Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка базы данных для идентификации пользователей при осуществлении контрольно-пропускного режима с использованием документов, удостоверяющих личность»

Студент ИУ7-61Б (Группа) _____ (Подпись, дата) Постнов С. А. (Фамилия И. О.)

Руководитель курсовой работы _____ (Подпись, дата) Кудрявцев М. А. (Фамилия И. О.)

2024 г.

РЕФЕРАТ

Расчетно-пояснительная записка 40 с., 13 рис., 13 табл., 24 источн., 1 прил.

ИДЕНТИФИКАЦИЯ НА КПП, БАЗЫ ДАННЫХ, ВЕБ-ПРИЛОЖЕНИЕ, MONGODB, POSTGRESQL.

Цель работы — разработка базы данных для идентификации пользователей при осуществлении контрольно-пропускного режима с использованием документов, удостоверяющих личность.

В результате работы был проведен анализ существующих СУБД и методов хранения изображений, спроектирована диаграмма вариантов использования, диаграмма базы данных, ограничения целостности и ролевая модель. Разработана база данных, триггер и веб-приложение. Проведено исследование зависимости времени получения и сохранения изображения от типа СУБД.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время все больше компаний вводят систему контрольно-пропускного режима, что объясняется некоторыми основными причинами [1]:

1) контроль времени прихода и ухода работника;
2) обеспечение только санкционированного доступа в офис (здание, помещение);
3) обеспечение сохранности имущества работодателя.

Целью курсовой работы является разработка базы данных для идентификации пользователей при осуществлении контрольно-пропускного режима с использованием документов, удостоверяющих личность.

Для достижения поставленной цели необходимо решить следующие задачи:

1) формализовать задачу и данные;
2) провести анализ существующих СУБД и методов хранения изображений на основе формализованной задачи, спроектировать диаграмму вариантов использования;
3) спроектировать диаграмму базы данных, ограничения целостности и ролевую модель;
4) выбрать средства реализации базы данных и приложения (в том числе выбор СУБД и средств для хранения изображений);
5) разработать базу данных и приложение;
6) описать интерфейс доступа к базе данных;
7) провести исследование зависимости времени получения и сохранения изображения от типа СУБД.

1 Аналитический раздел

1.1 Формализация задачи и данных

В соответствии с техническим заданием необходимо разработать приложение для идентификации пользователей при осуществлении контрольно-пропускного режима с использованием документов, удостоверяющих личность. Необходимо реализовать промежуточный слой бизнес-логики для взаимодействия с базой данных, а также интерфейс, обеспечивающий возможность авторизации и регистрации пользователя, поиска информации о человеке (людях) по следующим заданным параметрам:

1) номер телефона;

2) имя и фамилия.

По результатам поиска должна предоставляться возможность просмотреть фотографию любого найденного человека с целью дальнейшей идентификации с минимальным временем ожидания.

Для создания базы данных для идентификации пользователей необходимо выделить следующие сущности:

1) компания;

2) сотрудник службы безопасности;

3) сотрудник компании;

4) информационная карточка;

5) документ, удостоверяющий личность;

6) фото;

7) поле документа;

8) контрольно-пропускной пункт (КПП);

9) проход.

На рисунке 1.1 представлена диаграмма «сущность-связь» в нотации Чена.

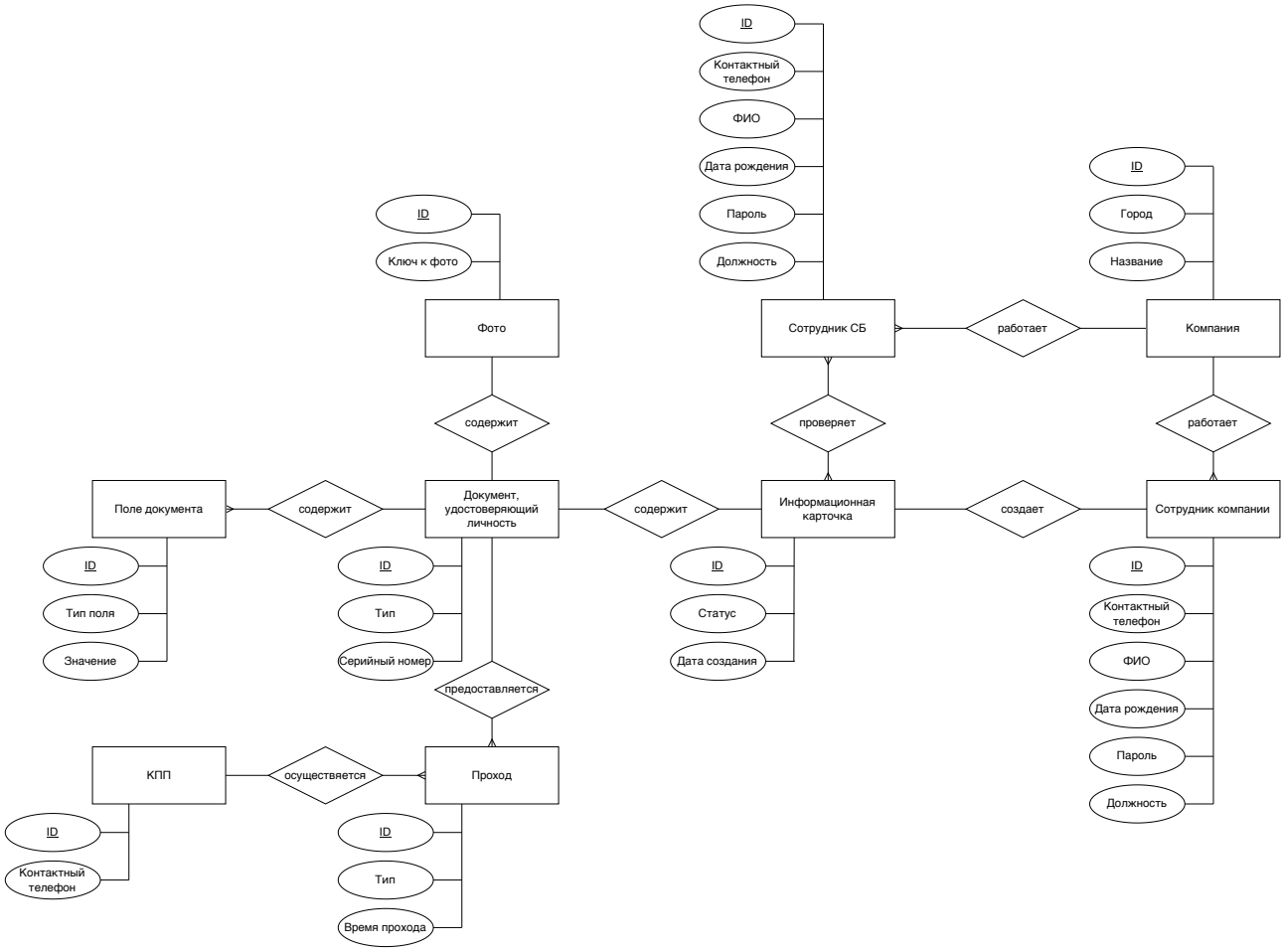

Рисунок 1.1 – Диаграмма «сущность-связь»

На рисунке 1.2 представлена спроектированная диаграмма вариантов использования.

