636)

[**8. НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ 19**](#_heading=h.32hioqz)

[8.1. Производительность и масштабируемость 19](#_heading=h.1hmsyys)

[8.2. Безопасность и защита данных 19](#_heading=h.41mghml)

[8.3. Надёжность и отказоустойчивость 19](#_heading=h.2grqrue)

[8.4. Кроссплатформенность и требования к ОС 19](#_heading=h.vx1227)

[**9. ТРЕБОВАНИЯ К БАЗЕ ДАННЫХ 21**](#_heading=h.3fwokq0)

[9.1. Структура БД 21](#_heading=h.1v1yuxt)

[9.2. Модель данных 21](#_heading=h.4f1mdlm)

[9.3. Синхронизация и хранение данных 21](#_heading=h.2u6wntf)

[9.4. Обеспечение целостности и безопасности 22](#_heading=h.19c6y18)

[**10. ПЛАН ТЕСТИРОВАНИЯ И ВАЛИДАЦИИ 23**](#_heading=h.3tbugp1)

[10.1. Методика тестирования 23](#_heading=h.28h4qwu)

[10.2. Юнит-тесты 23](#_heading=h.nmf14n)

[10.3. Интеграционные тесты 23](#_heading=h.37m2jsg)

[10.4. UI/UX тестирование 23](#_heading=h.1mrcu09)

[10.5. Нагрузочное тестирование 23](#_heading=h.46r0co2)

[**11. ЭКСПЛУАТАЦИОННЫЕ ТРЕБОВАНИЯ И ДОКУМЕНТАЦИЯ 24**](#_heading=h.2lwamvv)

[11.1. Руководство пользователя 24](#_heading=h.111kx3o)

[11.2. Инструкция по установке и настройке 24](#_heading=h.3l18frh)

[11.3. Техническая документация для сопровождения 24](#_heading=h.206ipza)

[**12. ПЛАН РЕАЛИЗАЦИИ И ЭТАПЫ РАБОТ 25**](#_heading=h.4k668n3)

[12.1. Этапы разработки 25](#_heading=h.2zbgiuw)

[12.2. Сроки выполнения 25](#_heading=h.1egqt2p)

[12.3. Ответственные лица и распределение ролей 25](#_heading=h.3ygebqi)

[**13. ДОПОЛНИТЕЛЬНЫЕ ТРЕБОВАНИЯ И ОСОБЕННОСТИ 27**](#_heading=h.2dlolyb)

[13.1. Локализация 27](#_heading=h.sqyw64)

[13.2. Масштабируемость и расширяемость 27](#_heading=h.3cqmetx)

[13.3. Сценарии использования 27](#_heading=h.1rvwp1q)

[**14. ПРИЛОЖЕНИЯ 28**](#_heading=h.4bvk7pj)

[Приложение A – Прототипы интерфейса 28](#_heading=h.2r0uhxc)

[**ЗАКЛЮЧЕНИЕ 30**](#_heading=h.1664s55)

# 1. ОБЩИЕ ПОЛОЖЕНИЯ

## 1.1. Назначение документа

Настоящее техническое задание определяет требования к функциональности, архитектуре, интерфейсу, безопасности и тестированию программного продукта – приложения «Time Tracker – Помидоро (25/5)». Документ предназначен для использования командой разработчиков, тестировщиков и технических специалистов, а также для контроля соответствия готового продукта поставленным требованиям.

## 1.2. Цель разработки

Основной целью разработки является создание удобного и функционального приложения для управления временем, реализующего технику «Помидоро» (циклы работы по 25 минут с перерывами по 5 минут). Приложение должно способствовать повышению производительности пользователей, позволяя планировать и отслеживать выполнение задач, а также предоставлять детализированную статистику по использованию времени.

## 1.3. Область применения

Приложение предназначено для широкого круга пользователей, в том числе для фрилансеров, менеджеров проектов, студентов и всех, кто заинтересован в эффективном управлении своим рабочим временем. Продукт рассчитан на работу в настольных операционных системах (Windows, Linux, с возможной поддержкой macOS) и предусматривает работу в клиент-серверном режиме с использованием удалённой базы данных PostgreSQL.

## 1.4. Нормативные ссылки

* ГОСТ 19.201-78 «Единая система программной документации. Техническое задание. Общие требования к содержанию и оформлению»
* ГОСТ 34.602-89 «Единая система программной документации. Алгоритмы и программные схемы»
* Методические рекомендации по разработке программных продуктов с использованием языка C++ и библиотеки Qt.

