Приложение А – Техническое задание

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**СИСТЕМА УПРАВЛЕНИЯ ДОСТУПОМ К БАЗЕ ДАННЫХ ДЛЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ «УЧЕТ ПОСЕЩАЕМОСТИ СТУДЕНТОВ»**

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (С.Н. Касьяненко) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Н.А. Райков) |

Иркутск 2025

**1. Введение**

**1.1.** **Общие сведения**

Документ представляет собой техническое задание на создание модуля управления доступом к базе данных для информационной системы «Учёт посещаемости студентов» на основе PostgreSQL. Система должна обеспечить гибкое разграничение прав пользователей с различными уровнями доступа, что важно для защиты и корректной работы с данными.

**1.2.** **Цели и задачи**

Цель проекта – разработка системы контроля доступа к базе данных PostgreSQL в рамках информационной системы «Учёт посещаемости студентов».

Задачи проекта:

1. Разработать структуру БД для учёта посещаемости.
2. Назначить роли и привилегии для 5 пользователей.
3. Настроить доступ с помощью SQL-команд.
4. Составить и оформить примеры SQL-команд.
5. Проверить корректность разграничения прав на тестовых данных.

**2. Основания для разработки**

**2.1.** **Нормативные документы**

Документ основывается на следующих нормативных документах:

* ГОСТ 34.602-2020 «Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы».
* ГОСТ Р 56477-2015 «Проектирование и внедрение информационных систем. Общие требования».

**2.2.** **Проектные документы**

Проектные документы включают:

* Пояснительную записку.

**3. Назначение системы**

**3.1. Общее описание**

Модуль отвечает за разграничение прав пользователей (например, администратор, преподаватель, секретарь, студент, гость), что позволяет обеспечить безопасную работу с таблицами БД. Основные действия – это чтение, добавление, обновление данных, ограниченный доступ и логическая изоляция пользователей.

**3.2.** **Преимущества и новизна**

* + Гибкое распределение прав.
  + Простота в использовании.
  + Надежная защита информации.
  + Возможность расширения под другие модули системы.

**4. Требования к системе**

**4.1.** **Функциональные требования**

* Система должна поддерживать создание базы данных и таблиц, предназначенных для фиксации посещаемости студентов.
* Модуль обязан обеспечивать разграничение прав доступа для не менее чем 5 категорий пользователей, включая:

1. Администратор – полный набор прав.
2. Преподаватель – возможность внесения и изменения данных.
3. Секретарь – доступ только к просмотру записей.
4. Студент – ограниченный доступ только к личной информации.
5. Гость – минимальный уровень доступа.

* Предусматривается управление правами пользователей посредством SQL-запросов.
* Примеры SQL-команд должны быть оформлены и включены в приложение к настоящему техническому заданию.

**4.2.** **Технические требования**

* В качестве системы управления базами данных используется PostgreSQL версии 13 и выше.
* Необходимо спроектировать и реализовать следующие сущности:

1. Студенты (Students).
2. Курсы (Courses).
3. Посещаемость (Attendance).

* Техническое задание и файлы реализации должны быть размещены в открытом репозитории GitHub.

**4.3. Эксплуатационные требования**

* Система должна обеспечивать стабильную работу при одновременном подключении не менее 10 пользователей.
* Время выполнения команды управления доступом не должно превышать 100 мс.
* Система должна быть доступна 99.9% времени.

**5. Требования к техническому обеспечению**

**5.1. Оборудование**

* Сервер с не менее 4 ГБ оперативной памяти
* Не менее 20 ГБ дискового пространства

**5.2. Сетевые требования**

* Доступ к сети Интернет со скоростью не менее 10 Мбит/с
* Поддержка защищенных соединений к базе данных (SSL/TLS).

**6. Требования к программному обеспечению**

**6.1. Программные компоненты**

* PostgreSQL 13 или выше.
* Git.
* pgAdmin.