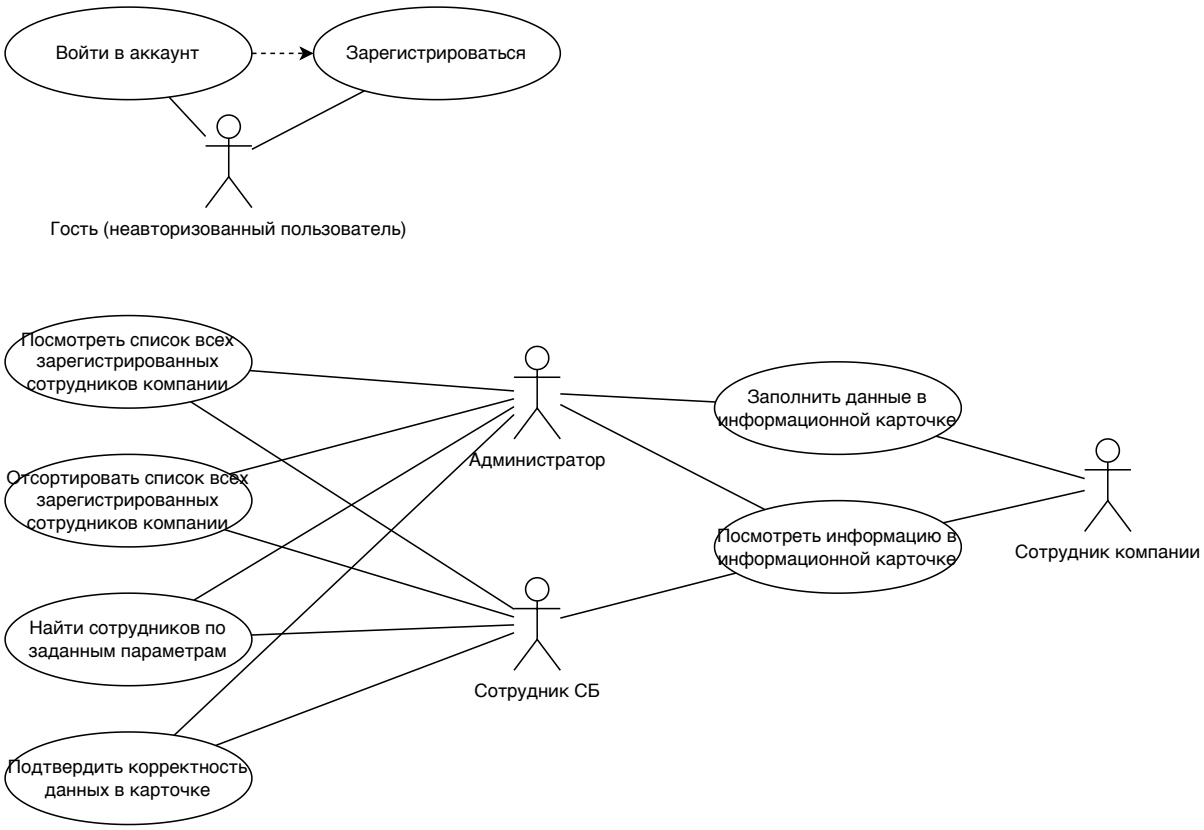

Рисунок 1.2 – Диаграмма вариантов использования

1.2 Описание предметной области

Информационная система – совокупность определенным образом организованных данных (база данных – БД) и комплекса аппаратно-программных средств хранения и манипулирования данными (система управления – СУ). База данных – основа информационной системы, выступающая в роли хранилища данных, которой необходимо управлять при помощи СУБД [2].

Исходя из потребности в манипуляции более сложными объектами (изображениями), а не только набором некоторых простых символьных или текстовых данных, необходимо рассмотреть классификацию СУБД по модели данных [3]:

1) реляционные;

2) нереляционные («NoSQL»).

1.3 Реляционные базы данных

Основоположником теории реляционных баз данных является британский ученый Эдгар Кодд, который в 1970 году опубликовал первую работу по реляционной модели данных. Реляционная база данных – это множество взаимосвязанных именованных отношений («таблиц»). Отношение – это информационная модель реального объекта («сущности») предметной области, формально представленная множеством однотипных кортежей. Кортеж отношения представляет экземпляр моделируемого объекта, свойства которого определяются значениями соответствующих атрибутов («полей») кортежа [2].

По мере значительного повышения мощности компьютеров и соединения их в сети цели их использования стали все более разнообразными. Применение реляционных БД распространилось на широкое множество сценариев далеко за пределами первоначальной обработки коммерческих данных. Работа большей части Интернета до сих пор обеспечивается реляционными БД: онлайн-публикации, дискуссионные форумы, социальные сети, интернет-магазины, игры, предоставляемые как сервис, офисные приложения и многое другое [3].

Далее будут рассмотрены самые популярные реляционные баз данных [4]:

1) Oracle Database;
2) MySQL;
3) Microsoft SQL Server;
4) PostgreSQL.

1.3.1 Oracle Database

Oracle Database – флагманский продукт компании Oracle. Это популярная система управления базами данных и хранения данных, используемая организациями по всему миру для управления своими данными и их хранения. Организации используют Oracle Database в различных отраслях, включая обслуживание конфиденциальных хранилищ данных и отслеживание крупных финансовых транзакций в режиме реального времени.

Основными преимуществами СУБД Oracle Database являются [5]:

1) многоверсионность данных для управления параллельными транзакциями;

2) поддержка ACID;

3) автоматический мониторинг и диагностика баз для выявления проблем производительности и возможность автоматической корректировки.

1.3.2 MySQL

MySQL – самая популярная в мире база данных с открытым исходным кодом. MySQL поддерживает многие наиболее популярные приложения, включая Netflix, Uber, Airbnb, Shopify и др. Поскольку MySQL имеет открытый исходный код, она включает в себя множество функций, разработанных в тесном сотрудничестве с пользователями на протяжении более чем 25 лет. Таким образом, большая часть языков программирования поддерживаются базой данных MySQL [6].

Основными преимуществами СУБД MySQL являются [7—9]:

1) открытый исходный код;

2) низкий порог входа;

3) удобство использования в случае простых приложений и запросов.

1.3.3 Microsoft SQL Server

Microsoft SQL Server – это реляционная система управления базами данных. Приложения и средства подключаются к экземпляру SQL Server или базе данных и взаимодействуют с помощью Transact-SQL. Среда MS SQL Server предоставляет множество различных функций для создания безопасных приложений баз данных. Платформа безопасности MS SQL Server управляет доступом к защищаемым сущностям при помощи проверки подлинности и авторизации [10; 11].

Основными преимуществами СУБД Microsoft SQL Server являются [5]:

1) встроенный механизм репликации;

2) улучшенная скорость за счёт скомпилированных модулей Transact-SQL;

3) поддержка всех существующих драйверов и фреймворков.

1.3.4 PostgreSQL

PostgreSQL – это мощная объектно-реляционная система баз данных с открытым исходным кодом, которая использует и расширяет язык SQL в сочетании со множеством функций, позволяющих безопасно хранить и масштабировать самые сложные рабочие нагрузки с данными. Истоки PostgreSQL восходят к 1986 году как части проекта Калифорнийского университета в Беркли, и более 35 лет ведется активная разработка на базовой платформе [12].

Основными преимуществами СУБД PostgreSQL являются [7—9]:

1) открытый исходный код и максимальное соответствие со стандартами SQL;

2) большое количество сторонних инструментов и библиотек;

3) соответствие требованиям ACID;

4) стабильность работы в условиях большого количества записей и сложных запросов.

Вывод

В таблице 1.1 приведены результаты сравнения рассмотренных реляционных СУБД, исходя из выдвинутых критериев [13].

Таблица 1.1 – Сравнительная таблица для рассмотренных реляционных СУБД

	PostgreSQL	Oracle Database	MySQL	Microsoft SQL Server
Лицензия	Открытая	Коммерческая	Открытая	Коммерческая
Поддержка ACID	Полная	Полная	Частичная	Полная
Возможность хранения бинарных данных	Есть	Есть	Есть	Есть

1.4 Нереляционные базы данных

Термин «NoSQL» был впервые применен в 1998 году Карло Строцци в качестве названия для его небольшой реляционной СУБД, которая не использовала язык SQL для манипулирования данными. С 2009 года термин «NoSQL» стал использоваться уже для обозначения растущего числа распределенных систем управления данными, которые отказывались от поддержки ACID-транзакций – одного из ключевых принципов работы с реляционными базами данных [14].

Существует несколько основных причин широкого внедрения NoSQL баз данных [3]:

1) потребность в больших возможностях масштабирования, чем у реляционных БД, включая обработку очень больших наборов данных или очень большую пропускную способность по записи;

2) специализированные запросные операции, плохо поддерживаемые реляционной моделью;

3) стремление к более динамичным и выразительным моделям данных.

Обычно NoSQL-системы делят, исходя из модели данных, на следующие основные классы [14]:

1) «ключ-значение»;

2) документные;

3) системы типа Google BigTable.

Необходимо рассмотреть только самые популярные нереляционные СУБД «ключ-значение» и документных СУБД соответственно, так как в рамках курсового проекта отсутствует потребность в работе с колоночными СУБД [4]:

1) Redis;

2) MongoDB.