# 2. ТЕРМИНОЛОГИЯ И ОПРЕДЕЛЕНИЯ

## 2.1. Основные термины

* **Пользователь** – конечный исполнитель, использующий приложение для управления временем.
* **Задача** – единица работы, планируемая пользователем, характеризующаяся дедлайном, названием и количеством планируемых временных интервалов (отображаемых в виде «звёздочек»).
* **Помидоро** – временной интервал, состоящий из 25 минут работы и 5 минут перерыва.
* **Таймер** – инструмент, отображающий текущий рабочий или нерабочий интервал с визуальной индикацией оставшегося времени.
* **QSS** – язык описания стилей для элементов интерфейса, аналог CSS, используемый в Qt.

## 2.2. Сокращения и условные обозначения

* **UI** – User Interface (Пользовательский интерфейс)
* **DB** – Database (База данных)
* **API** – Application Programming Interface (Программный интерфейс приложения)
* **CRUD** – Create, Read, Update, Delete (Операции создания, чтения, обновления и удаления данных)

# 3. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

## 3.1. Назначение приложения

Приложение «Time Tracker – Помидоро (25/5)» предназначено для:

* Планирования задач с использованием метода «Помидоро».
* Отслеживания выполнения задач по дням недели с сортировкой по дате.
* Предоставления пользователю возможности гибкой настройки задач, таймера, звуков и тем оформления.

## 3.2. Описание бизнес-процессов

* **Планирование задач:** Пользователь добавляет задачи, указывая название, дедлайн и количество «звёздочек» (каждая звезда = один цикл 25/5). Задачи автоматически сортируются по дате с разбивкой по дням недели.
* **Управление задачами:** При выборе задачи открывается окно настроек, позволяющее изменить параметры, запустить задачу или удалить её.
* **Запуск таймера:** Помимо индивидуальных задач, предусмотрена возможность запуска «бесконечного» таймера по правилу 25/5.
* **История задач:** Все выполненные и прерванные задачи сохраняются в истории с подробной информацией (например, число временных промежутков, если таймер завершался раньше).
* **Настройка приложения:** Пользователь может изменять звуковые уведомления, выбирать и добавлять новые темы оформления (QSS-файлы), а также настраивать параметры таймера.
* **Анализ статистики:** На странице профиля доступны данные о количестве выполненных, незавершённых и оставшихся задач.

## 3.3. Сценарии использования

Приведены примеры типичных сценариев:

* **Сценарий 1:** Пользователь заходит на главную страницу, просматривает запланированные задачи, добавляет новую задачу с дедлайном и нужным количеством «звёздочек», после чего запускает задачу, редактируя её параметры в появившемся окне.
* **Сценарий 2:** Пользователь переходит на страницу истории, где выбирает конкретную задачу для просмотра детальной информации о выполненных временных интервалах и прерванных циклах.
* **Сценарий 3:** В разделе настроек пользователь изменяет звуковые сигналы, выбирает новую тему оформления для приложения и настраивает параметры визуализации таймера.
* **Сценарий 4:** На странице профиля пользователь просматривает статистику за определённый период, анализирует количество выполненных и незавершённых задач, а также настраивает параметры учетной записи.

## 3.4 Сравнение с аналогами на рынке

Приложение было вдохновлено приложением Forest и Toggl Track, которые существуют только на мобильных устройствах или в браузерной версии. Ввиду этого, а также отсутствия удобной истории задач и возможности применять разные стили, данный продукт и является, если не концептуально новым, то явно отличающимся от своих аналогов. Создаваемый продукт разрабатывается больше для личных целей и личного удобства. Разработка ведется не в коммерческих целях

<https://www.forestapp.cc/> - Forest

<https://toggl.com/> - Toggl Track

# 4. СТРУКТУРА И СОСТАВ СИСТЕМЫ

## 4.1. Основные компоненты приложения

Приложение состоит из следующих модулей:

* **Модуль управления задачами:** Обеспечивает создание, редактирование, удаление и запуск задач.
* **Модуль отображения интерфейса:** Отвечает за рендеринг главной страницы, страницы истории, настроек и профиля.
* **Модуль таймера:** Отвечает за работу таймера с разделением на фазу работы и отдыха, визуальное отображение оставшегося времени (анимация круга).
* **Модуль настроек:** Позволяет пользователю изменять звуки, темы оформления (с использованием QSS-файлов) и параметры таймера.
* **Модуль профиля и статистики:** Обеспечивает авторизацию, отображение статистических данных и управление аккаунтом.
* **Модуль работы с БД:** Интегрирован с PostgreSQL для хранения задач, истории, настроек и учетных данных пользователей.

