Проблема:

В нашем продукте есть несколько микросервисов. Для работы каждого из них требуется знать права пользователя, который запустил ту или иную задачу. На архитектурном комитете приняли решение централизовать работу с привилегиями пользователя и вынести в отдельный сервис.

Задача:

Необходимо реализовать микросервис для работы с привилегиями пользователей (создание/удаление пользователя, добавить/убрать права пользователя, проверка прав пользователя). Сервис должен предоставлять HTTP API и принимать/отдавать запросы/ответы в формате JSON.

Сценарии использования:

Далее описаны несколько упрощенных кейсов приближенных к реальности.

1. Сервис, отвечающий за распределение задач (task-manager) между агентами (agent), получил очередную задачу на выполнение - ему необходимо проверить имеет ли пользователь, запустивший данную задачу, права доступа к конкретному агенту
2. За каталогизацию архивов отвечает отдельный сервис(archive-manager) - пользователь хочет просмотреть все доступные ему записи об архивах.
3. В ближайшее время планируется интеграция с одним из провайдеров OpenID. Мы решили заранее предусмотреть такую возможность и заложить ее в архитектуру нашего сервиса.

Требования к коду:

1. Язык разработки: Go/C++. Мы готовы рассматривать решения на любом языке, но приоритетом для нас являются именно эти.
2. Фреймворки и библиотеки можно использовать любые
3. Реляционная СУБД: PostgreSQL
4. Весь код должен быть выложен на GitHub/GitLab с README файлом, содержащим инструкцию по запуску и примерами запросов/ответов
5. При возникновении вопросов по ТЗ оставляем принятие решения за кандидатом (в таком случае Readme файле к проекту должен быть указан список вопросов с которыми кандидат столкнулся и каким образом он их решил)
6. Разработка интерфейса в браузере НЕ ТРЕБУЕТСЯ. Взаимодействие с АПИ предполагается посредством запросов из кода другого сервиса. Для тестирования можно использовать любой удобный инструмент. Например: в терминале через curl или Postman.

Будет плюсом:

1. Использование docker и docker-compose для поднятия и развертывания dev-среды.
2. Методы АПИ возвращают человеко-читабельные описания ошибок и соответвующие статус коды при их возникновении.
3. Написаны unit/интеграционные тесты.
4. Реализован пакет-клиент к сервису.
5. Реализован GitHub Actions/GitLab CI для проекта.

Основное задание (минимум):

Метод создания пользователя с определенными доступами к конкретным агентам.

Метод проверки привелегий пользователя

Интеграция с одним с OpenID провайдером

Детали по заданию:

1. По умолчанию сервис не содержит в себе никаких данных о пользователях (пустая табличка в БД).

2. Сервисы, которые запрашивают данные о пользователях (task-manager, archive-manager) можно реализовать как мок объекты.
3. Валидацию данных и обработку ошибок оставляем на усмотрение кандидата.
4. Глубина проработки задания целиком и полностью инициатива кандидата