Projeto rasplus-api

O objetivo deste projeto é construir a aplicação do rasplus-api, desenvolvido nos cursos de spring rest e spring security para uma arquitetura de MS.

Sobre o Rasmoo Plus

O **Rasmoo Plus** é uma plataforma de cursos de programação baseada em assinatura, projetada para atender às necessidades de quem busca aprimorar suas habilidades na área. Os alunos podem escolher entre três tipos de planos: **mensal**, **anual** e **vitalício**, garantindo flexibilidade para se adaptar ao seu objetivo e orçamento.

Após fazer login, o usuário tem acesso à vasta biblioteca de cursos disponíveis e pode começar imediatamente suas aulas. Além disso, a plataforma permite a personalização do perfil, possibilitando a atualização de informações cadastrais de forma simples e prática.

Link das telas que serão desenvolvidas nos cursos de front

Figma

Repositório ws-rasmoo-plus
Repositório rasplus-api-keycloak

Para atender as telas do fluxo de login, criação e recuperação de conta, aŕea logada com a listagem dos cursos e edição de dados cadastrados, inicialmente, foi projeto o seguinte design

Módulo 2 - rasplus-customer-data-api

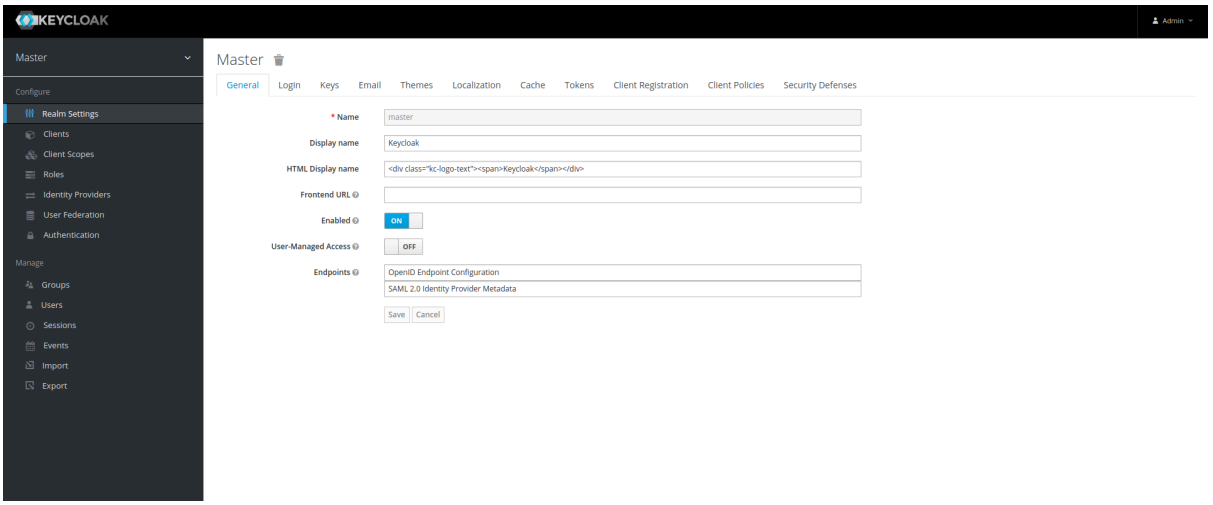
Rotas:

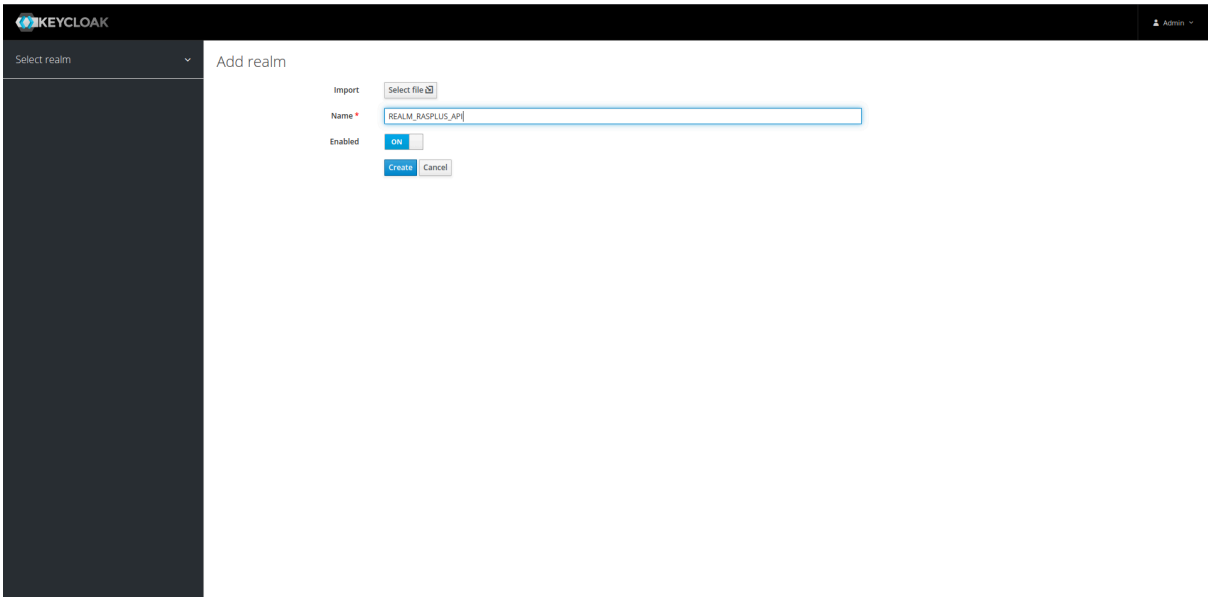
- (POST)/api/customer/v1/auth/: Realiza integração com o keycloak para autenticação
- (POST)/api/customer/v1/auth/recovery-code/send: Envia código de recuperação para atualizar credenciais
- (GET)/api/customer/v1/auth/recovery-code: Verifica se código de recuperação é válido
- (PATCH)/api/customer/v1/auth/recovery-code/password: Atualiza senha
- (POST)/api/customer/v1/auth/representations/credentials
- (PUT)/api/customer/v1/auth/representations/credentials/{email}

