**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «ВГУ»)**

Факультет Компьютерных наук

Кафедра технологий обработки и защиты информации

Курсовой проект

Сайт для учёта рабочего времени «ITime»

09.03.02 Информационные системы и технологии

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Смирнов А.К., 3 курс, д/о*

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Железной А.С., 3 курс, д/о*

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Шаталов А.А., 3 курс, д/о*

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ старший преподаватель *Тарасов В.С.*

Воронеж 2021

**Содержание**

[1 Введение 3](#_Toc71322137)

[2 Постановка задачи 4](#_Toc71322138)

[3 Анализ предметной области 5](#_Toc71322139)

[3.1 Глоссарий 5](#_Toc71322140)

[3.2 Анализ существующих решений 6](#_Toc71322141)

[3.3 Анализ задачи 8](#_Toc71322142)

[3.3.1. Варианты использования приложения 8](#_Toc71322143)

[3.3.2. Взаимодействие компонентов системы 9](#_Toc71322144)

[3.3.3. Варианты состояния системы 9](#_Toc71322145)

[3.3.4. Варианты действия в системе 10](#_Toc71322146)

[3.3.5. Развёртывание приложения 12](#_Toc71322147)

[3.3.6. IDEF0 13](#_Toc71322148)

[4 Реализация 14](#_Toc71322149)

[4.1 Сущности 14](#_Toc71322150)

[4.2 Сценарии воронок конверсии 17](#_Toc71322151)

[4.3 Графический интерфейс 17](#_Toc71322152)

[5 Тестирование 18](#_Toc71322153)

[5.1 Дымовое тестирование 18](#_Toc71322154)

[5.2 UI тесты 18](#_Toc71322155)

[5.3 Юзабилити тестирование 18](#_Toc71322156)

[6 Заключение 19](#_Toc71322157)

# Введение

В современном мире есть множество компаний, предприятий, которым надо следить за работой сотрудников. Это можно делать разными способами, но в 21 веке удобнее и практичнее применять современные технологии. В настоящее время существуют различные приложения по учету времени работы сотрудников, но у каждого из таких приложений различные функциональные возможности, разный интерфейс.

Наше web-приложение должно облегчить и упростить введение учета рабочего времени сотрудников. web-приложение должно предоставлять основную необходимую функциональность:

* Учёт времени и контроль сотрудников
* Мониторинг времени работы сотрудников над заданиями проекта
* Администрирования работы сотрудников

Данный курсовой проект посвящен разработке именно такого, простого в использовании, но в то же время выполняющего самые необходимые функции, web-приложения, способного прозрачно показывать затраченное время на работу сотрудником.

# Постановка задачи

Цель курсовой работы: реализовать web-приложение, которое отвечает следующим требованиям:

* стабильная работа на современных веб-браузерах,
* сдержанный внешний вид, выдержанный в едином стиле,
* отсутствие необходимости в справке для возможности осуществления основных задач:

1) авторизация,

2) работа с календарем,

3) учет времени,

4) администрирование проектов и задач,

5) редактирование информации в личном кабинете,

Для достижения данной цели были выделены следующие задачи:

1) разработка Front-end части сайта,

2) разработка Back-end части сайта,

3) создание связи между Front-end и Back-end частями сайта,

4) разработка базы данных.

# Анализ предметной области

## Глоссарий

* **front-end** - клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса.
* **back-end** - программно-аппаратная часть сервиса.
* **REST API** — это стиль архитектуры программного обеспечения для построения распределенных масштабируемых веб-сервисов.
* **Микросервисная архитектура -** вариант сервис-ориентированной архитектуры программного обеспечения, направленный на взаимодействие насколько это возможно небольших, слабо связанных и легко изменяемых модулей — ***микросервисов*** (структурная единица микросерверной архитектуры).
* **GitHub** - крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки.
* **Веб-сервис, интернет-сервис, система, веб-приложение, проект** - идентифицируемая уникальным веб-адресом (URL-адресом) программная система со стандартизированными интерфейсами, а также HTML-документ сайта, отображаемый браузером пользователя.
* **Header** - визуальный элемент, расположенный в верхней части страницы.
* **Footer** - визуальный элемент, расположенный в нижней части страницы.
* **Администратор** - человек, имеющий доступ к расширенному функционалу веб-сервиса.
* **Личный кабинет** — это раздел сервиса, в котором Пользователь может получить доступ к персональным данным
* **Контент** - наполнение сайта.
* **Разработчик –** человек или организация, задействованная в разработке ПО.
* **Task Leader –** руководитель задачи
* **Project Leader –** руководитель проекта
* **Проект** — это направленная деятельность, нацеленная на получение определенных результатов.
* **Задание** — это декомпозиция проекта на определённые этапы.
* **Сотрудник –** лицо, обладающее не только определенными правами, но и кругом возложенных на него должностных обязанностей.