1.4.1 Redis

Redis – система управления данными типа «ключ-значение» с открытым исходным кодом, написанная на C и поддерживающая достаточно богатую модель данных. Значения могут содержать не только строки, но и множества, списки и другие структуры данных. Redis работает в оперативной памяти, за счёт чего достигается высокая производительность. Клиентские библиотеки для работы с Redis доступны для большинства языков программирования, а сама система используется в таких крупных проектах, как Github, StackOverflow и др.

Основными преимуществами СУБД Redis являются [15]:

1) открытый исходный код;

2) быстрые чтение и запись;

3) гибкое использование как кеш-систему или брокер сообщений.

1.4.2 MongoDB

MongoDB – документная СУБД с открытым исходным кодом, написанная на C++ и разрабатываемая компанией 10gen. MongoDB обладает достаточно богатой функциональностью и является одной из самых популярных NoSQL-систем на данный момент и позволяет оперировать JSON-документами (хранимыми и передаваемыми в виде BSON – более компактного двоичного представления JSON), объединяемыми в коллекции, которые, объединяются в базы данных. Заранее предопределенной схемы данных нет, то есть документы в одной коллекции могут содержать разные наборы полей [14].

Основными преимуществами СУБД MongoDB являются [15]:

1) открытый исходный код;

2) богатый язык запросов с возможностью гибкой выборки;

3) функциональные и простые в создании индексы;

4) гибкая сегментация и резервное копирование;

5) поддержка хранения как на диске, так и в оперативной памяти;

6) встроенный инструмент для наиболее оптимальной работы с файлами любых размеров GridFS;

7) соответствие требованиям ACID.

Вывод

В таблице 1.2 приведены результаты сравнения рассмотренных нереляционных СУБД, исходя из выдвинутых критериев [15].

Таблица 1.2 – Сравнительная таблица для рассмотренных нереляционных СУБД

	MongoDB	Redis
Лицензия	Открытая	Открытая
Поддержка ACID	Полная	Отсутствует
Поддержка сложных запросов и расширенной фильтрации	Есть	Частичная
Модель хранения	Как на диске, так и в оперативной памяти	Только в оперативной памяти
Возможность хранения бинарных данных	Есть	Есть

Вывод

В аналитическом разделе были формализованы задача и данные, рассмотрены основные реляционные и нереляционные СУБД, рассмотрены методы хранения изображений и составлены сравнительные таблицы. Спроектирована диаграмма вариантов использования.

2 Конструкторский раздел

2.1 Диаграмма базы данных

На рисунке 2.1 представлена спроектированная диаграмма базы данных и ограничения целостности.

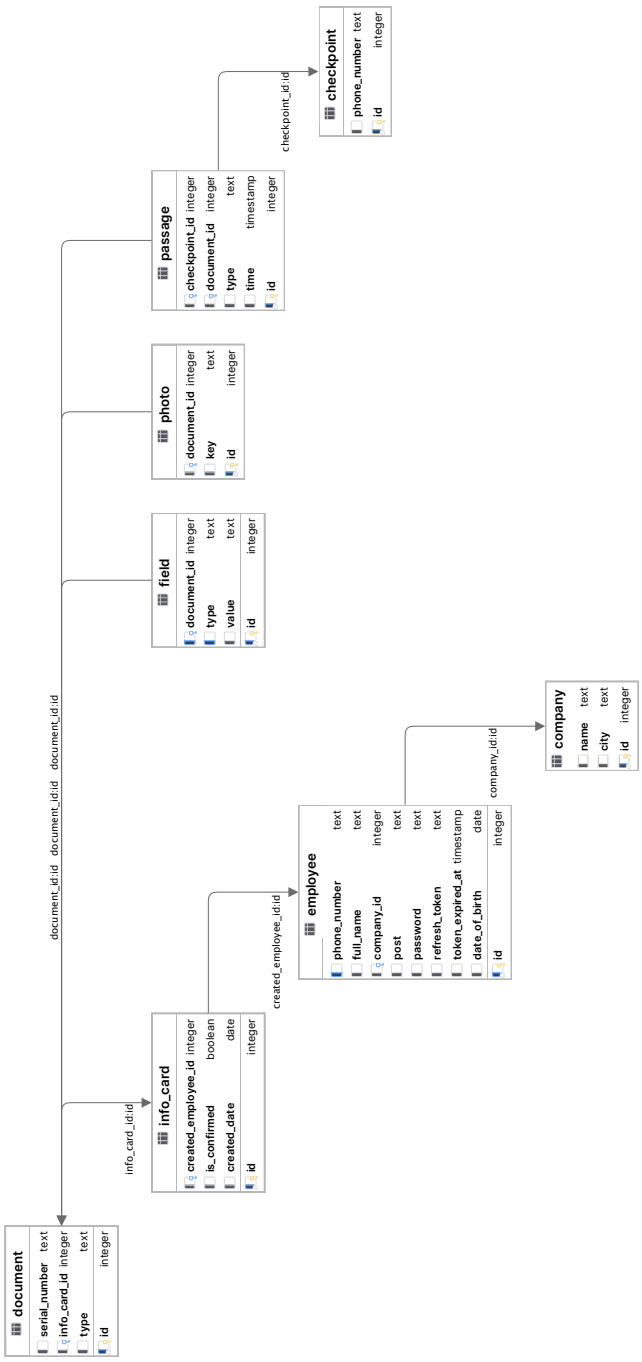

Рисунок 2.1 – Диаграмма базы данных и ограничения целостности

Для проектируемой базы данных необходимы следующие таблицы:

1) таблица компаний – «`company`»;
2) таблица сотрудников компании – «`employee`»;
3) таблица информационных карточек о сотрудниках – «`info_card`»;
4) таблица документов, удостоверяющих личность – «`document`»;
5) таблица полей документов, удостоверяющих личность – «`field`»;
6) таблица фотографий сотрудников из документов, удостоверяющих личность – «`photo`»;
7) таблица контрольно-пропускных пунктов (КПП) – «`checkpoint`»;
8) связующая таблица проходов сотрудников через КПП – «`passage`».

В таблице 2.1 описаны отношения таблицы компаний «`company`».

Таблица 2.1 – Отношения таблицы «`company`»

Атрибут	Тип	Описание
`id`	`integer`	Идентификатор компании
`name`	`text`	Название компании
`city`	`text`	Город расположения офиса/предприятия компании

В таблице 2.2 описаны отношения таблицы сотрудников «employee».

Таблица 2.2 – Отношения таблицы «employee»

Атрибут	Тип	Описание
id	integer	Идентификатор сотрудника
phone_number	text	Номер телефона сотрудника
full_name	text	Полное имя сотрудника
company_id	integer	Идентификатор компании, в которой сотрудник работает
post	text	Должность сотрудника
password	text	Пароль сотрудника
refresh_token	text	Токен для обновления токена доступа к сайту
token_expired_at	text	Время деактивации refresh_token
date_of_birth	date	Дата рождения сотрудника

В таблице 2.3 описаны отношения таблицы информационных карточек о сотрудниках «info_card».

Таблица 2.3 – Отношения таблицы «info_card»

Атрибут	Тип	Описание
id	integer	Идентификатор карточки
created_employee_id	integer	Идентификатор сотрудника, создавшего карточку
is_confirmed	boolean	Подтверждена ли карточка сотрудником СБ
created_date	date	Дата создания карточки

В таблице 2.4 описаны отношения таблицы документов, удостоверяющих личность «`document`».

Таблица 2.4 – Отношения таблицы «`document`»

Атрибут	Тип	Описание
`id`	`integer`	Идентификатор документа
`serial_number`	`text`	Серийный номер документа
`info_card_id`	`integer`	Идентификатор информационной карточки, к которой привязан документ
`type`	`text`	Тип документа

В таблице 2.5 описаны отношения таблицы полей документов, удостоверяющих личность «`field`».

Таблица 2.5 – Отношения таблицы «`field`»

Атрибут	Тип	Описание
`id`	`integer`	Идентификатор поля документа
`document_id`	`integer`	Идентификатор документа, которому принадлежит поле
`type`	`text`	Тип поля
`value`	`text`	Значение поля

В таблице 2.6 описаны отношения таблицы фотографий сотрудников из документов, удостоверяющих личность «photo».

Таблица 2.6 – Отношения таблицы «photo»

Атрибут	Тип	Описание
id	integer	Идентификатор фотографии
document_id	integer	Идентификатор документа, которому принадлежит фотография
key	text	Ключ для получения доступа к фотографии

В таблице 2.7 описаны отношения связующей таблицы проходов сотрудников через КПП «passage».