Github estrutura de branchs
“class_{class_numer}-{class_description}”

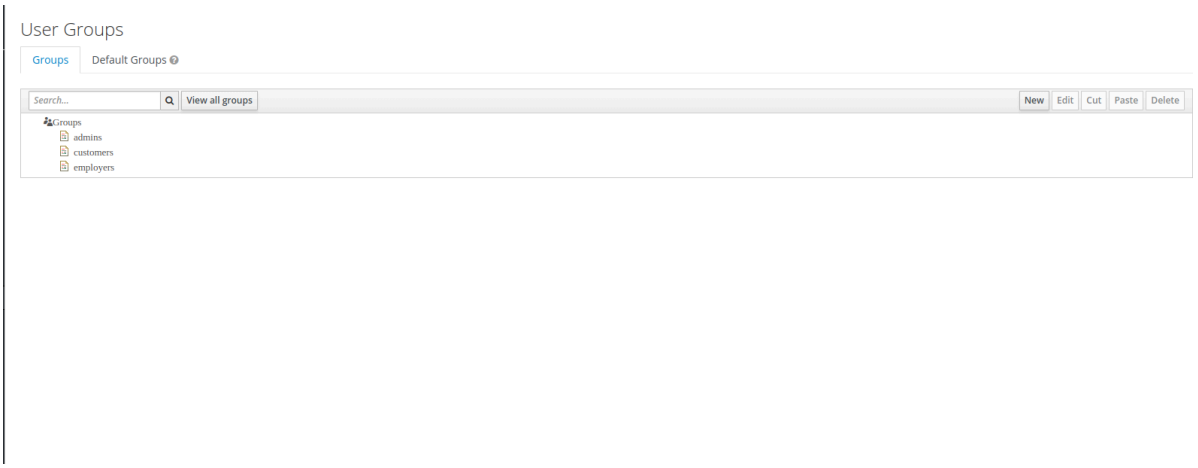
Configuração do Keycloak

1 - Crie o Realm REALM_RASPLUS_API - Clique em Master > add realm > REALM_RASPLUS_API