## Анализ существующих решений

1. Time Doctor - учёт и контроль рабочего времени команд дистанционных сотрудников. Программа Time Doctor содержит в себе множество функциональных особенностей – от онлайн-мониторинга сотрудников до учета рабочего времени. Интерфейс выглядит несколько устаревшим, но это надежная платформа, которая определенно заслуживает внимания.   
 Time Doctor рассчитан на широкую пользовательскую аудиторию – будь то удаленные сотрудники (даже один фрилансер) или инженеры, разрабатывающие ПО в крупных компаниях, таких как Apple и Verizon.   
Дополнительные функции: управление проектами, отслеживание сотрудников.

Достоинства:

* Хорошие функциональные возможности для контроля персонала
* Есть опция для выписки инвойсов в мобильном приложении
* Интеграция со сторонними сервисами, такими как Asana.

Недостатки:

* Устаревший интерфейс. Для функции таймера нужно использовать десктопную версию или расширение в Chrome.
* Из-за неудобного интерфейса и некоторых других недостатков по части юзабилити Time Doctor заметно уступает конкурентам.

2. Kickidler - система по контролю и учету рабочего времени для работников организаций. Сервис акцентирует внимание на наблюдении за работниками в текущий момент, а также видеофиксации всех нарушений. Использование системы гарантирует, что специалисты будут тратить свое рабочее время на выполнение трудовых обязанностей, а не личных нужд.

Достоинства:

* Онлайн-мониторинг ПК: с его помощью можно увидеть, что делает сотрудник, какие утилиты использует
* Учет рабочего времени: тайм-трекер - контроль рабочих часов, выявление переработок, опозданий, ранних уходов, любых отвлечений персонала от рабочих моментов. Вся статистика отображается в виде диаграмм;
* Анализ эффективности сотрудников: система показывает, какую долю времени специалист посвящает рабочим моментам, а какую - соцсетям, мессенджерам, играм и т.д.;
* Запись видео с монитора: видеофиксация всех действий в рабочее время - настройка сжатия и частоты кадров экономят место на серверах;
* Удаленный доступ: дистанционное управление компьютером пользователя;
* Кейлоггер: запоминание нажатий клавиш сотрудником - сопоставление набора и действий на экране монитора.

Недостатки:

* Возможные проблемы с установкой, но они решаются с помощью эффективной тех поддержкой