Таблица 2.7 – Отношения таблицы «passage»

Атрибут	Тип	Описание
id	integer	Идентификатор прохода
checkpoint_id	integer	Идентификатор КПП, через который был проход
document_id	integer	Идентификатор документа, по которому был проход
type	text	Тип прохода (вход или выход)
time	timestamp	Время прохода

В таблице 2.8 описаны отношения таблицы КПП «checkpoint».

Таблица 2.8 – Отношения таблицы «checkpoint»

Атрибут	Тип	Описание
id	integer	Идентификатор КПП
phone_number	text	Номер телефона КПП

2.2 Ролевая модель

Исходя из спроектированной диаграммы вариантов использования 1.2, выделяются следующие роли:

1) гость (неавторизованный сотрудник компании);

2) сотрудник компании(пользователь, зарегистрированный в приложении);

3) сотрудник службы безопасности (сотрудник компании, обеспечивающий безопасность в рамках компании/СБ);

4) администратор.

В таблице 2.9 описаны режимы доступа к данным для представленных выше ролей.

Таблица 2.9 – Режимы доступа к данным для ролей базы данных

Роль	Таблицы с доступом на просмотр	Таблицы с доступом на изменение
`гость`	-	-
`сотрудник компании`	`employee`, `info_card`, `document`, `field`, `photo`	`employee`, `info_card`, `document`, `field`, `photo`
`сотрудник СБ`	все таблицы	`info_card`, `passage`
`админ`	все таблицы	все таблицы

2.3 Триггер базы данных

В течение дня важно отслеживать проходы сотрудников через КПП в целях безопасности и контроля рабочего дня [1].

Для предоставления нужной информации необходимо реализовать триггер базы данных, который после очередного прохода через КПП будет выводить информацию об общем количестве входов и выходов на территории предприятия/офиса.

Схема алгоритма работы триггера представлена на рисунке 2.2.

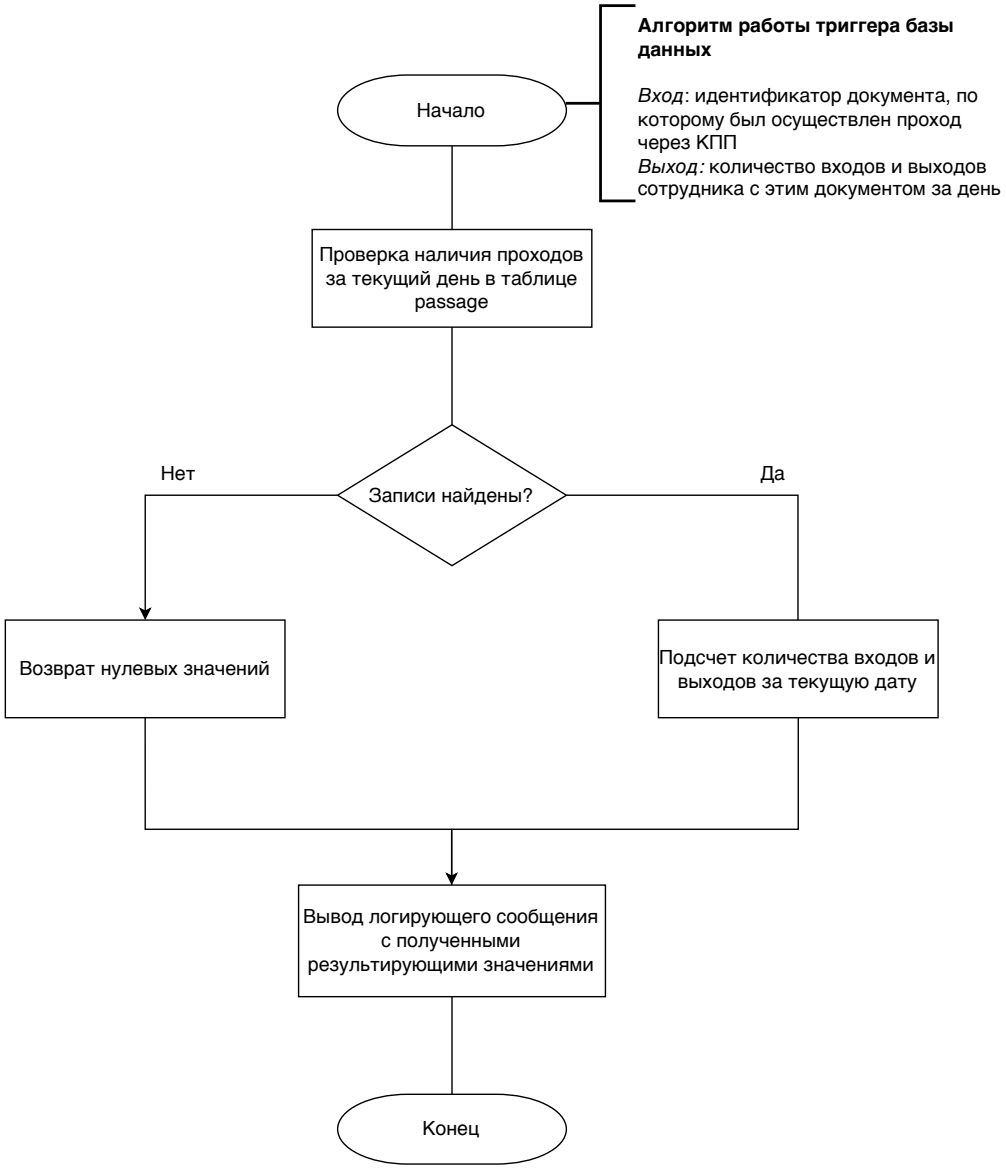

Рисунок 2.2 – Схема алгоритма работы триггера базы данных

Вывод

В конструкторском разделе были спроектированы диаграмма базы данных, ограничения целостности, ролевая модель и триггер базы данных.

3 Технологический раздел

3.1 Выбор средств реализации базы данных и приложения

Исходя из сравнительной таблицы 1.1, в качестве основной СУБД для хранения данных была выбрана PostgreSQL, так как она является бесплатной (в отличие от Oracle Database), а также полностью поддерживает ACID (в отличие от MySQL).

Исходя из сравнительной таблицы 1.2, в качестве вспомогательной СУБД для хранения изображений была выбрана MongoDB, так как она предоставляет возможность хранения как на диске, так и в оперативной памяти, полностью поддерживает ACID и расширенную фильтрацию (в отличие от Redis), а также обладает большей эффективностью работы с бинарными данными за счет встроенного инструмента GridFS, позволяющего оптимально взаимодействовать с файлами любых размеров (в отличие от реляционных СУБД) [16—18].

Приложение предоставляет графический веб-интерфейс и делится на две части: backend и frontend. Для реализации каждой части приложения были выбраны следующие средства:

1) язык программирования для backend части – Go [19];
2) язык программирования для frontend части – Vue.js [20].
3) СУБД – PostgreSQL и MongoDB;
4) расширение языка SQL – PL/pgSQL [21];
5) фреймворк для взаимодействия с СУБД PostgreSQL — pgx [22];
6) фреймворк для взаимодействия с СУБД MongoDB — mongo [23];
7) аутентификация на основе JWT-токенов [24].

3.2 Создание таблиц и объектов базы данных

В листинге А.1 приведены запросы для создания таблиц базы данных с ограничениями целостности. В листинге А.2 приведены запросы для создания триггера базы данных.

3.3 Создание ролей базы данных

В листинге 3.1 приведены запросы для создания роли гостя в базе данных.

Листинг 3.1 – Запросы для создания роли гостя в базе данных

```
create role guest with
    login
    nosuperuser
    nocreatedb
    nocreaterole
    noreplication
    password '123'
    connection limit -1;
```

В листинге 3.2 приведены запросы для создания роли сотрудника в базе данных.

Листинг 3.2 – Запросы для создания роли сотрудника в базе данных

```
create role employee with
    login
    nosuperuser
    nocreatedb
    nocreaterole
    noreplication
    password '123'
    connection limit -1;

grant select, update, insert on
    employee,
    info_card,
    document,
    field,
    photo to employee;
```

В листинге 3.3 приведены запросы для создания роли сотрудника СБ в базе данных.

