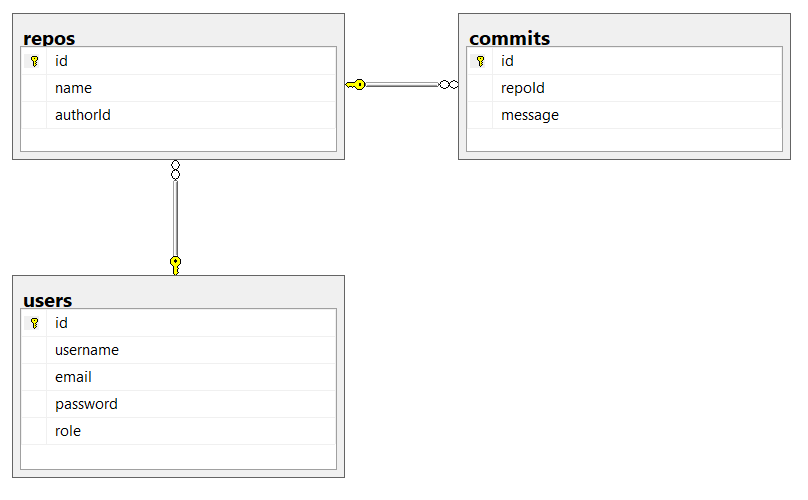
Лабораторная работа 19

ПСКП

ПОИТ-3

**Задание 01**

1. Разработайте приложение **19-01**, представляющее собой EXPRESS-HTTP-сервер, прослушивающий порт ***3000***.
2. Приложение **19-01** предназначено для демонстрации управления доступом на основе ролей.
3. В приложении **19-01** должны быть определены 3 сущности (User, Repo и Commit). В качестве СУБД использовать MS SQL Server или MySQL. Для работы с базой данных использовать пакет **sequelize** или **prisma**.



1. Приложение **19-01** должно реагировать на запросы к следующим ресурсам (использовать в качестве основы приложение **17-02**).

|  |  |  |
| --- | --- | --- |
| Метод  запроса | URI | Назначение |
| GET | /login | То же, что и в приложении **17-02.** |
| POST | /login |
| GET | /register |
| POST | /register |
| GET | /logout |
| GET | api/ability | набор привилегий текущего пользователя в формате json |
| GET | api/user | получить список всех пользователей (без пароля) в json-формате |
| GET | api/user/:id | id – идентификатор пользователя, получить информацию о пользователе (без пароля) с идентификатором id |
| GET | api/repos | получить список всех репозиториев в json-формате |
| GET | api/repos/:id | id – идентификатор репозитория, получить информацию об указанном репозитории |
| POST | api/repos | добавить новый репозиторий, данные в json-формате |
| PUT | api/repos/:id | id – идентификатор репозитория, корректировать информацию об указанном репозитории, данные в json-формате |
| DELETE | api/repos/:id | id – идентификатор репозитория, удалить указанный репозиторий |
| GET | api/repos/:id/commits | id – идентификатор репозитория, получить список всех коммитов к указанному репозиторию в json-формате |
| GET | api/repos/:id/commits/:commitId | id – идентификатор репозитория, commitId – идентификатор коммита, получить информацию об указанном коммите |
| POST | api/repos/:id/commits/ | добавить новый коммит к указанному репозиторию, данные в json-формате |
| PUT | api/repos/:id/commits/:commitId | id – идентификатор репозитория, commitId – идентификатор коммита, корректировать информацию об указанном коммите, данные в json-формате |
| DELETE | api/repos/:id/commits/:commitId | id – идентификатор репозитория, commitId – идентификатор коммита, удалить указанный коммит |
| \* | Остальные URI | Вернуть ответ со статусом 404 или 405 (если неподдерживаемый метод) |

1. Добавить в приложение обработку ошибок. В случае возникновения ошибки возвращать клиенту сообщение с соответствующим статус кодом и пояснением.
2. В приложении **19-01** должна быть поддержка 3 ролей: гость, зарегистрированный пользователь и администратор.
3. Приложение **19-01** должно предоставлять доступ к API по приведенным правилам. Для управления правами использовать пакет **casl**.