## Анализ задачи

## Варианты использования приложения

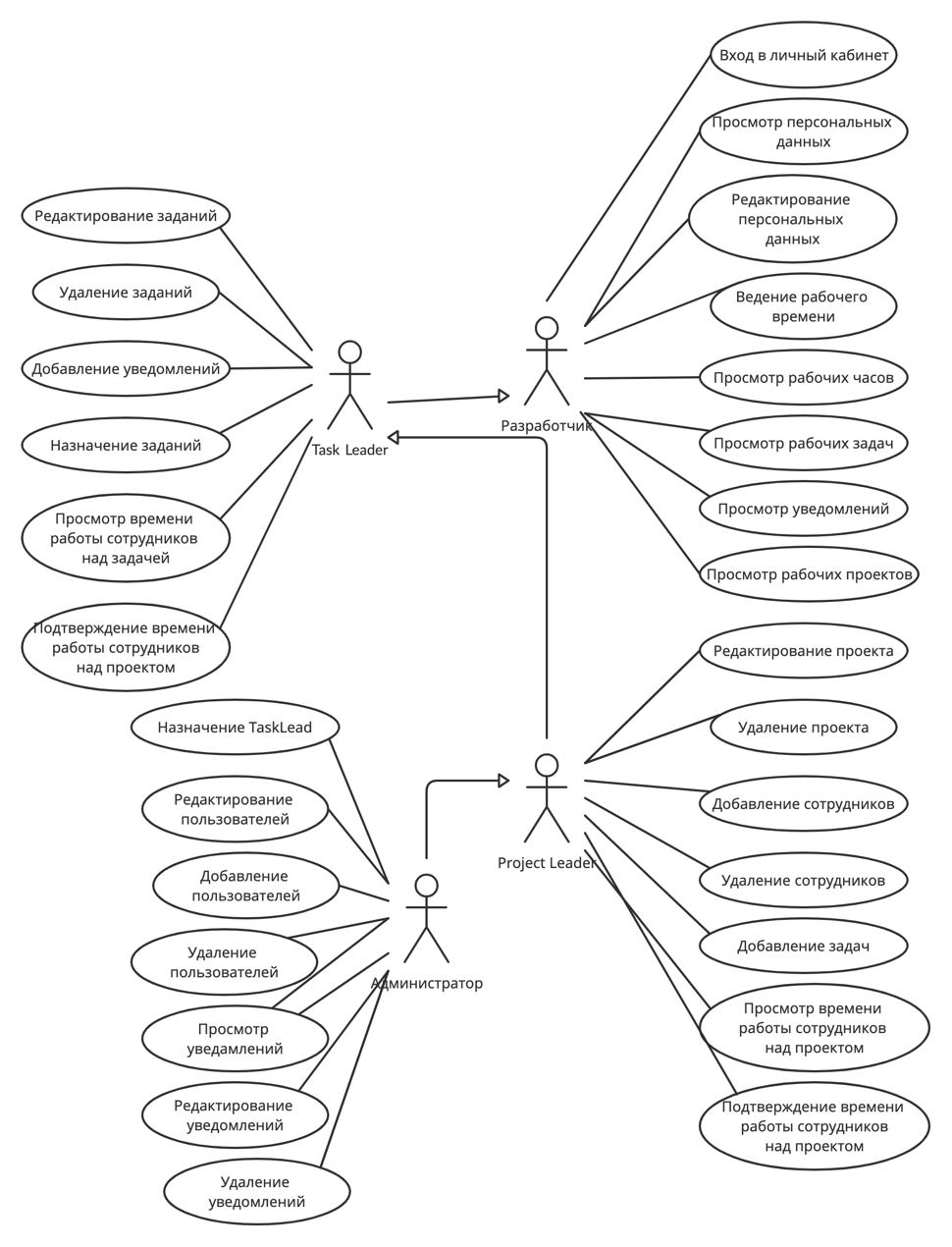


Рисунок 1 - Диаграмма прецедентов

При взаимодействии с web-приложением у пользователя есть определенный список возможностей, зависящий от уровня доступа, который изображен на рисунке 1.