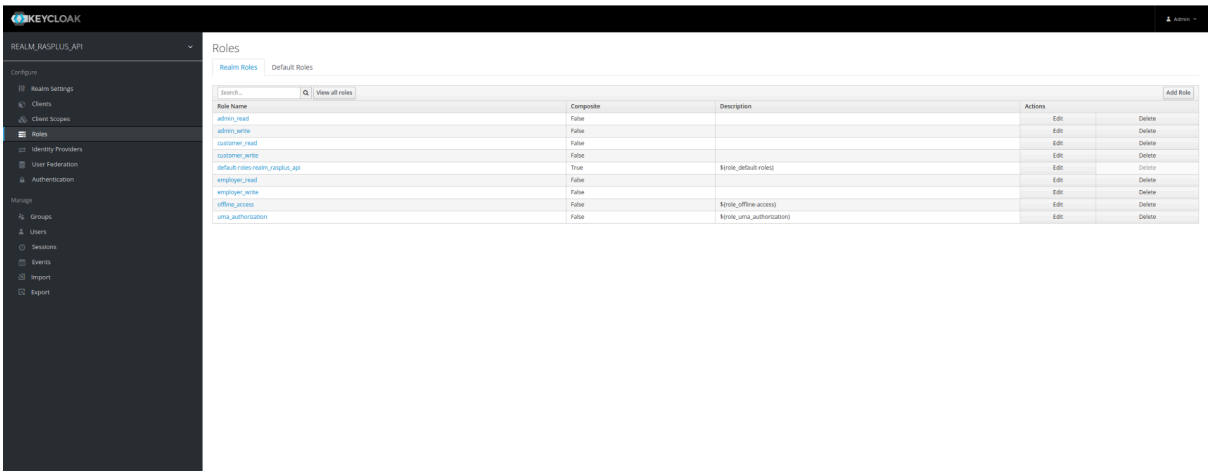




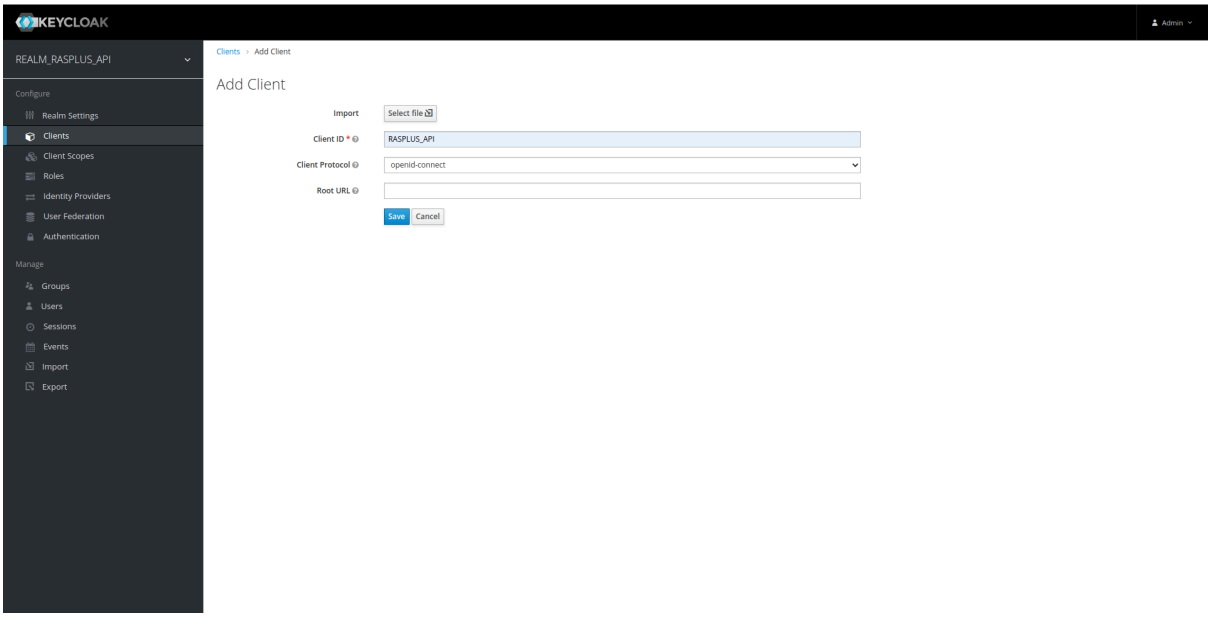
2 - Crie os grupos “admins”, “employers” e “customers”. Vá até groups>new group> nome do grupo. Faça isso 3 vezes.



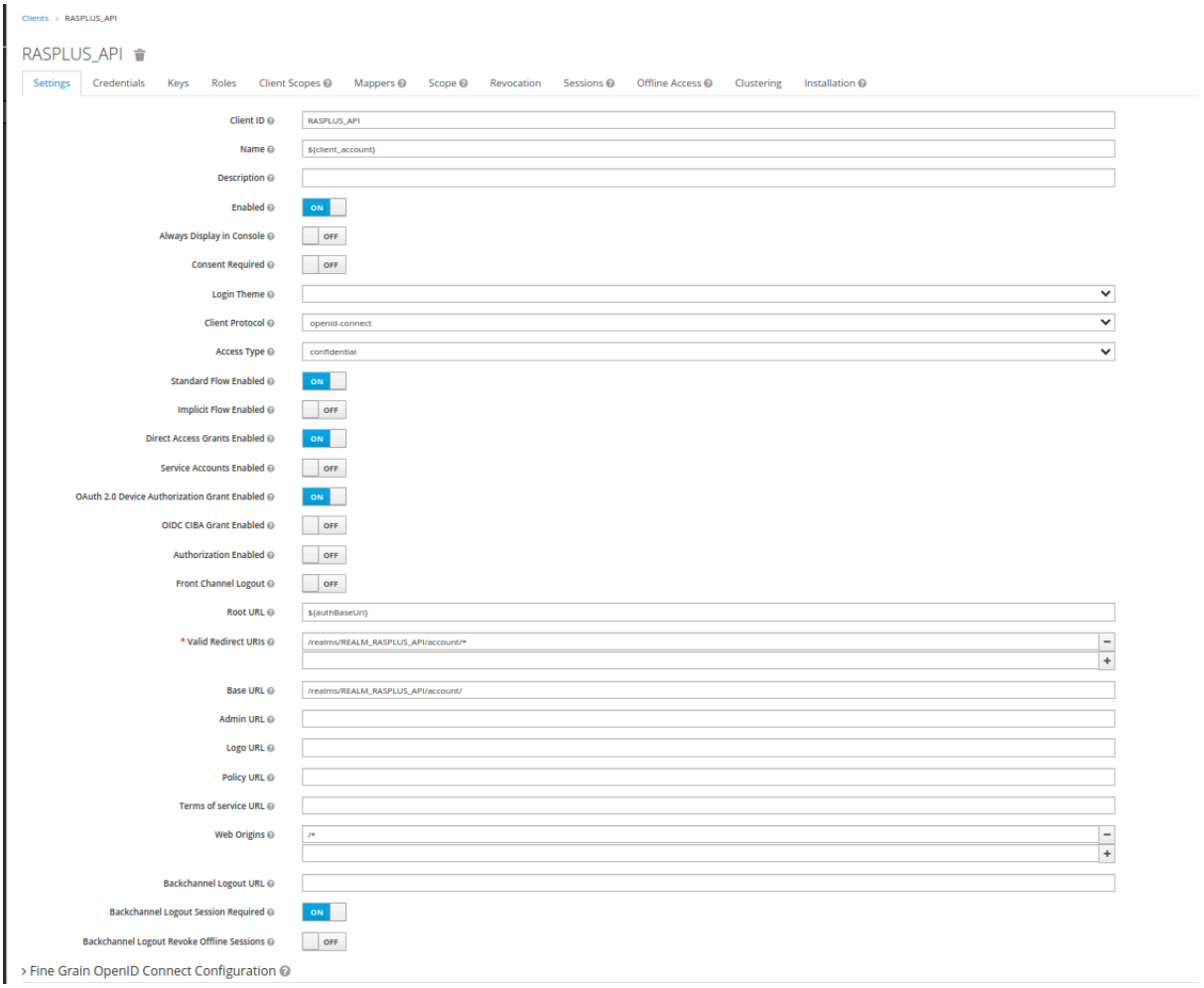
3 - Cadastre as roles admin_write, admin_read, customer_write, customer_read, employer_write, employer_read. Roles > add Role