## 4.2. Обзор функциональных страниц

* **Главная страница:** Содержит список запланированных задач, отсортированных по дате, с разбивкой по дням недели, кнопки для добавления задачи и запуска бесконечного таймера.
* **Страница истории:** Отображает все задачи в виде, аналогичном главной странице, с возможностью просмотра подробной информации о каждой задаче (без возможности редактирования).
* **Страница настроек:** Позволяет настраивать звуковые уведомления, темы оформления (подключаемые QSS-файлы) и параметры таймера.
* **Страница профиля:** Предоставляет доступ к личному кабинету с информацией об аккаунте, статистикой выполненных и незавершённых задач.

## 4.3. Взаимодействие модулей и компонентов

Все модули взаимодействуют посредством внутреннего API, обеспечивающего:

* Обмен данными между интерфейсом и БД.
* Вызов функций управления задачами и таймером.
* Обработку событий пользовательского интерфейса (нажатия кнопок, выбор опций и т.д.).
* Синхронизацию настроек пользователя с удалённой БД.

# 5. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

## 5.1. Главная страница

**Функциональность:**

* **Список задач:** Отображение задач с сортировкой по дате, разделённых по дням недели. Каждая задача должна содержать:
  + **Название задачи**
  + **Дедлайн** (дата и время)
  + **Планируемое время выполнения,** отображаемое в виде «звёздочек» (каждая звезда = 25 минут работы + 5 минут перерыва)
* **Кнопка «Добавить задачу»:** Открывает диалоговое окно для ввода параметров новой задачи.
* **Кнопка «Запустить бесконечный таймер»:** Инициирует запуск таймера без привязки к конкретной задаче.
* **Взаимодействие:** При нажатии на название задачи открывается окно настроек, где возможно:
  + Изменить параметры задачи (название, дедлайн, количество звёзд)
  + Удалить задачу
  + Запустить задачу

## 5.2. Страница истории задач

**Функциональность:**

* Отображение всех ранее созданных задач в виде, аналогичном главной странице.
* При выборе задачи отображается подробная информация:
  + Количество фактически выполненных временных промежутков
  + Детали прерываний таймера (если таймер завершался раньше)
  + Дата и время выполнения
* Возможность сортировки и фильтрации по датам, статусу задачи и другим параметрам.
* Режим просмотра только для чтения – редактирование и удаление задач не предусмотрены.

## 5.3. Страница настроек

**Функциональность:**

* **Настройка звуков:** Пользователь может выбрать звуковые уведомления для таймера, а также настроить громкость и тип звука.
* **Добавление новых тем:**
  + Возможность загрузки и выбора новых QSS-файлов для изменения внешнего вида элементов интерфейса.
  + Отдельное оформление для таймера через отдельный QSS-файл.
* **Настройка таймера:**
  + Возможность изменения параметров таймера, таких как длительность фаз работы и отдыха, а также визуальные эффекты (цвет, толщина контура, анимация уменьшения круга).

## 5.4. Страница профиля

**Функциональность:**

* **Авторизация:** Вход в аккаунт пользователя с использованием логина и пароля.
* **Просмотр статистики:** Отображение информации:
  + Количество завершённых задач
  + Количество незавершённых задач
  + Оставшиеся задачи
  + Другие статистические показатели (например, суммарное время работы, эффективность)
* **Управление аккаунтом:** Возможность редактирования личных данных, смены пароля и выхода из системы.

## 5.5. Таймер (отдельное окно)

**Функциональность:**

* **Режимы работы:** Таймер имеет две основные фазы:
  + **Фаза работы:** 25 минут
  + **Фаза отдыха:** 5 минут
* **Визуальное отображение:**
  + В центре окна располагается круг, контур которого уменьшается пропорционально оставшемуся времени.
  + Возможность раскрашивания таймера через выбор QSS-файла в настройках.
* **Управление:**
  + Возможность приостановки, сброса или изменения длительности текущего цикла.
  + Аудио уведомления при переключении между фазами.