## Взаимодействие компонентов системы

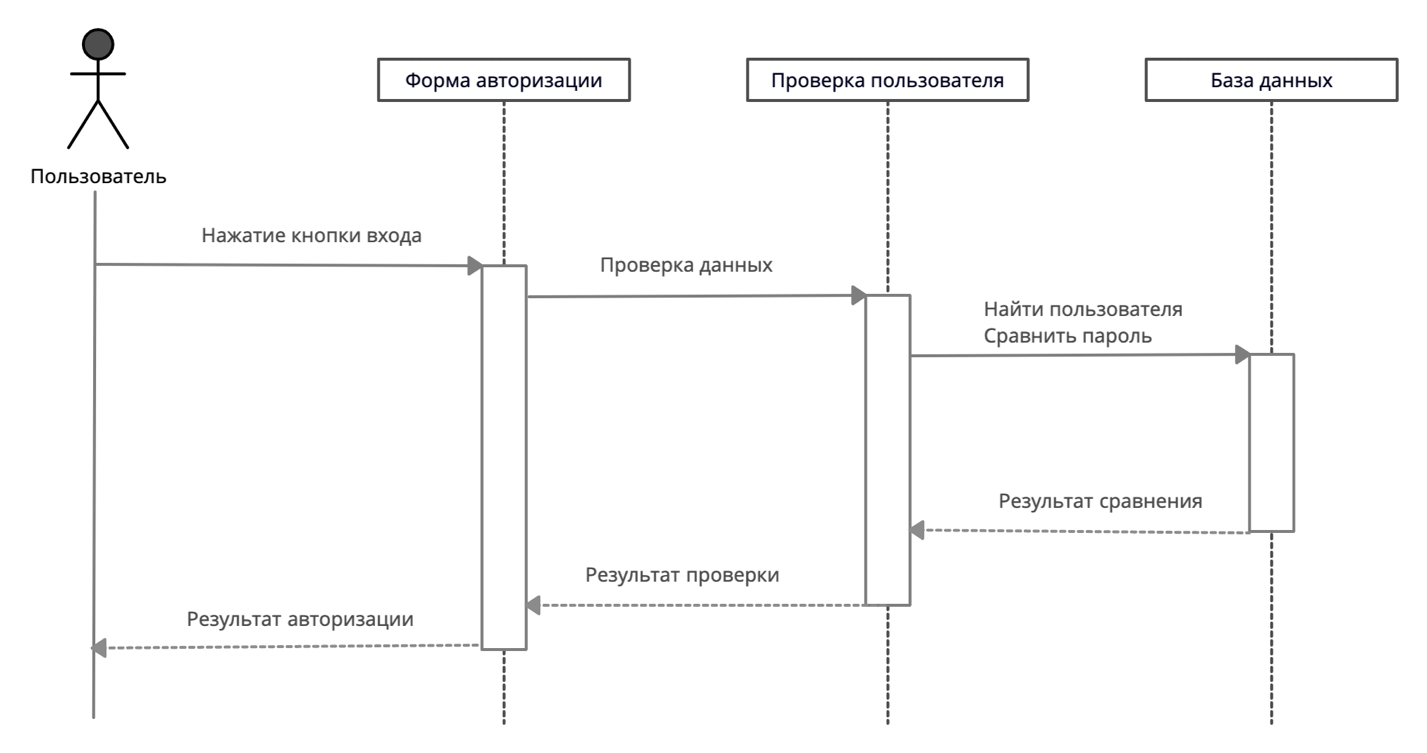


Рисунок 2 - Диаграмма последовательностей

На рисунке 2 показана диаграмма последовательности, на которой изображено упорядоченное во времени взаимодействие объектов при авторизации пользователя.

Для авторизации пользователь обращается к форме авторизации, которая передаёт введённые данные на проверку в модуль проверки пользователя. Тот в свою очередь проверяет существование данного пользователя в базе данных и совпадение введённого пароля с паролем, хранящимся в базе данных. Модуль проверки пользователя посылает ответ проверки на форму авторизации, которая выводит пользователю результат авторизации.

## Варианты состояния системы

Рисунок 3 - Диаграмма состояний

Изображенная на Рисунке 3, диаграмма состояний, отражает возможные состояния отработанного времени. После старта времени, таймер запускается, когда время останавливается, оно отправляется на проверку и переходит в состояние ожидания проверка. Если время прошло проверку, то оно переходит в состояние подтверждения и добавляется в общий список времени работы сотрудника. Если время не прошло проверку, то оно переходит в состоянии отклонения и удаляется.