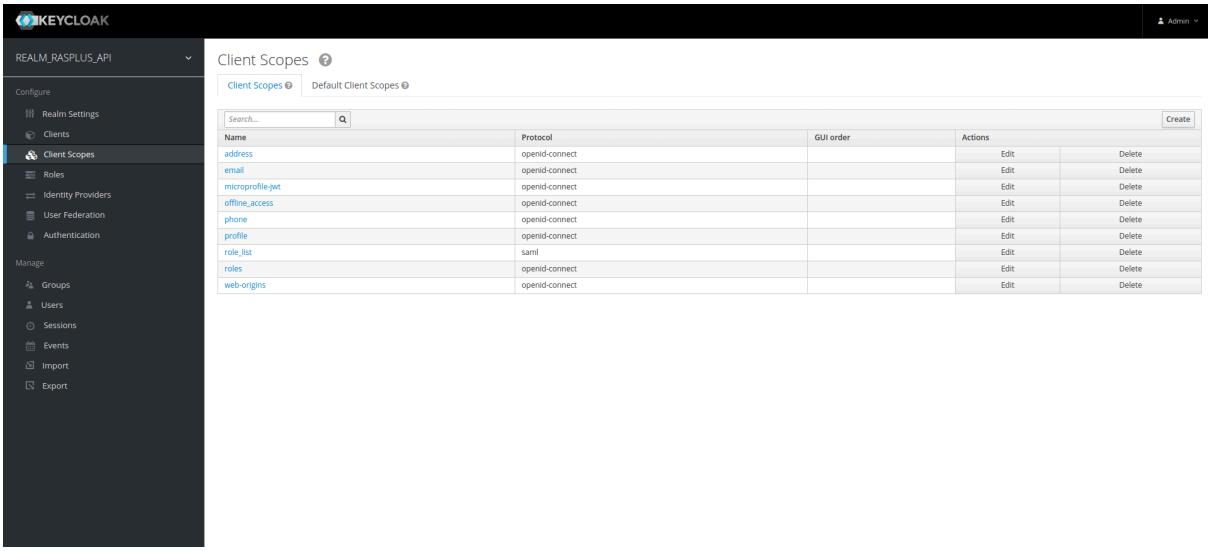
4 - Crie o Client RASPLUS_API



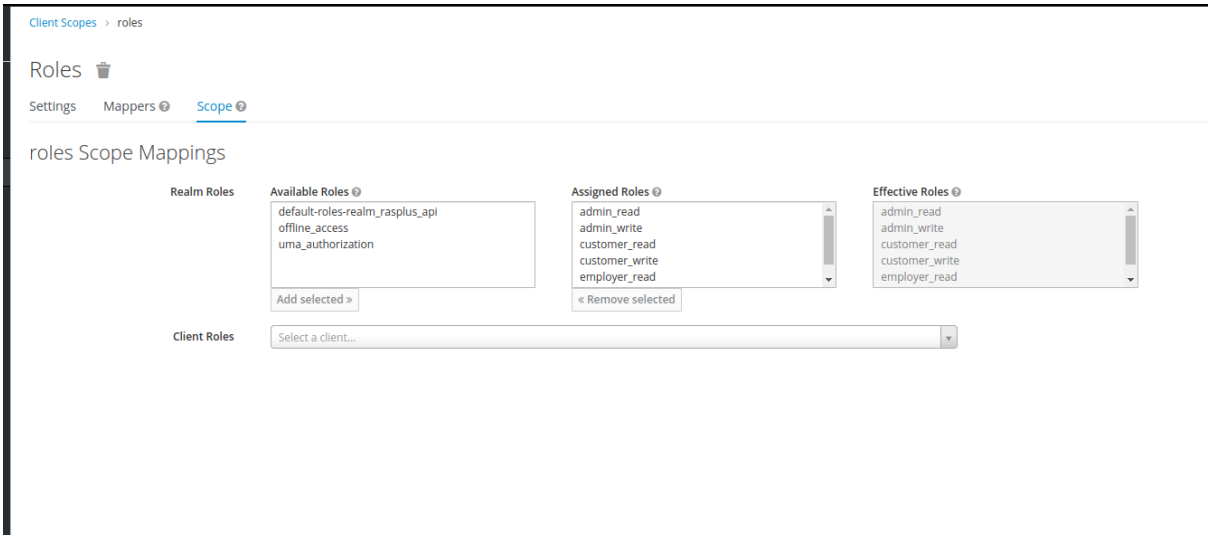
Configure o Client



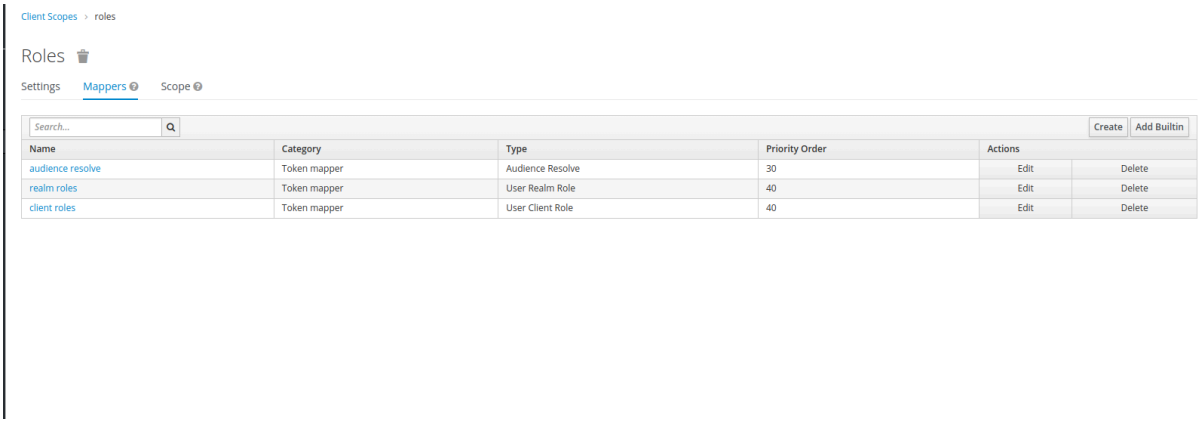