## 5.6. Управление задачами (CRUD-операции)

**Функциональность:**

* **Создание задачи:** Открытие диалогового окна для ввода всех параметров (название, дедлайн, планируемое время в виде количества «звёзд»).
* **Редактирование задачи:** Доступно через окно настроек, открывающееся при клике по названию задачи.
* **Удаление задачи:** Опция удаления через окно настроек с подтверждением действия.
* **Запуск задачи:** Возможность начать выполнение задачи с привязкой к таймеру.

## 5.7. Управление аккаунтом и статистикой

**Функциональность:**

* **Аутентификация:** Вход по логину/паролю с возможностью восстановления забытого пароля.
* **Сохранение данных:** Все изменения пользователя (настройки, история задач, статистика) синхронизируются с базой данных PostgreSQL.
* **Отображение статистики:** Графическое и табличное представление выполненных, незавершённых и ожидающих задач.

# 6. ТРЕБОВАНИЯ К ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ

## 6.1. Общие принципы дизайна

* Интерфейс должен быть интуитивно понятным, минималистичным и не перегруженным информацией.
* Применение принципов адаптивного дизайна для корректного отображения на различных разрешениях экрана.
* Стандартные темы должны соблюдать принцип единообразия, т.е. каждая страница так или иначе похожа на каждую

## 6.2. Описание элементов интерфейса по страницам

* **Главная страница:**
  + Список задач в виде карточек с краткой информацией.
  + Кнопки «Добавить задачу» и «Запустить бесконечный таймер» размещены в верхней части экрана.
* **Страница истории:**
  + Табличное или карточное представление задач с возможностью развёрнутого просмотра.
* **Страница настроек:**
  + Разделы для настройки звуков, тем оформления и параметров таймера.
* **Страница профиля:**
  + Личный кабинет с блоком статистики, настройками аккаунта и кнопкой выхода.

## 6.3. Использование QSS для оформления

* Приложение должно поддерживать динамическую подгрузку QSS-файлов, позволяющих менять оформление:
  + Общий вид интерфейса (цвета, шрифты, отступы)
  + Оформление таймера (цвета, анимация уменьшения контура)
* Интерфейс должен предоставлять возможность предварительного просмотра выбранной темы перед применением.

## 6.4. Анимационные эффекты

* Визуальный индикатор оставшегося времени (уменьшающийся круг) должен обновляться плавно.
* Плавные переходы между экранами и анимация появления/исчезновения диалоговых окон.

# 7. АРХИТЕКТУРА И ТЕХНОЛОГИИ РЕАЛИЗАЦИИ

## 7.1. Выбор языка и инструментов разработки

* **Основной язык разработки:** C++ (с использованием современного стандарта C++17 или выше).
* При необходимости использования вспомогательных языков (например, для написания скриптов или UI-логики) допускается применение Python, JavaScript (с учётом требований безопасности и интеграции).

## 7.2. Используемые библиотеки и фреймворки

* **Qt Framework:** Для создания графического интерфейса пользователя, работы с QSS, обработки событий и реализации анимаций.
* **Boost (при необходимости):** Для вспомогательных функций и обеспечения кроссплатформенной совместимости.
* **Библиотеки для работы с PostgreSQL:** Например, libpqxx или аналогичные для интеграции с БД.

## 7.3. Интеграция с базой данных PostgreSQL

* Приложение должно поддерживать соединение с удалённой базой данных PostgreSQL.
* Необходима реализация механизма синхронизации данных, позволяющего сохранять и обновлять информацию о задачах, истории и настройках в режиме реального времени.
* Обеспечить защиту данных (шифрование паролей, защита соединения).

## 7.4. Механизмы авторизации и управления учётными записями

* Реализовать безопасный процесс авторизации с использованием хэширования паролей (например, с использованием алгоритма bcrypt).
* Организовать разграничение прав доступа для различных категорий пользователей (например, администратор и обычный пользователь).

# 8. НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

## 8.1. Производительность и масштабируемость

* Приложение должно обеспечивать быстрый отклик интерфейса (время отклика не более 200 мс при стандартных условиях).
* Возможность работы с большим объёмом задач (до 10 000 записей) без существенного снижения производительности.
* Поддержка одновременной работы нескольких пользователей при интеграции с удалённой БД.

## 8.2. Безопасность и защита данных

* Обеспечить шифрование передаваемых данных (использование SSL/TLS для соединения с БД).
* Защита от SQL-инъекций и других типов атак.
* Регулярное обновление и патчинг зависимостей и библиотек.

