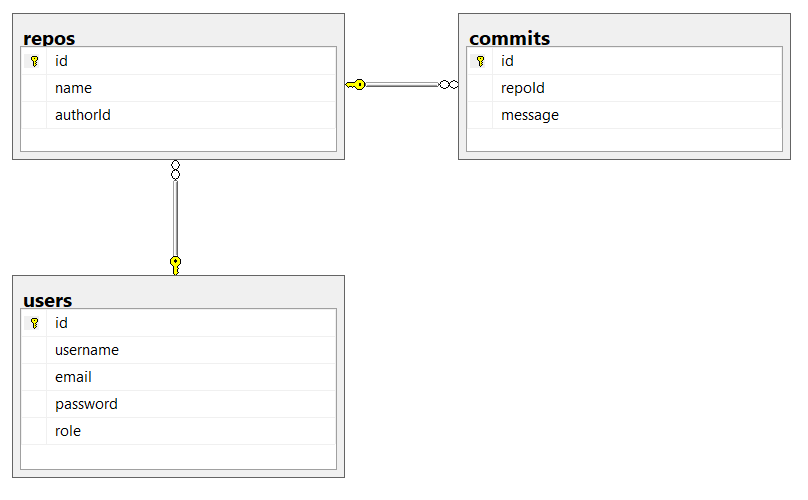
Лабораторная работа 19

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.
4. Вот объяснения понятий:

* Авторизация: Это процесс проверки и подтверждения прав доступа пользователя к определенным ресурсам или функциям системы. Авторизация определяет, имеет ли пользователь достаточные права для выполнения определенных действий или получения доступа к определенным данным или ресурсам.
* Роль: Роль - это набор правил, предназначенных для группировки пользователей с общими привилегиями или обязанностями в системе. Роль определяет, какие действия и ресурсы доступны пользователю, принадлежащему этой роли. Например, в системе управления содержимым (Content Management System) может быть роль "Администратор", которая имеет право создавать, редактировать и удалять контент, а также роль "Пользователь", которая может только просматривать контент.
* Привилегия: Привилегия - это определенное разрешение или право, которое предоставляется пользователю или роли для доступа к определенным действиям, ресурсам или функциям системы. Привилегии определяют, какие конкретные действия могут быть выполнены или ресурсы, к которым можно получить доступ. Например, в системе управления базами данных привилегия "SELECT" позволяет пользователю выполнять операцию выборки данных из таблицы.
* ACL (Access Control List): Список контроля доступа (Access Control List) - это механизм управления доступом, используемый для определения прав доступа к объектам или ресурсам. ACL содержит список субъектов (например, пользователей или групп) и их соответствующие права доступа к объектам. Этот список может включать различные разрешения, такие как чтение, запись, выполнение и т. д. Когда пользователь запрашивает доступ к объекту, система проверяет его идентификатор в ACL, чтобы определить, должен ли пользователь получить доступ к объекту и с какими правами.

1. CASL (Capability-based Security Language) - это язык, разработанный для описания политик безопасности на основе принципов, известных как "способности" (capabilities). В рамках модели безопасности на основе способностей каждый субъект имеет набор способностей (capabilities), которые определяют, какие действия он может выполнять. Этот подход к безопасности позволяет более гранулярно управлять доступом и предотвращает распространение привилегий.
2. Некоторые известные модели управления доступом включают:

* Модель дискреционного управления доступом (Discretionary Access Control, DAC): В этой модели права доступа к ресурсам определяются владельцем ресурса. Владелец решает, кому предоставить доступ к ресурсу и с какими правами.
* Модель мандатного управления доступом (Mandatory Access Control, MAC): В этой модели права доступа определяются на основе определенных политик безопасности и уровней секретности. Например, система может определить, что только пользователи с определенным уровнем секретности имеют доступ к определенным данным.
* Модель ролевого управления доступом (Role-Based Access Control, RBAC): В этой модели доступ основывается на ролях, которые назначаются пользователям. Каждая роль имеет определенные привилегии, и пользователи получают доступ к ресурсам на основе своих ролей.
* Модель полномочий на основе атрибутов (Attribute-Based Access Control, ABAC): В этой модели доступ определяется на основе атрибутов пользователя, ресурса и контекста. Различные атрибуты, такие как идентификатор пользователя, местоположение, время и другие факторы, используются для принятия решений о предоставлении доступа.
* Модель политики управления доступом (Policy-Based Access Control, PBAC): Это гибкая модель, в которой доступ определяется на основе набора правил или политик. Политики определяют, какие действия разрешены или запрещены для определенных пользователей или ролей в определенных контекстах.

Это лишь некоторые известные модели управления доступом, и существуют и другие подходы и модели, разработанные для обеспечения безопасности и управления доступом в информационных системах.