## Варианты действия в системе

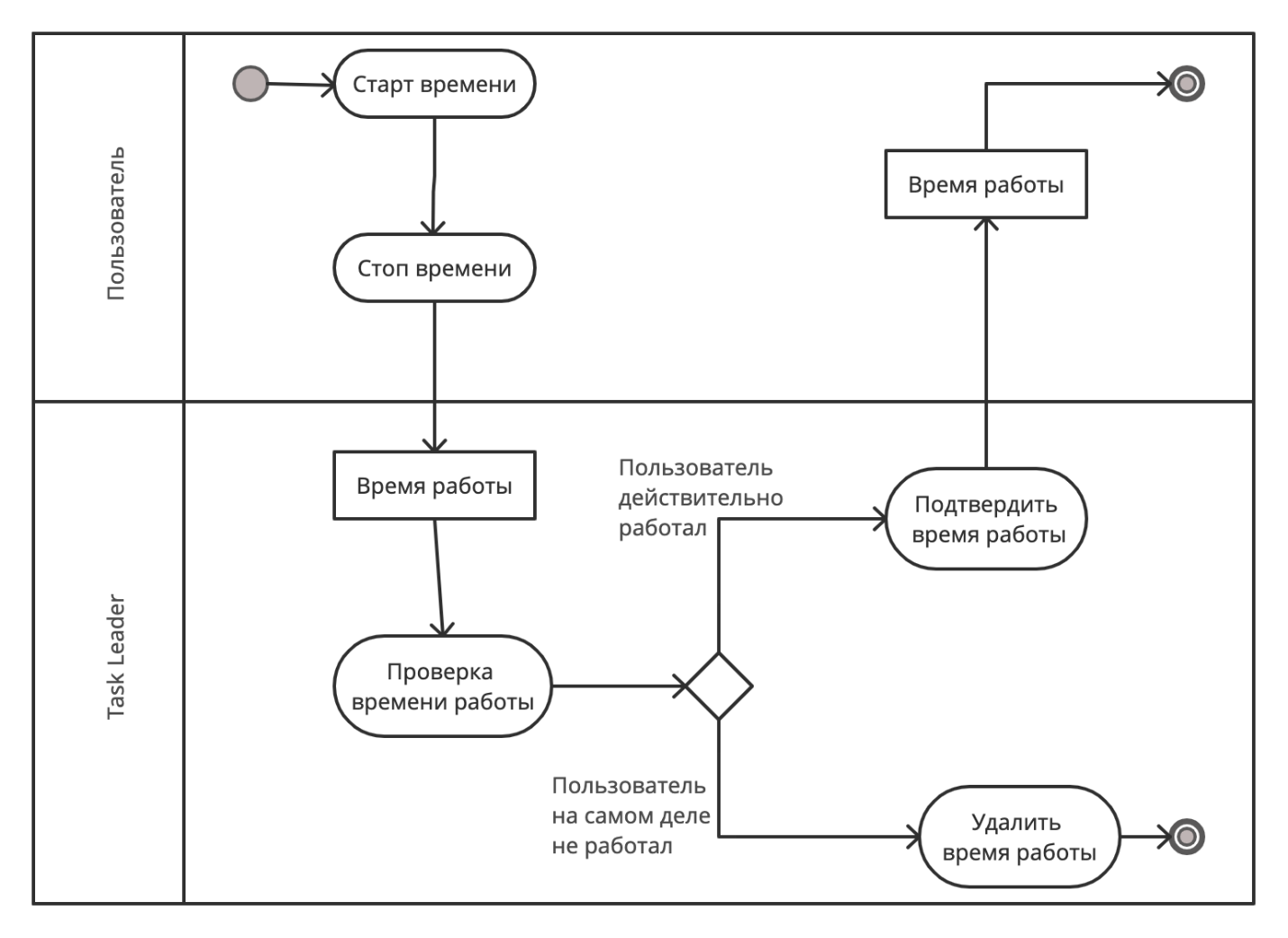


Рисунок 4 - Диаграмма активностей

Диаграмма активности, изображенная на Рисунке 5, отражает действия пользователя и Task Leader. После того, как пользователь заканчивает работу над проектом, время переходит к администратору. При получении времени Task Leader должен ее проверить. Если время не соответствует требованиям, то администратор удаляет ее. Если время соответствует требованиям, то Task Leader подтверждает ее, и время переходит пользователю. Пользователь получает собственное время, и время в свою очередь закрепляется за ним.