## 8.3. Надёжность и отказоустойчивость

* Обеспечить автоматическое сохранение данных в случае сбоев работы приложения.
* Реализовать механизм резервного копирования базы данных.
* Восстановление сессии пользователя после нештатного завершения работы.

## 8.4. Кроссплатформенность и требования к ОС

* Приложение должно работать на Windows, Linux (и, при возможности, на macOS).
* Минимальные системные требования:
  + Процессор: 1 ГГц и выше
  + ОЗУ: не менее 2 ГБ
  + Свободное место на диске: 100 МБ для установки приложения и 500 МБ для базы данных (в зависимости от объёма хранимых данных) ((значения НЕ итоговые и могут поменяться))

# 9. ТРЕБОВАНИЯ К БАЗЕ ДАННЫХ

## 9.1. Структура БД

* Использование СУБД PostgreSQL.
* Разделение БД на таблицы, отвечающие за:
  + Пользовательские учётные записи (users)
  + Задачи (tasks)
  + Историю выполнения задач (task\_history)
  + Настройки пользователя (user\_settings)
  + Статистику и логи (statistics, logs)

## 9.2. Модель данных

* Каждая задача должна содержать следующие поля:
  + **id** (уникальный идентификатор)
  + **user\_id** (идентификатор пользователя)
  + **title** (название задачи)
  + **deadline** (дата и время дедлайна)
  + **planned\_cycles** (количество запланированных циклов, отображаемых звёздочками)
  + **status** (статус задачи: активная, выполненная, отменённая)
  + **created\_at, updated\_at** (даты создания и последнего изменения)

## 9.3. Синхронизация и хранение данных

* Все операции с данными должны выполняться через транзакции для обеспечения целостности данных.
* Реализовать механизм кеширования для ускорения отображения списка задач.

## 9.4. Обеспечение целостности и безопасности

* Применение индексов для оптимизации поиска по дате и статусу задач.
* Использование механизмов контроля доступа к данным на уровне БД.

# 10. ПЛАН ТЕСТИРОВАНИЯ И ВАЛИДАЦИИ

## 10.1. Методика тестирования

* Тестирование проводится по методологии «разработки через тестирование» (TDD), с обязательным написанием юнит-тестов для каждого модуля.
* Проведение интеграционного тестирования для проверки взаимодействия между компонентами системы.

## 10.2. Юнит-тесты

* Покрытие тестами функций управления задачами, авторизации, работы с БД и обработки пользовательских событий.
* Автоматизированное тестирование с использованием фреймворков для C++ (например, Google Test).

## 10.3. Интеграционные тесты

* Тестирование корректности передачи данных между модулем UI и модулем работы с БД.
* Проверка синхронизации настроек и обновлений статистики в режиме реального времени.

## 10.4. UI/UX тестирование

* Проведение ручного и автоматизированного тестирования пользовательского интерфейса.
* Оценка удобства, интуитивности и визуальной эстетики приложения.

## 10.5. Нагрузочное тестирование

* Моделирование одновременной работы большого числа пользователей.
* Проверка устойчивости системы при большом объёме данных (до 10 000 задач).

# 11. ЭКСПЛУАТАЦИОННЫЕ ТРЕБОВАНИЯ И ДОКУМЕНТАЦИЯ

## 11.1. Руководство пользователя

* Подробное описание функциональности каждой страницы.
* Инструкции по добавлению, редактированию и удалению задач.
* Описание работы таймера и настроек звуков и тем оформления.

## 11.2. Инструкция по установке и настройке

* Пошаговая инструкция по установке приложения на различных операционных системах.
* Рекомендации по настройке соединения с сервером БД PostgreSQL.
* Описание необходимых зависимостей и способов их установки.

## 11.3. Техническая документация для сопровождения

* Описание архитектуры приложения, схемы модулей и взаимосвязей.
* Документация по API внутренних модулей.
* Рекомендации по обновлению и масштабированию системы.

# 12. ПЛАН РЕАЛИЗАЦИИ И ЭТАПЫ РАБОТ

## 12.1. Этапы разработки

1. **Анализ и проектирование:**
   * Сбор и анализ требований
   * Составление архитектурной схемы и прототипов интерфейса
2. **Разработка базового функционала:**
   * Реализация модулей управления задачами, таймера и работы с БД
3. **Интеграция UI и функциональных модулей:**
   * Настройка взаимодействия между модулями, интеграция с QSS
4. **Тестирование и отладка:**
   * Проведение юнит-, интеграционных и нагрузочных тестов
5. **Подготовка эксплуатационной документации и релиз:**
   * Составление руководств, инструкции по установке

## 12.2. Сроки выполнения

Общий срок реализации проекта – 18-24 недели в зависимости от сложности и объёма работ.