**6.2. Интерфейсы**

* Использование командной оболочки PostgreSQL (psql) для выполнения операций управления доступом.
* Подключение к базе данных должно поддерживаться через общепринятые драйверы PostgreSQL, такие как psycopg2.

**7. Организационно-технические требования**

**7.1. Этапы разработки**

В таблице 1 представлены сроки и этапы разработки модуля.

Таблица 1 – Сроки и этапы разработки модуля

|  |  |
| --- | --- |
| Наименование этапов | Срок |
| Проектирование структуры базы данных | 05.04.2025 |
| Создание таблиц и тестовых данных | 07.04.2025 |
| Определение ролей и прав доступа | 10.04.2025 |
| Реализация команд управления доступом | 11.04.2025 |
| Тестирование системы | 14.04.2025 |
| Подготовка приложения с примерами | 17.04.2025 |

**Приложение Б – Примеры SQL-команд для реализации модуля управления доступом к базе данных**

Данное приложение содержит фрагменты SQL-кода, отражающие основные этапы создания структуры базы данных и настройки разграничения прав пользователей в информационной системе «Учёт посещаемости студентов», согласно техническому заданию.

## **1 Создание ролей пользователей**

CREATE ROLE admin\_user LOGIN PASSWORD 'admin\_pass';  
CREATE ROLE teacher\_user LOGIN PASSWORD 'teacher\_pass';  
CREATE ROLE secretary\_user LOGIN PASSWORD 'secretary\_pass';  
CREATE ROLE student\_user LOGIN PASSWORD 'student\_pass';  
CREATE ROLE guest\_user LOGIN PASSWORD 'guest\_pass';

## **2 Создание базы данных**

CREATE DATABASE pm\_07 OWNER admin\_user;

## **3 Структура таблиц базы данных**

-- Таблица students  
CREATE TABLE students (  
 id SERIAL PRIMARY KEY,  
 full\_name VARCHAR(100) NOT NULL,  
 email VARCHAR(100) UNIQUE NOT NULL,  
 group\_name VARCHAR(50),  
 phone VARCHAR(20),  
 password\_hash TEXT NOT NULL  
);  
  
-- Таблица courses  
CREATE TABLE courses (  
 id SERIAL PRIMARY KEY,  
 course\_name VARCHAR(100) NOT NULL UNIQUE,  
 description TEXT  
);  
  
-- Таблица attendance  
CREATE TABLE attendance (  
 id SERIAL PRIMARY KEY,  
 student\_id INTEGER NOT NULL REFERENCES students(id) ON DELETE CASCADE,  
 course\_id INTEGER NOT NULL REFERENCES courses(id) ON DELETE CASCADE,  
 date\_attended DATE NOT NULL,  
 status VARCHAR(20) NOT NULL CHECK (status IN ('присутствовал', 'отсутствовал', 'опоздал'))  
);

## **4 Назначение прав доступа**

-- Администратор  
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO admin\_user;  
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO admin\_user;  
  
-- Преподаватель  
GRANT SELECT, INSERT, UPDATE ON students TO teacher\_user;  
GRANT SELECT, INSERT, UPDATE ON courses TO teacher\_user;  
GRANT SELECT, INSERT, UPDATE ON attendance TO teacher\_user;  
  
-- Секретарь  
GRANT SELECT ON students TO secretary\_user;  
GRANT SELECT ON courses TO secretary\_user;  
GRANT SELECT ON attendance TO secretary\_user;  
  
-- Студент  
GRANT SELECT ON students TO student\_user;  
GRANT SELECT ON attendance TO student\_user;  
  
-- Гость  
CREATE VIEW guest\_view\_students AS  
SELECT full\_name, group\_name FROM students;  
GRANT SELECT ON guest\_view\_students TO guest\_user;

## **5 Права на последовательности (serial)**

GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO teacher\_user;  
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO secretary\_user;  
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO student\_user;