5 - Habilite o client scopes em “client scopes > roles”



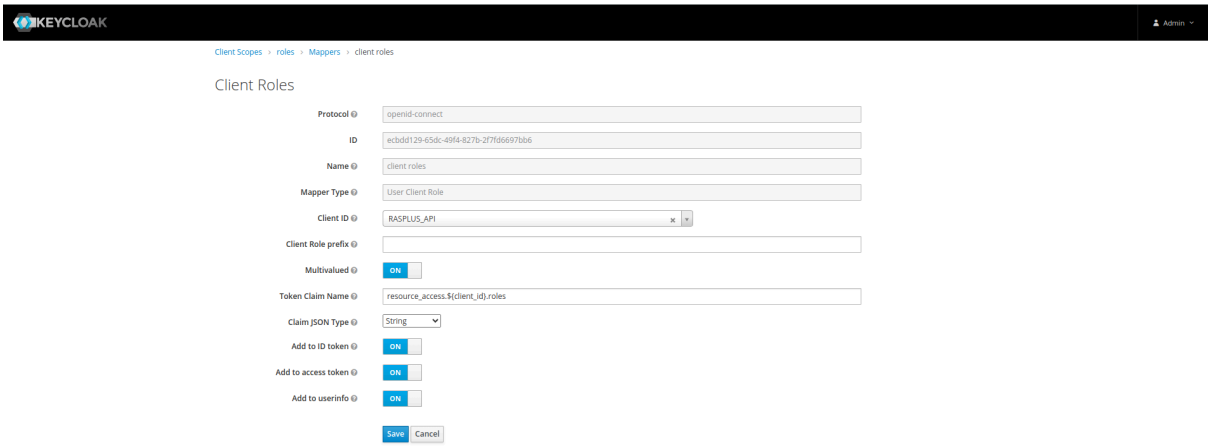
Adicione os scopes criados



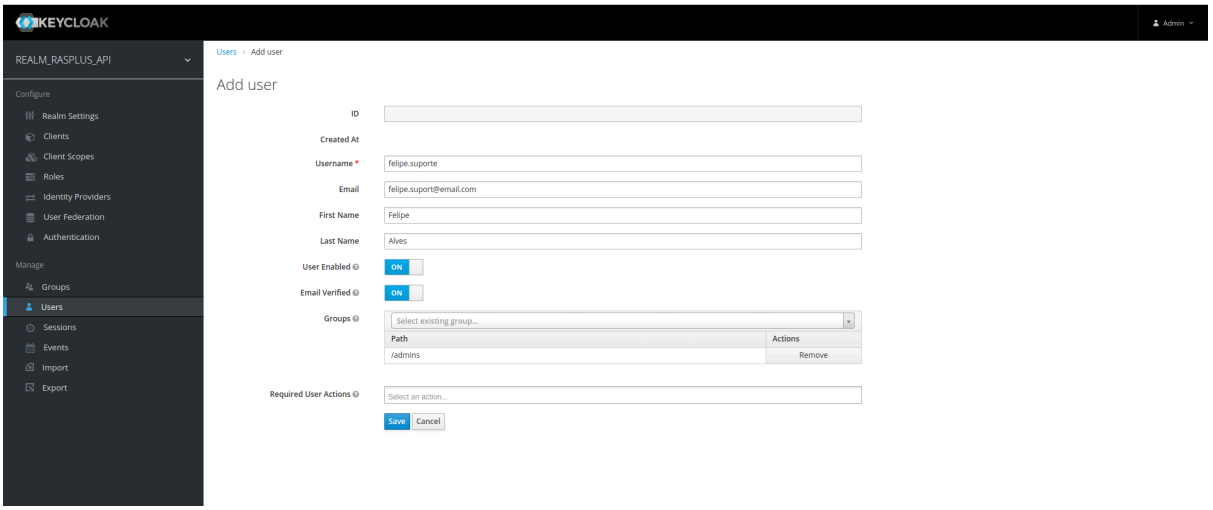
Vá em “mappers > clients roles”