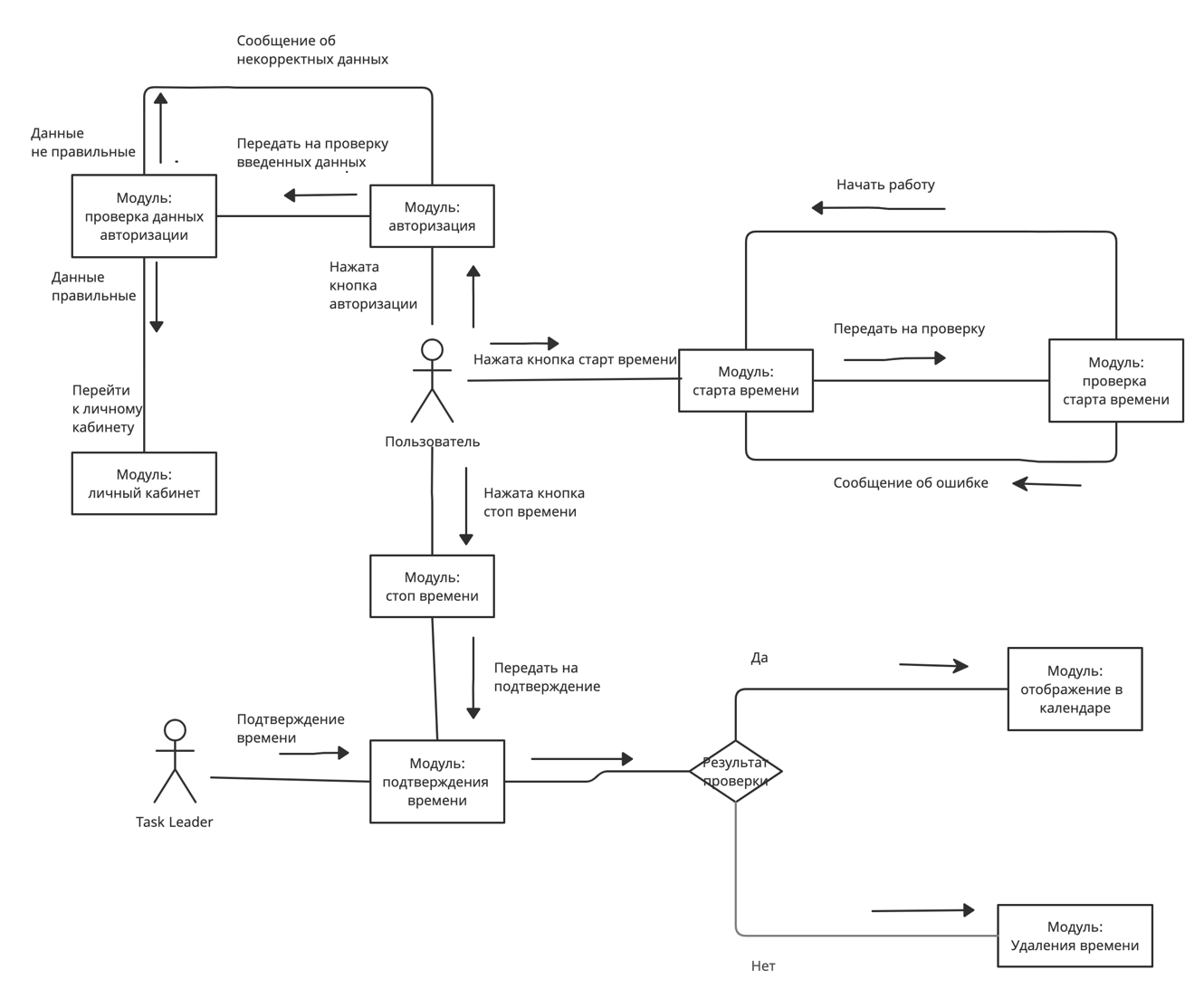


Рисунок 5 - Диаграмма взаимодействий

Диаграмма взаимодействий, изображенная на рисунке 5, отражает возможные действия пользователя и системы.

Если пользователь захочет авторизоваться, ему надо заполнить форму авторизации и нажать соответствующую кнопку, после чего система проверяет корректность введённых данных, если все правильно пользователь попадает в личный кабинет, в противном случае выдается ошибка.

Если пользователь захочет начать работу над проектом, ему надо запустить таймер нажав кнопку старта времени, система проверит возможность старта времени, если это невозможно, то система предупредит его об этом. Если старт времени возможен, то отчет запущен, после нажатия кнопки стоп времени, время попадает на модерацию, где Task Leader одобряет время. Если время одобрено, то оно заноситься в календарь, а если не одобрено удаляется.

## Развёртывание приложения

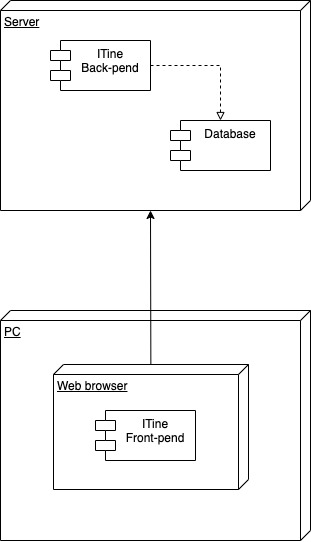


Рисунок 6 - Диаграмма развертывания

На Рисунке 6 представлена диаграмма развертывания, чтобы определить какие аппаратные компоненты («узлы») существуют, какие программные компоненты работают на каждом узле и как различные части этого комплекса соединяются друг с другом. Для разрабатываемого web-приложения узлом устройства является персональный компьютер и сервер, а в качестве узла среды выполнения выступает web-браузер. В браузере развернут front-end приложения, а на серверной части back-end и база данных.