Листинг 3.3 – Запросы для создания роли сотрудника СБ в базе данных

```
create role security_employee with
    login
    nosuperuser
    nocreatedb
    nocreaterole
    noreplication
    password '123'
    connection limit -1;

grant select on
    all tables in schema public to security_employee;

grant insert, update, delete on
    info_card,
    passage to security_employee;
```

В листинге 3.4 приведены запросы для создания роли администратора в базе данных.

Листинг 3.4 – Запросы для создания роли администратора в базе данных

```
create role service_admin with
    login
    superuser
    createdb
    createrole
    replication
    password '123'
    connection limit -1;
```

3.4 Примеры работы приложения

На рисунках 3.1, 3.2 представлены примеры регистрации и входа сотрудника на сайте соответственно.

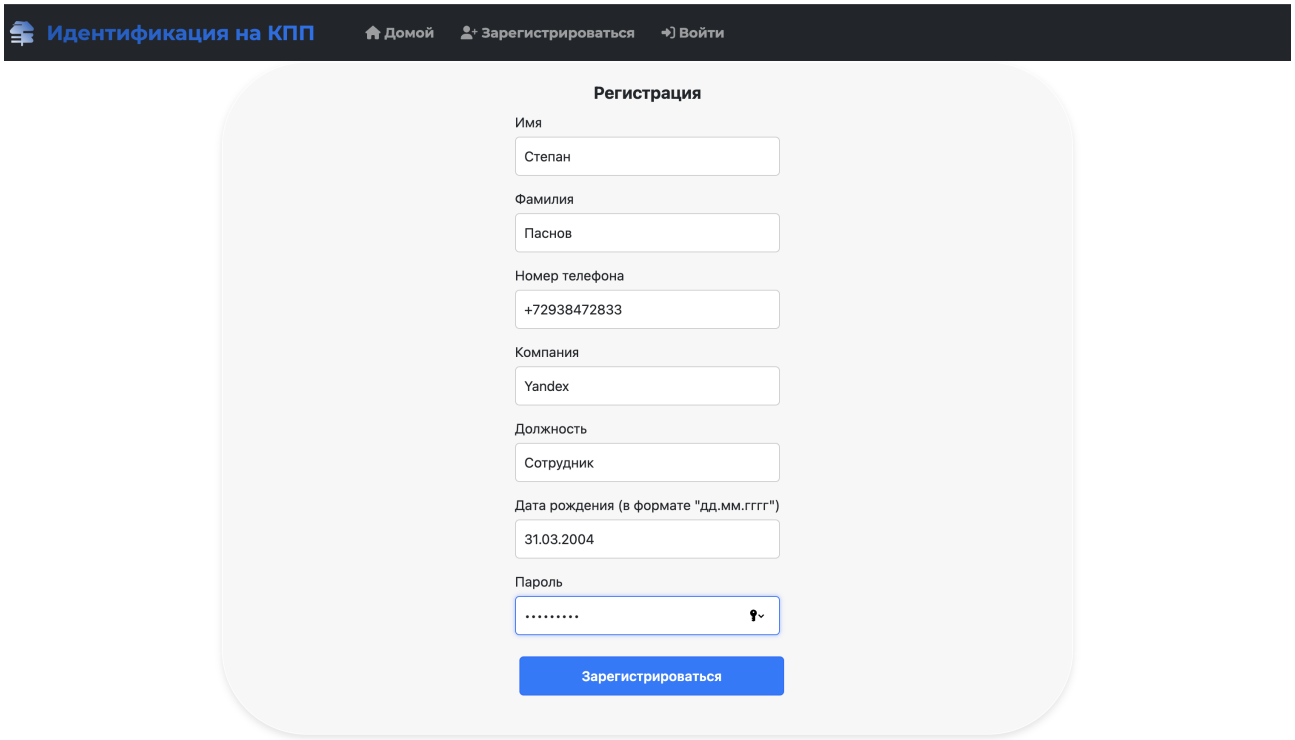

Рисунок 3.1 – Пример регистрации сотрудника

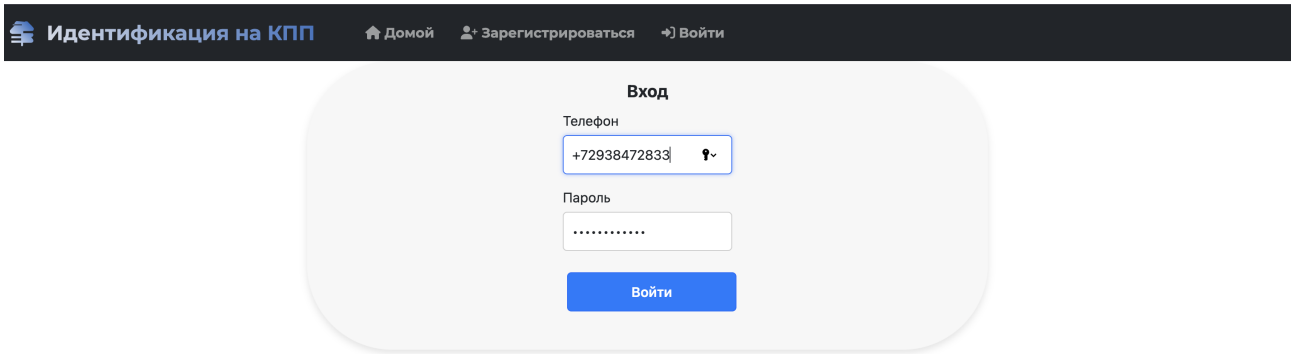

Рисунок 3.2 – Пример входа сотрудника

На рисунке 3.3 представлен пример заполнения сотрудником информации о себе.

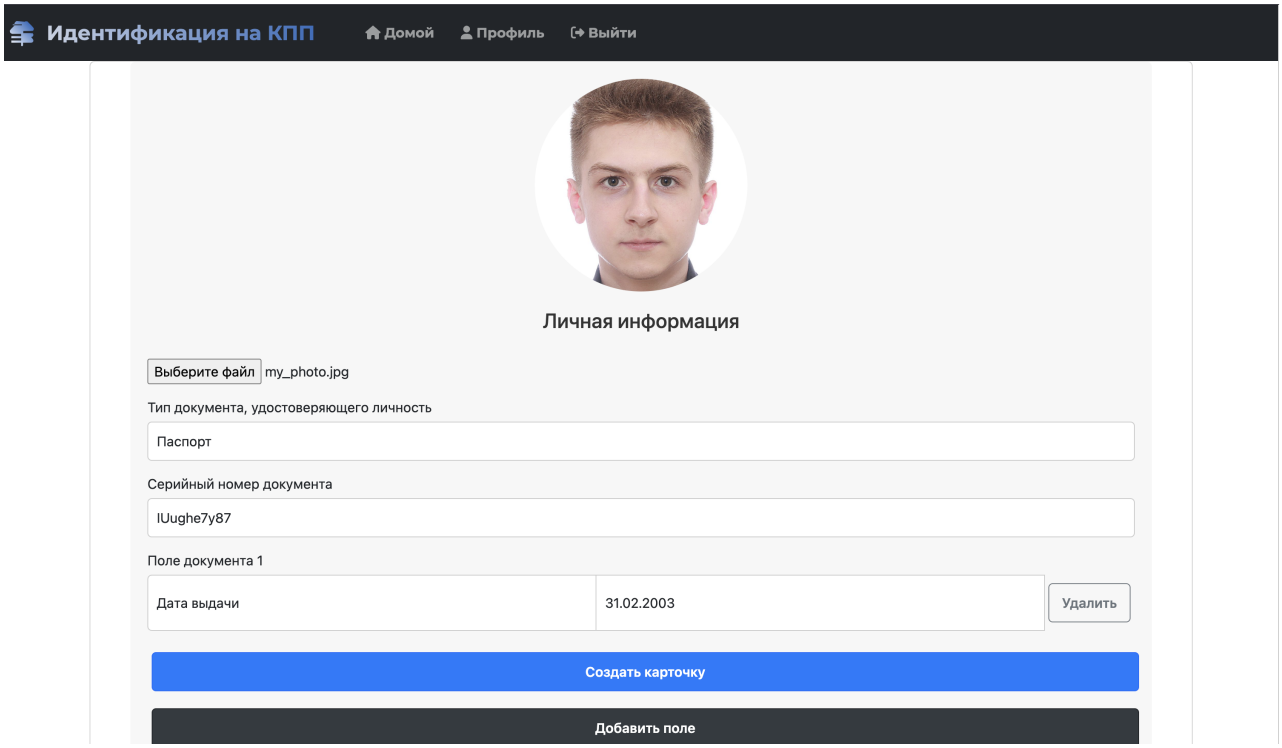

Рисунок 3.3 – Пример заполнения сотрудником информации о себе

После создания сотрудником карточки, представитель СБ находит ее в поисковой строке и проверяет корректность, как показано на рисунках 3.4, 3.5.