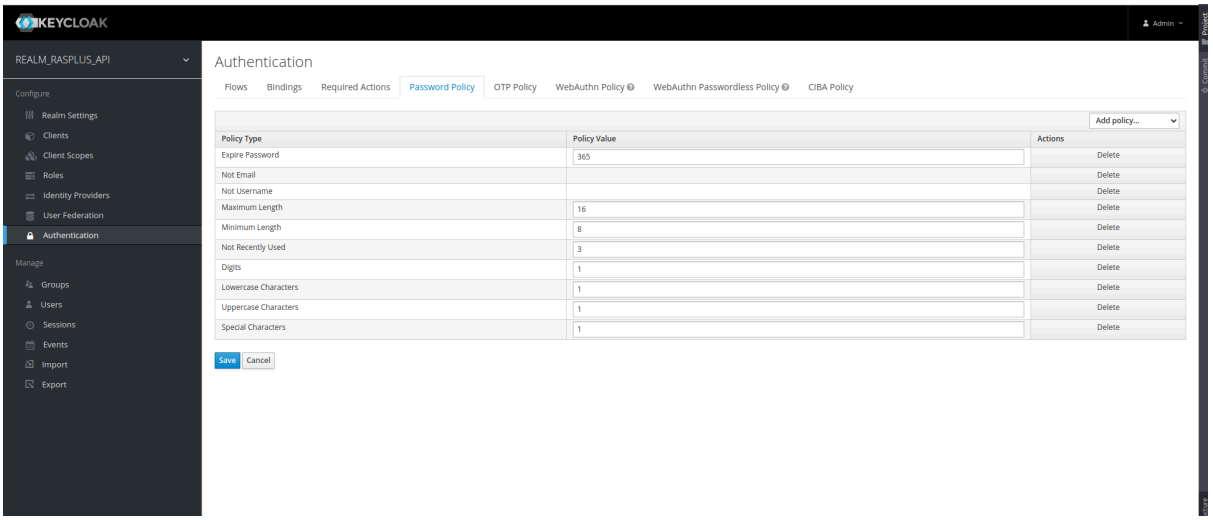
Realize as configurações que serão enviadas.



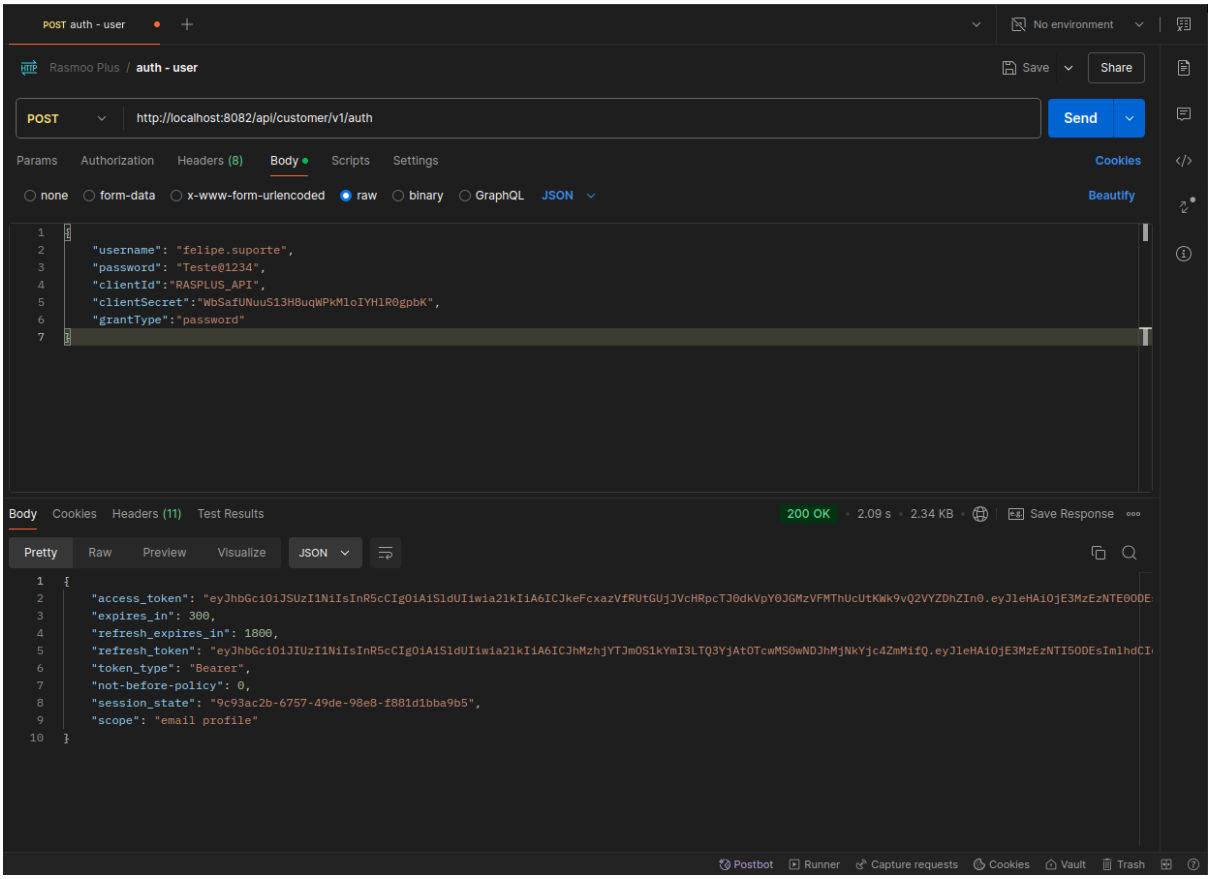
6 - Crie um usuário administrativo em “Users > add user”