## IDEF0



Рисунок 7 – IDEF0

Рассмотрим основной бизнес – процесс на примере контекстной диаграммы, представленной на Рисунок 7 - Контекстная диаграмма. Данная диаграмма представляет собой общее видение процесса работы Сервиса. Работу сервиса регулируют Правила Компании и Пользовательское соглашение, которые определяются совместно с Законы РФ, Уровни доступа к Системе, которые определяются Администратор. Обеспечивают работу Системы Администратора Исходная БД. На вход в Систему поступает Сотрудник компании, которому надо учитывать рабочее время. На выходе Система выдает Подтвержденное/Неподтвержденное время.

# Реализация

## Сущности

Рисунок 8 - Диаграмма классов

На Рисунке 8 изображена диаграмма классов. Она отображает их отношения.

1. Класс «Speciality» – представляет собой отражение сущности специальности. Класс имеет следующие свойства:

* «specialty\_id» – уникальный идентификатор,
* «specialty\_data» – наименование специальности.

1. Класс «Position» – отражает информацию о позиции для какого-либо разработчика (junior, middle, senior и т.д.). Класс имеет следующие свойства:

* «position\_id» – уникальный идентификатор,
* «position\_data» – название/описание позиции.

1. Класс «Working\_time» – представляет собой отражение сущности. Класс имеет следующие свойства:

* «id» – уникальный идентификатор,
* «start time» – начало работы над проектом,
* «end time» – конец работы над проектом,
* «idUser» – уникальный идентификатор developer, который работал это время,
* «idTask» – уникальный идентификатор task над каким работали это время.

1. Класс «Tasks» – класс задачи в рамках проекта:

* «task\_id» - уникальный идентификатор задачи,
* «title» - название задачи,
* «results» - текстовое описание результатов задачи,
* «planned\_deadline» - планируемое время окончания задачи,
* «actual\_deadline» - актуальное время окончания задачи,
* «project\_id» - уникальный идентификатор проекта, в рамках которого выполняется данная задача,
* «task\_lead\_id» - уникальный идентификатор developer, являющегося ответственным за эту задачу.

1. Класс «Project» – отражает информации о проекте. Класс имеет следующие свойства:

* «project\_id» – уникальный идентификатор Project
* «title» – название проекта
* «description» – описание проекта
* «deadline» – дата, к которой должен быть выполнен проект

1. Класс «Developers» – класс разработчика с полями:

* «developer\_id» - уникальный идентификатор разработчика,
* «name» - имя разработчика,
* «surname» - фамилия разработчика,
* «patronymic» - отчество / middle name разработчика,
* «position» - уникальный идентификатор, по которому получается информация о позиции разработчика,
* «specialty» - уникальный идентификатор, по которому получается информация о специальности разработчика,
* «date\_of\_birth» - дата рождения разработчика,
* «password» - хешированное значение пароля для разработчика,
* «email» - почта разработчика, логин по совместительству.

1. Класс «Notifications» – содержит информацию о уведомлениях. Класс имеет следующие свойства:

* «notification\_id» – уникальный идентификатор уведомления,
* «sender» - уникальный идентификатор отправителя уведомления,
* «content» – текстовая информация уведомления,
* «date» – дата отправки уведомления.

1. Класс «Changing\_dates» – класс отражает историю изменения актуального срока выполнения задачи. Класс имеет следующие свойства:

* «changing\_id» – уникальный идентификатор изменения дедлайна,
* «cause» – причина изменения актуального срока,
* «deadline\_before» – дата актуального срока выполнения задачи до изменения,
* «deadline\_after» – новая дата дедлайна,
* «task\_id» – уникальный идентификатор задачи, у которой изменили актуальный срок.

1. Класс «Achievements» – класс достижений, которые может получать разработчик в течение своей работы с приложением. Класс имеет следующие свойства:

* «achievents\_id» - уникальный идентификатор достижения,
* «title» - название достижения,
* «description» - описание достижения. За что было получено, информация о достижении и т.д.,
* «linkImg» - ссылка на изображение достижения.

