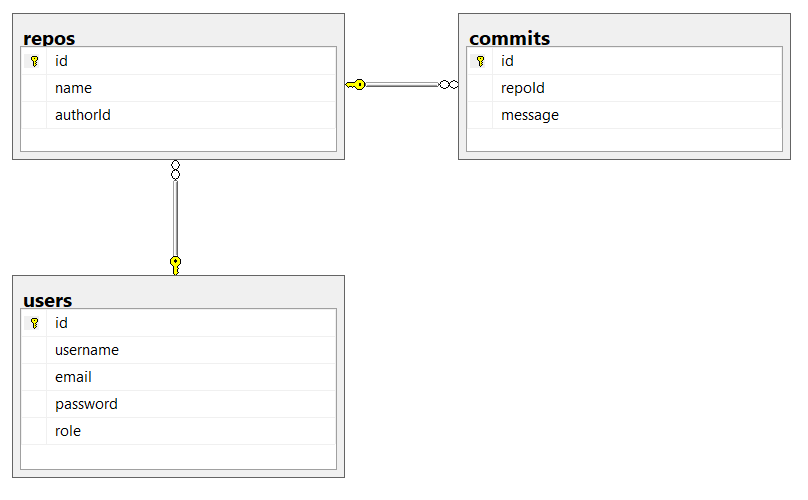
Лабораторная работа 10

СТРWП

**Задание 01**

1. Разработайте приложение **10-01**, представляющее собой EXPRESS-HTTP-сервер, прослушивающий порт ***3000***.
2. Приложение **10-01** предназначено для демонстрации управления доступом на основе ролей.
3. В приложении **10-01** 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 **10-01** должно реагировать на запросы к следующим ресурсам (использовать в качестве основы приложение **08-02**).

|  |  |  |
| --- | --- | --- |
| Метод  запроса | URI | Назначение |
| GET | /login | То же, что и в приложении **08-02.** |
| 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 **10-01** должна быть поддержка 3 ролей: гость, зарегистрированный пользователь и администратор.
3. Приложение **10-01** 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. **Авторизация** - это процесс проверки прав доступа пользователя к определенным ресурсам или функциональности системы. Он определяет, имеет ли пользователь право выполнить определенное действие, такое как чтение, запись, удаление и т. д. на конкретном ресурсе или в определенной области системы.

**Роль** - это набор разрешений, который назначается пользователю в системе. Роль определяет набор действий, которые пользователь может выполнять в системе в зависимости от своей роли. Например, пользователь может иметь роль администратора, модератора или обычного пользователя.

**Привилегия** - это конкретное разрешение или право, предоставленное пользователю или роли для выполнения определенного действия или доступа к определенному ресурсу. Например, привилегия может быть разрешением на чтение, запись, удаление или выполнение определенной операции.

**ACL (Access Control List)** - это список разрешений, который определяет, какие пользователи или роли имеют доступ к определенным ресурсам или функциональности системы. Каждый элемент списка содержит информацию о том, какие действия разрешены или запрещены для конкретного пользователя или роли в отношении определенного ресурса.

1. **CASL** (Casbin) - это библиотека для управления доступом, разработанная на языке программирования Go. Она предоставляет инструменты для создания и управления различными моделями управления доступом, такими как RBAC (Role-Based Access Control), ABAC (Attribute-Based Access Control) и другими. CASL позволяет определять правила доступа на основе различных атрибутов пользователя и ресурсов, что делает его мощным инструментом для обеспечения безопасности в приложениях.
2. Известные модели управления доступом включают:

* **RBAC (Role-Based Access Control)** - модель управления доступом, основанная на ролях пользователей. В этой модели разрешения предоставляются пользователям на основе их роли в системе. Пользователи могут иметь одну или несколько ролей, каждая из которых определяет их права доступа к определенным ресурсам или функциональности.
* **ABAC (Attribute-Based Access Control)** - модель управления доступом, основанная на атрибутах субъектов, ресурсов и окружения. В этой модели разрешения определяются на основе комбинации атрибутов, таких как роль пользователя, тип ресурса, время доступа и другие атрибуты. ABAC обеспечивает более гибкий и динамичный подход к управлению доступом, чем RBAC.
* **MAC (Mandatory Access Control)** - модель управления доступом, в которой разрешения определяются администратором на основе политик безопасности. В этой модели каждый объект и субъект имеют метки безопасности, и доступ к ресурсам контролируется на основе их меток. MAC обеспечивает высокий уровень контроля и защиты, но требует более сложной настройки и управления.
* **DAC (Discretionary Access Control)** - модель управления доступом, в которой владелец ресурса имеет право управлять доступом к нему. В этой модели доступ к ресурсам контролируется владельцами ресурсов, которые могут предоставлять или отзывать разрешения на доступ к своим ресурсам. DAC обеспечивает гибкость и удобство в управлении доступом, но может привести к потенциальным проблемам безопасности, если у владельца недостаточно знаний или интереса к безопасности.