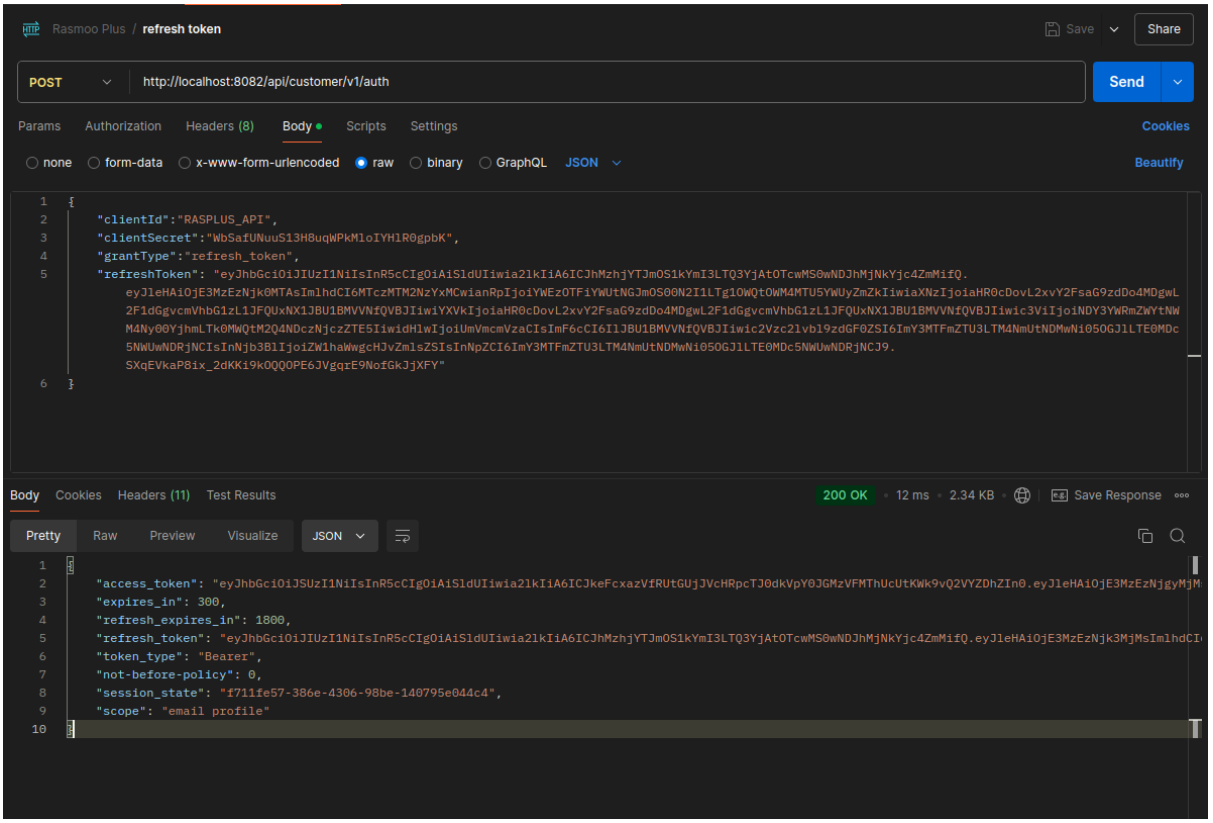
7 - Adicione as políticas de senha em “Authentication > Password Policy”