## 12.3. Ответственные лица и распределение ролей

* **Руководитель проекта:** Координация работ, контроль сроков.
* **Ведущий разработчик:** Архитектурные решения, реализация основных модулей.
* **Инженер по UI/UX:** Разработка макетов и прототипов, настройка QSS.
* **Специалист по БД:** Проектирование и настройка PostgreSQL, обеспечение безопасности данных.
* **Тестировщик:** Разработка сценариев тестирования, проведение функциональных и нагрузочных тестов.

# 13. ДОПОЛНИТЕЛЬНЫЕ ТРЕБОВАНИЯ И ОСОБЕННОСТИ

## 13.1. Локализация

* Приложение должно поддерживать мультиязычность с первоначальной версией на русском языке.
* Возможность добавления дополнительных языковых пакетов без значительных доработок.

## 13.2. Масштабируемость и расширяемость

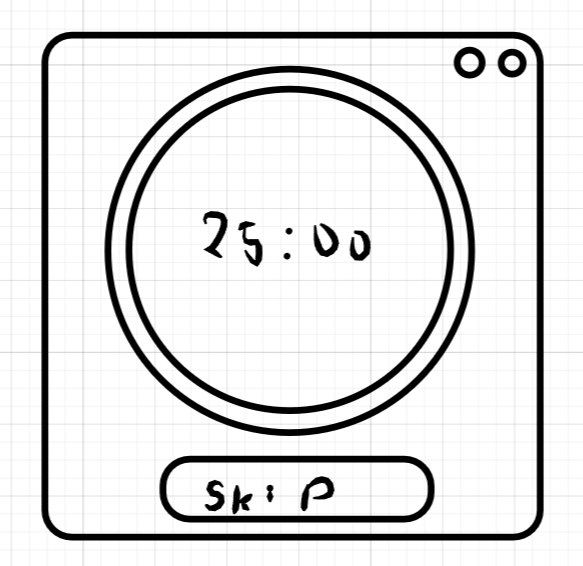
* Архитектура приложения должна предусматривать возможность расширения функционала (например, интеграция с календарями, модулями анализа времени и т.д.).
* Использование модульного подхода для упрощения внедрения новых функций.

## 13.3. Сценарии использования

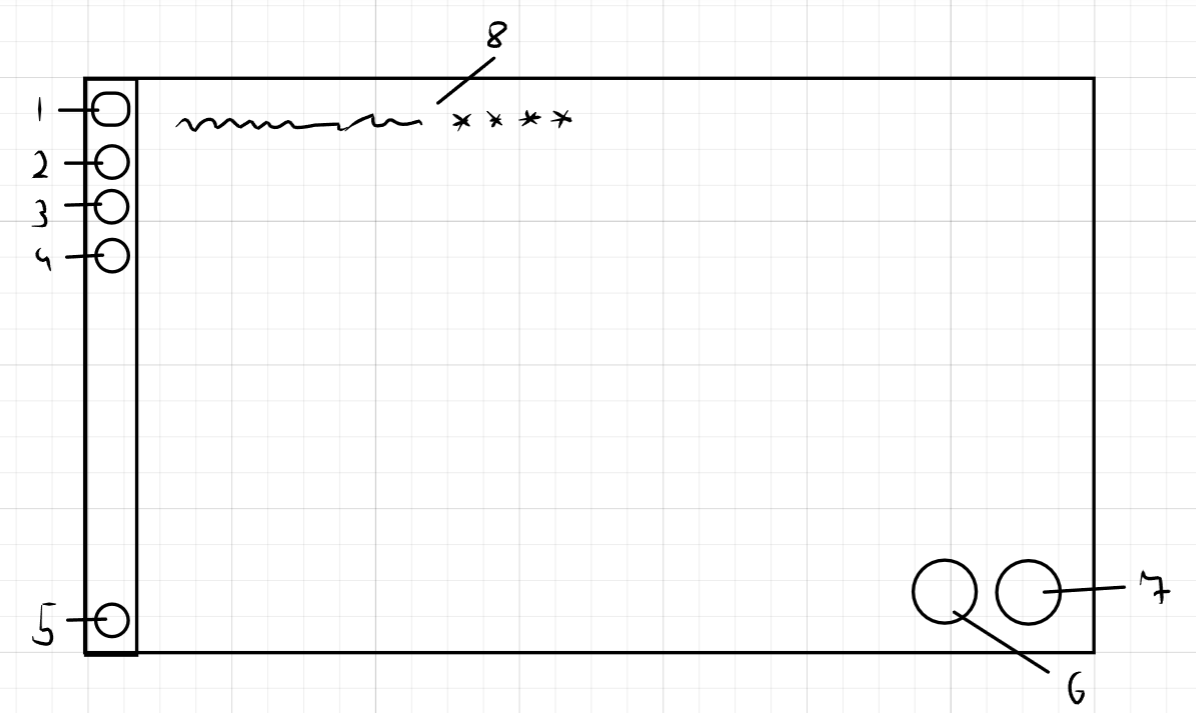
* **Основной рабочий сценарий:** Планирование задач, запуск таймера, мониторинг выполнения.
* **Сценарий восстановления:** Автоматическое сохранение данных и восстановление сеанса пользователя после сбоя.
* **Сценарий обновления:** Возможность обновления приложения без потери пользовательских настроек и истории.