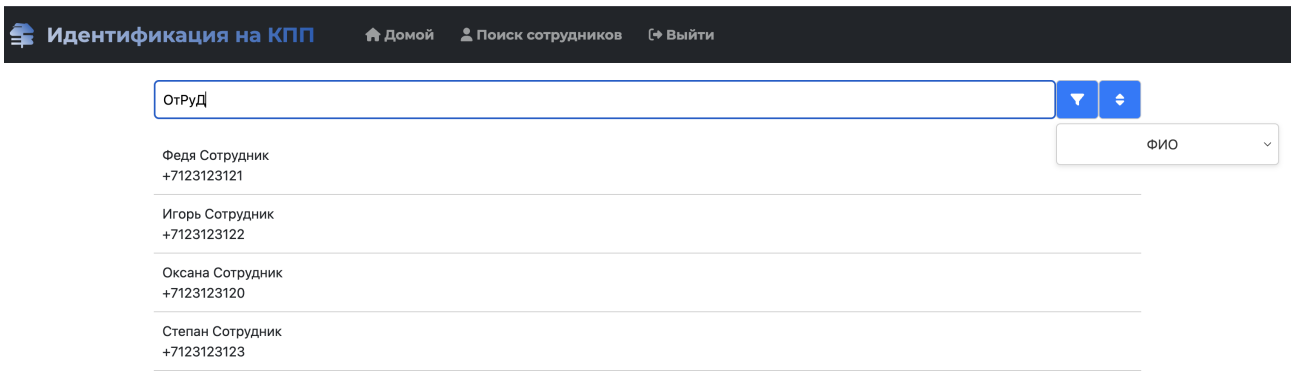

Рисунок 3.4 – Пример поиска информации о сотруднике представителем СБ

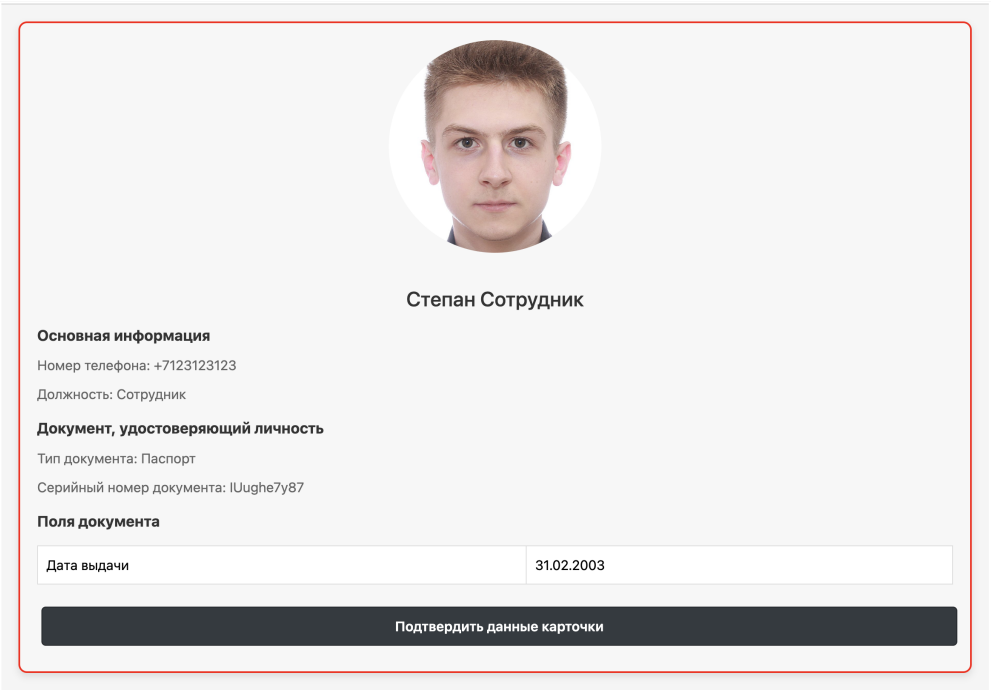

Рисунок 3.5 – Пример неподтвержденной карточки сотрудника

Далее, если данные корректны, сотрудник СБ может подтвердить карточку, как показано на рисунке 3.6.

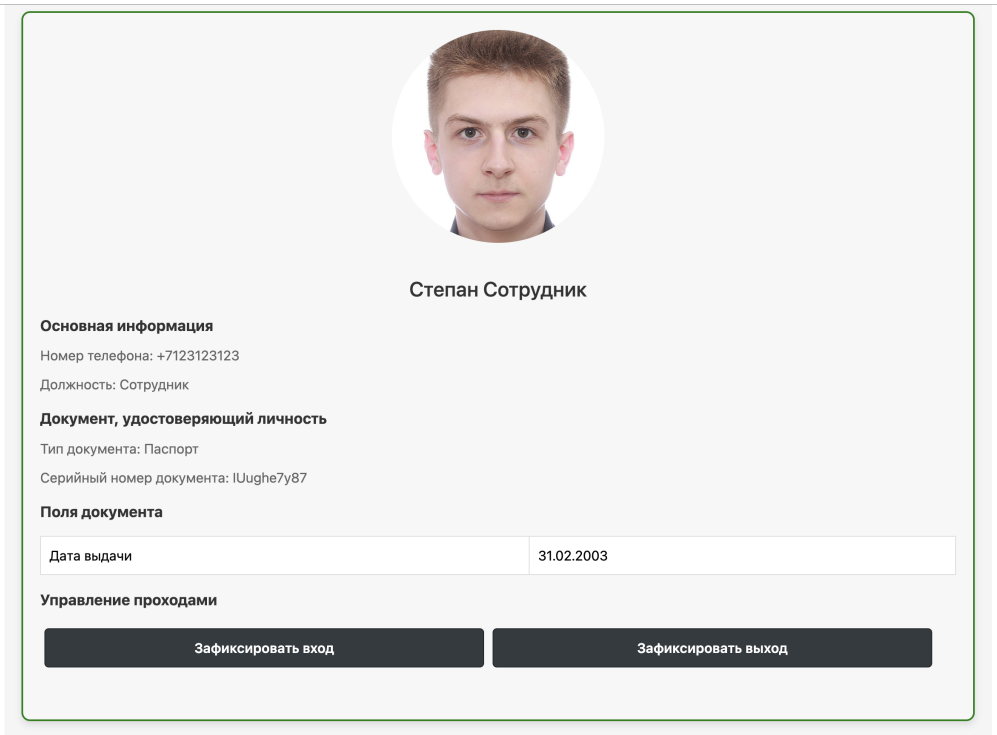

Рисунок 3.6 – Пример подтвержденной карточки сотрудника

После подтверждения появляется возможность логировать информацию о деятельности сотрудника на предприятии через КПП (рисунок 3.7).