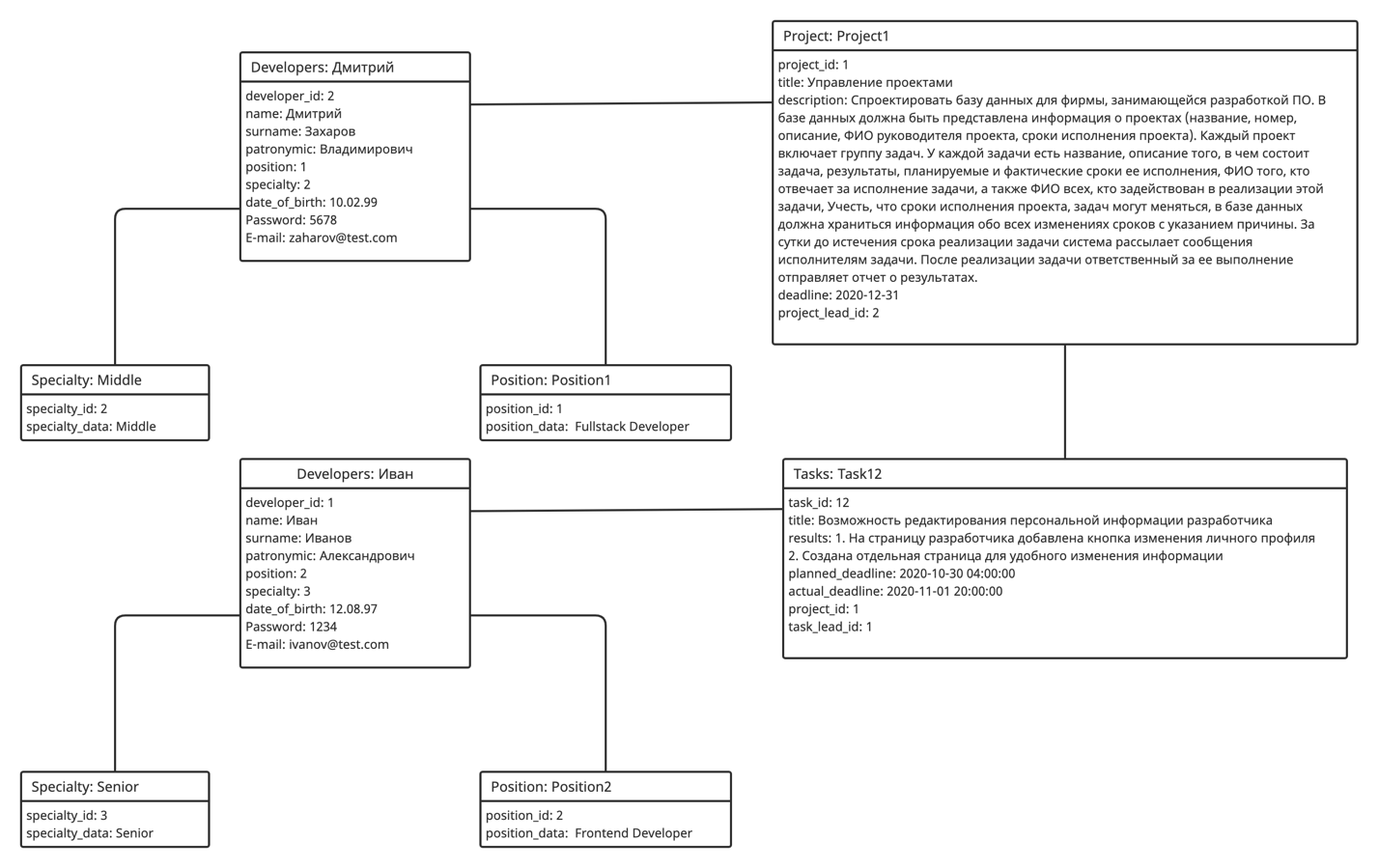


Рисунок 9 - Диаграмма объектов

На Рисунке 9 изображена диаграмма объектов, которая отражает множество экземпляров классов и отношений между ними в некоторый момент времени. На ней изображены два экземпляра класса “Developers”, экземпляр класса “Task”, которая хранит в себе ссылку на “Developers: Иван” и экземпляр класса “Project”, которая хранит в себе ссылку на “Developers: Дмитрий”. В свою очередь экземпляры класса “Developers” хранит в себе ссылку на экземпляр класса “Specialty” и “Position”.

## Сценарии воронок конверсии

## Графический интерфейс

# Тестирование

## Дымовое тестирование

## UI тесты

## Юзабилити тестирование

# Заключение