# 14. ПРИЛОЖЕНИЯ

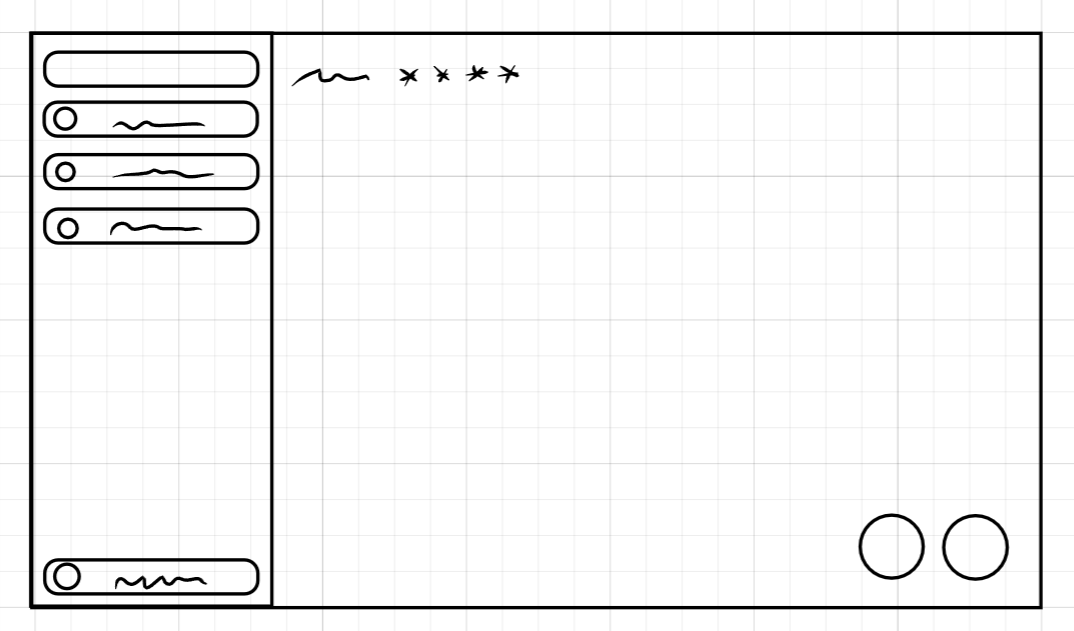
## Приложение A – Прототипы интерфейса



а) окно таймера



б) страница с задачами: 1 – развернуть меню; 2 – страница с задачами; 3 – страница истории; 4 – страница профиля; 5 – страница настроек; 6 – добавить задачу; 7 – бесконечный таймер; 8 – задача



в) меню в развернутом виде

# ЗАКЛЮЧЕНИЕ

Данное техническое задание определяет полный перечень функциональных, нефункциональных и эксплуатационных требований к приложению «Time Tracker – Помидоро (25/5)». Реализация проекта по данному ТЗ позволит создать современное, удобное и надёжное средство управления временем, соответствующее методике «Помидоро» и удовлетворяющее потребности пользователей в эффективном планировании рабочего процесса.