|  |  |
| --- | --- |
| Роль | Привилегии |
| Гость | * просматривать свои привилегии (/ability), коммиты и репозитории |
| Зарегистрированный пользователь | * просматривать свои привилегии (/ability), информацию о себе, коммиты и репозитории * создавать репозитории * создавать коммиты к своим репозиториям * редактировать свои репозитории и коммиты |
| Администратор | * просматривать свои привилегии (/ability), информацию о всех пользователях, коммиты и репозитории * редактировать любые репозитории и коммиты * удалять любые репозитории и коммиты |

1. Определение привилегий осуществлять в глобальном middleware.
2. Тестирование выполнить с помощью браузера и Postman.

**Задание 02** Ответьте на следующие вопросы

1. Поясните понятия «авторизация», «роль», «привилегия», «ACL».
2. Что такое casl?
3. Перечислите известные модели управления доступом.

10. \*\*Понятия:\*\*

- \*\*Авторизация (Authorization):\*\* Это процесс проверки прав доступа пользователя к определенным ресурсам или действиям в системе. Авторизация обычно следует за аутентификацией (Authentication), которая подтверждает личность пользователя. После аутентификации система определяет, какие действия или ресурсы пользователь имеет право использовать, основываясь на его роли или других атрибутах.

- \*\*Роль (Role):\*\* Роль определяет набор разрешений или привилегий, которые назначаются пользователю в системе. Роль определяет тип доступа, который пользователь может иметь к различным функциям или ресурсам. Например, роль "администратор" может иметь доступ ко всем функциям системы, тогда как роль "пользователь" имеет ограниченные права.

- \*\*Привилегия (Privilege):\*\* Привилегия определяет конкретное разрешение на выполнение определенных действий или доступ к ресурсам в системе. Привилегии могут быть связаны с определенными ролями или присваиваться непосредственно пользователям.

- \*\*ACL (Access Control List):\*\* Это список разрешений, который определяет, какие пользователи или группы пользователей имеют доступ к определенным объектам или действиям в системе. ACL представляет собой набор разрешений, связанных с определенными ресурсами или объектами.

11. \*\*CASL (с англ. "CASL")\*\*:

CASL (созданный как стандартный язык управления доступом) является библиотекой для создания сложных правил авторизации на основе разных факторов, таких как роли пользователя, типы ресурсов и контекст запроса. CASL позволяет определять разрешения на доступ к объектам в более гибком и декларативном формате. Он позволяет создавать правила авторизации на основе контекста запроса, а не только на основе статических ролей или привилегий.

12. \*\*Модели управления доступом:\*\*

Существует несколько моделей управления доступом, которые определяют способы предоставления доступа пользователям к ресурсам в системе. Некоторые из известных моделей включают:

- \*\*RBAC (Role-Based Access Control):\*\* Это модель управления доступом, основанная на ролях. Пользователи получают доступ к ресурсам в системе на основе их роли. Роли определяют набор разрешений, которые пользователь получает при назначении этой роли.

- \*\*ABAC (Attribute-Based Access Control):\*\* Это модель управления доступом, которая основана на атрибутах пользователей, ресурсов и контекста запроса. Пользователям предоставляется доступ на основе атрибутов (например, возраст, должность, отдел) и условий, заданных в политиках доступа.

- \*\*PBAC (Policy-Based Access Control):\*\* Это модель, в которой доступ к ресурсам определяется с помощью политик доступа. Политики могут быть выражены на языке выражений или правил, которые определяют разрешения на основе атрибутов и контекста запроса.

- \*\*MAC (Mandatory Access Control):\*\* Это модель управления доступом, в которой доступ к ресурсам определяется на основе уровней безопасности и меток доступа, назначаемых ресурсам и пользователям. MAC часто используется в высокоуровневых системах безопасности, таких как государственные или военные сети.

Каждая из этих моделей управления доступом имеет свои особенности и подходы к определению и контролю доступа в информационных системах.