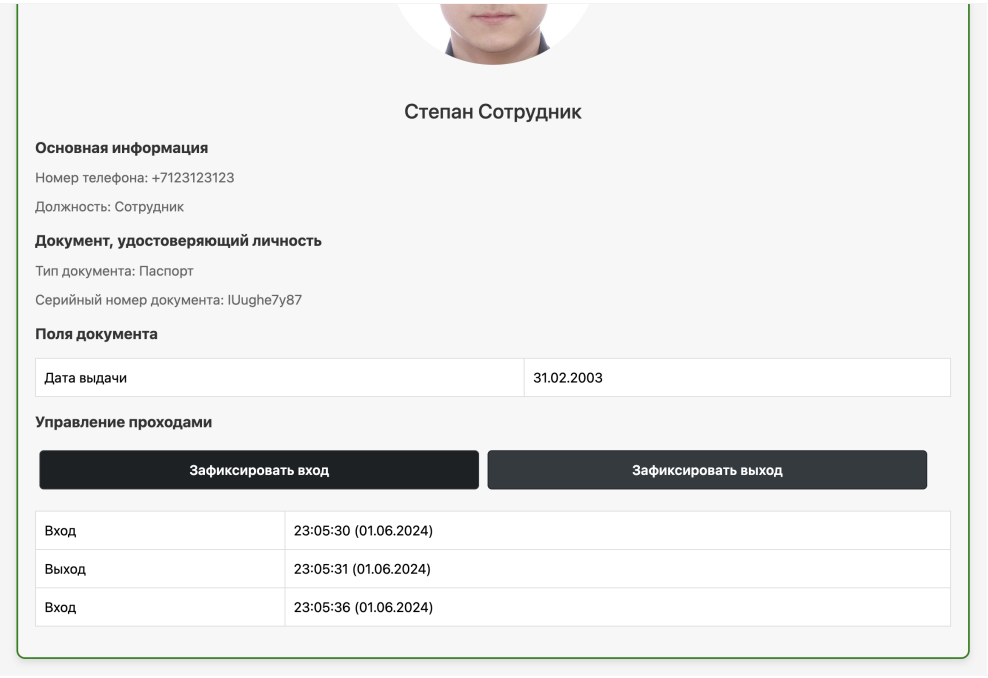

Рисунок 3.7 – Пример логирования информации о деятельности сотрудника

Вывод

В технологическом разделе были выбраны средства реализации базы данных и приложения (в том числе выбор СУБД и средств для хранения изображений) и описаны интерфейсы доступа к базе данных.

4 Исследовательский раздел

4.1 Описание исследования

Целью исследования является определение зависимости времени получения и сохранения изображений различных размеров от типа СУБД (на примере PostgreSQL и MongoDB). Для каждого изображения время получения и вставки замеряется 100 раз, после чего берется среднее значение.

4.2 Результаты исследования

Результаты исследования времени сохранения изображений различных размеров в зависимости от типа СУБД представлены в таблице 4.1.

Таблица 4.1 – Количественные данные, полученные в результате исследования

Размер изображения (Кбайт)	Время сохранения изображения в PostgreSQL (мкс)	Время получения изображения в MongoDB (мкс)
19612	522108	49090
16208	368289	32572
11047	251669	22274
7494	173262	16101
5101	113389	13625
3482	67236	20543
2388	46250	15806
1647	34640	10678
1060	20348	7880
859	16409	7549
477	9295	5662
325	6693	4208
224	5060	3729
156	4417	3372

На рисунке 4.1 представлен график зависимости времени сохранения изображений различных размеров от типа СУБД.

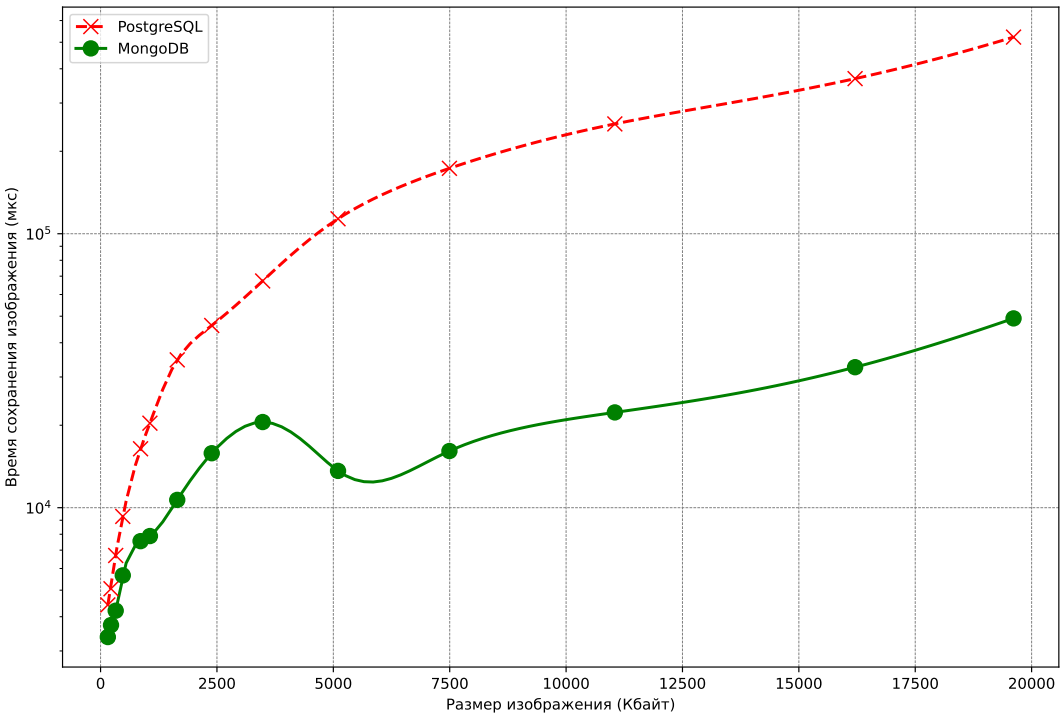

Рисунок 4.1 – Зависимость времени сохранения изображений различных размеров от типа СУБД

Результаты исследования времени получения изображений различных размеров в зависимости от типа СУБД представлены в таблице 4.2.

Таблица 4.2 – Количественные данные, полученные в результате исследования

Размер изображения (Кбайт)	Время получения изображения в PostgreSQL (мкс)	Время получения изображения в MongoDB (мкс)
19612	10911	195
16208	7914	155
11047	6240	119
7494	5316	79
5101	3718	70
3482	3408	64
2388	1971	57
1647	1466	53
1060	614	48
859	590	45
477	279	45
325	182	45
224	139	44
156	111	44

На рисунке 4.2 представлен график зависимости времени получения изображений различных размеров от типа СУБД.

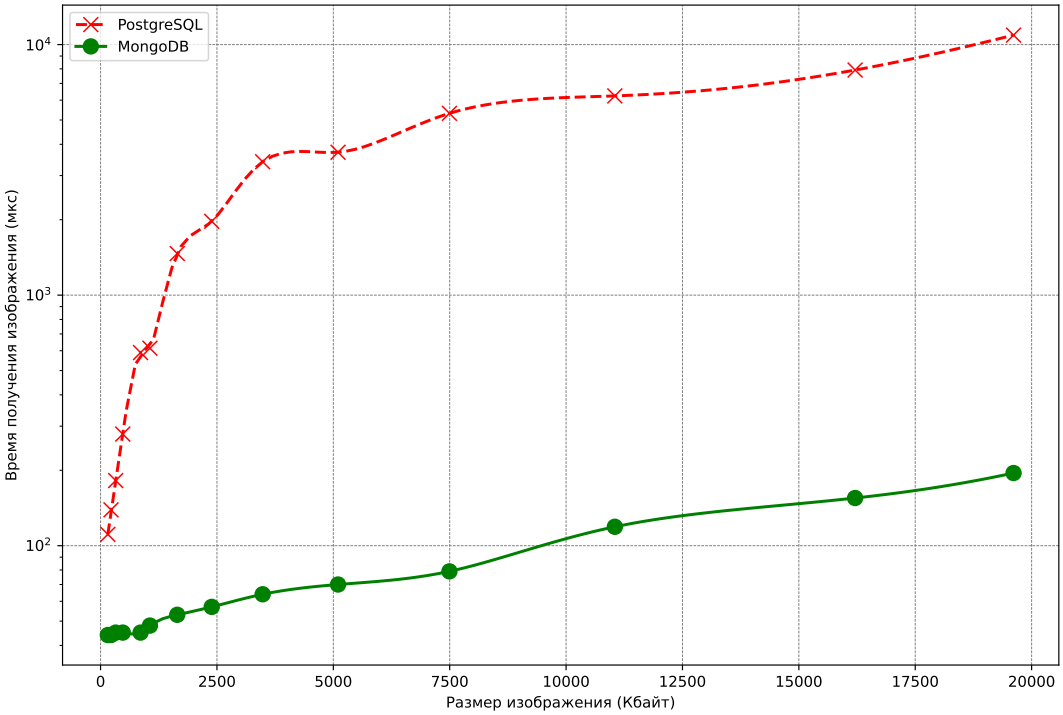

Рисунок 4.2 – Зависимость времени получения изображений различных размеров от типа СУБД

Вывод

В исследовательском разделе было проведено исследование зависимости времени получения и сохранения изображения от типа СУБД (на примере PostgreSQL и MongoDB).

Исходя из полученных в таблицах 4.1, 4.2 результатов, был сделан вывод, что время сохранения изображений, размер которых превышает 5000 Кбайт, в СУБД MongoDB в ∼11 раз меньше, чем в СУБД PostgreSQL, которая относится к типу реляционных баз данных, и в ∼1.5–3 раза меньше для меньших размерностей.

Аналогично время получения изображений, размер которых больше 3000 КБайт, в СУБД MongoDB в ∼50 раз меньше, чем в СУБД PostgreSQL, и в ∼5–17 раз меньше для меньших размерностей.

ЗАКЛЮЧЕНИЕ

Цель работы, заключавшаяся в разработке базы данных для идентификации пользователей при осуществлении контрольно-пропускного режима с использованием документов, удостоверяющих личность, была достигнута. Для достижения поставленной цели были решены следующие задачи:

1) формализована задача и данные;
2) проведен анализ существующих СУБД и методов хранения изображений на основе формализованной задачи, спроектирована диаграмма вариантов использования;
3) спроектирована диаграмма базы данных, ограничения целостности и ролевая модель;
4) выбраны средства реализации базы данных и приложения (в том числе СУБД и средства для хранения изображений);
5) разработана база данных и приложение;
6) описан интерфейс доступа к базе данных;
7) проведено исследование зависимости времени получения и сохранения изображения от типа СУБД.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Статья: Электронный пропускной режим: контроль рабочего времени. — [Электронный ресурс]. — Режим доступа: https://ric501.ru/wp-content/uploads/sites/29/2021/06/Statya-Elektronnyj-propusknoj-rezhim-kontrol-rabochego-vrem.pdf (дата обращения: 23.03.24).
2. *М. М. Крикунов*, *А. Н. Поручиков*. ОСНОВЫ БАЗ ДАННЫХ. //. — 2021. — С. 84.
3. *М. Клеппман*. Высоконагруженные приложения. Программирование, масштабирование, поддержка. //. — 2018. — С. 640.
4. Popularity Ranking of Database Management Systems. — [Электронный ресурс]. — Режим доступа: https://arxiv.org/pdf/2301.00847.pdf (дата обращения: 24.03.24).
5. Проектирование информационной обучающей системы выбор СУБД и способа доступа к данным. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_36316536_76885812.pdf (дата обращения: 24.03.24).
6. About MySQL. — [Электронный ресурс]. — Режим доступа: https://www.oracle.com/mysql/what-is-mysql/ (дата обращения: 24.03.24).
7. Сравнение различных СУБД. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_26575086_73187539.pdf (дата обращения: 24.03.24).
8. СРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕЛЯЦИОННЫХ СУБД, ПРИМЕНЯЕМЫХ ДЛЯ WEB - ПРИЛОЖЕНИЯ. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_48375110_15267617.pdf (дата обращения: 24.03.24).
9. СРАВНЕНИЕ MYSQL и POSTGRESQL. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_58803813_49724862.pdf (дата обращения: 24.03.24).
10. About Microsoft SQL Server. — [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16 (дата обращения: 25.03.24).

11. Выбор системы управления базами данных для программного комплекса приемной комиссии ДонНТУ. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_42755755_59333779.pdf (дата обращения: 25.03.24).

12. About PostgreSQL. — [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/about/ (дата обращения: 24.03.24).

13. PostgreSQL vs MySQL: The Critical Differences. — [Электронный ресурс]. — Режим доступа: https://www.integrate.io/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/ (дата обращения: 27.03.24).

14. Системы управления данными категории NoSQL. — [Электронный ресурс]. — Режим доступа: https://www.ispras.ru/publications/nosql_data_management_systems.pdf (дата обращения: 24.03.24).

15. Сравнение MongoDB и Redis. — [Электронный ресурс]. — Режим доступа: https://www.mongodb.com/compare/mongodb-vs-redis (дата обращения: 25.03.24).

16. СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ РЕЛЯЦИОННЫХ И НЕРЕЛЯЦИОННЫХ БАЗ ДАННЫХ НА ПРИМЕРЕ MYSQL И MONGODB. — [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_29650154_63824441.pdf (дата обращения: 25.03.24).

17. A Distributed Storage and Access Approach for Massive Remote Sensing Data in MongoDB. — [Электронный ресурс]. — Режим доступа: https://www.mdpi.com/2220-9964/8/12/533 (дата обращения: 27.03.24).

18. A COMPARISON BETWEEN NOSQL AND RDBMS: STORAGE AND RETRIEVAL. — [Электронный ресурс]. — Режим доступа: https://www.minarjournal.com/dergi/a-comparison-between-nosql-and-rdbms-storage-and-retrieva20230620121910.pdf (дата обращения: 27.03.24).

19. About Go. — [Электронный ресурс]. — Режим доступа: https://go.dev/project (дата обращения: 13.04.24).

20. The Progressive JavaScript Framework. — [Электронный ресурс]. — Режим доступа: https://vuejs.org/ (дата обращения: 13.04.24).

21. PL/pgSQL — SQL Procedural Language. — [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/docs/current/plpgsql.html (дата обращения: 13.04.24).

22. About Go driver and toolkit for PostgreSQL. — [Электронный ресурс]. — Режим доступа: https://pkg.go.dev/github.com/jackc/pgx/v5 (дата обращения: 13.04.24).

23. About MongoDB Driver API for Go. — [Электронный ресурс]. — Режим доступа: https://pkg.go.dev/go.mongodb.org/mongo-driver/mongo (дата обращения: 13.04.24).

24. About JWT. — [Электронный ресурс]. — Режим доступа: https://jwt.io/introduction (дата обращения: 13.04.24).

ПРИЛОЖЕНИЕ А

Листинг А.1 – Запросы для создания таблиц базы данных с ограничениями целостности

```
create table if not exists company
(
    id serial primary key,
    name text,
    city text
);

create table if not exists employee
(
    id serial primary key,
    phone_number text unique,
    full_name text,
    company_id int references company(id),
    post text,
    password text,
    refresh_token text,
    token_expired_at timestamp,
    date_of_birth date
);

create table if not exists info_card
(
    id serial primary key,
    created_employee_id int references employee(id),
    is_confirmed boolean,
    created_date date default now()
);

create table if not exists document
(
    id serial primary key,
    serial_number text,
    info_card_id int references info_card(id),
    type text
);
```

```
create table if not exists photo
(
    id serial primary key,
    document_id int references document(id),
    key text
);

create table if not exists field
(
    id serial primary key,
    document_id int references document(id),
    type text,
    value text,

    unique (document_id, type)
);

create table if not exists checkpoint
(
    id serial primary key,
    phone_number text
);

create table if not exists passage
(
    id serial primary key,
    checkpoint_id int references checkpoint(id),
    document_id int references document(id),
    type text,
    time timestamp
);
```

Листинг А.2 – Запросы для создания триггера базы данных

```
create or replace function employee_passages_for_day
    (passage_document_id int, check_date date)
    returns table
            (
                document_id   int,
                entries_count bigint,
                exits_count   bigint
            ) as $$
begin
```

```
    if exists(
        select 1
        from passage p
        where
            p.document_id = passage_document_id
          and date(p.time) = check_date
        group by p.document_id
    ) then
        return query
            select
                p.document_id,
                count(case when p.type = 'Вход' then 1 end) as
                   entries_count,
                count(case when p.type = 'Выход' then 1 end) as
                   exits_count
            from passage p
            where
                p.document_id = passage_document_id
              and date(p.time) = check_date
            group by p.document_id;
    else
        return query
            select passage_document_id, 0::bigint, 0::bigint;
    end if;
end;
$$ language plpgsql;

create or replace function log_daily_employee_passages()
    returns trigger as
$$
declare
    document_id   int;
    entries_count bigint;
    exits_count   bigint;
begin
    select *
    into
        document_id,
        entries_count,
        exits_count
    from employee_passages_for_day(
```

```
            new.document_id,
            current_date
        );

    raise notice E'За % по документу с идентификатором % ' ||
        E'сотрудник: \n- вошел % раз(а)\n- вышел % раз(а)',
        current_date, document_id, entries_count, exits_count;
    return new;
end;
$$ language plpgsql;

create or replace trigger day_passages
    after insert on passage
    for each row
execute function log_daily_employee_passages();
```