8 - Teste a requisição de login

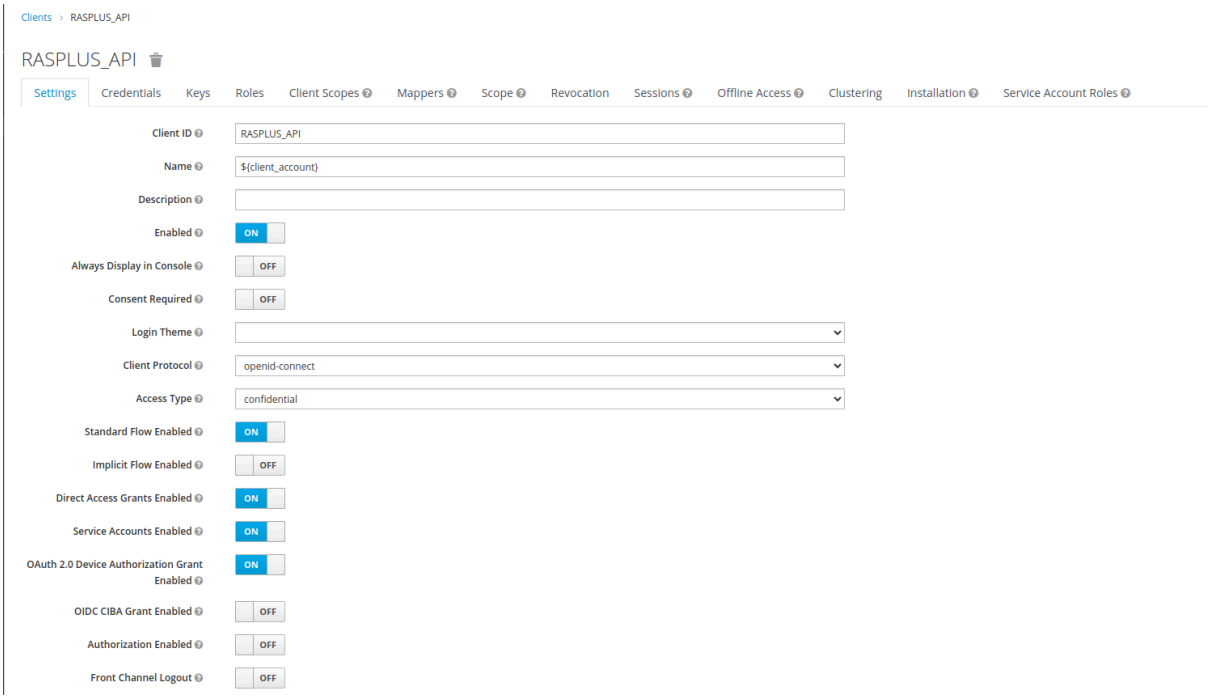


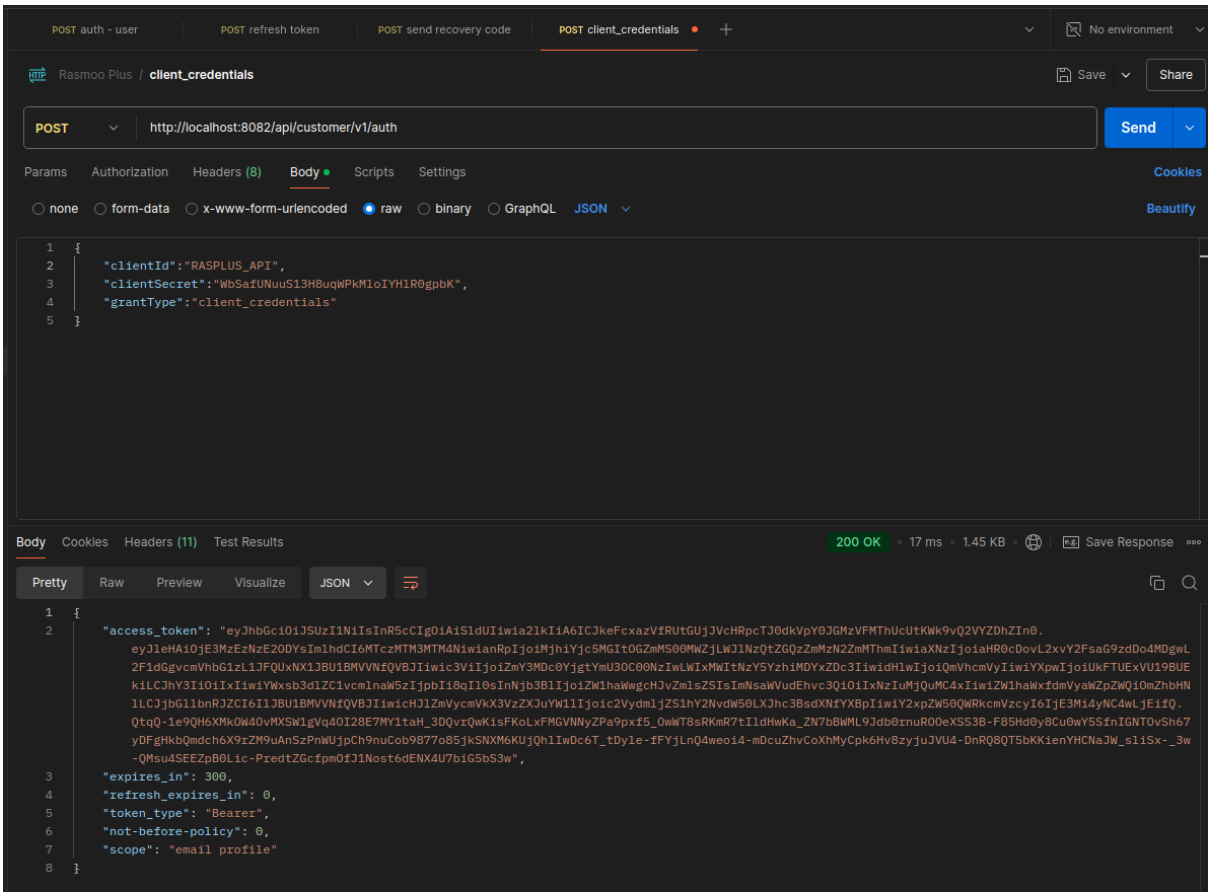
9 - Teste do refresh_token



10 - Teste o fluxo de client_credentials

Habilite o “Service account Enabled” em “clients > RASPLUS_API”

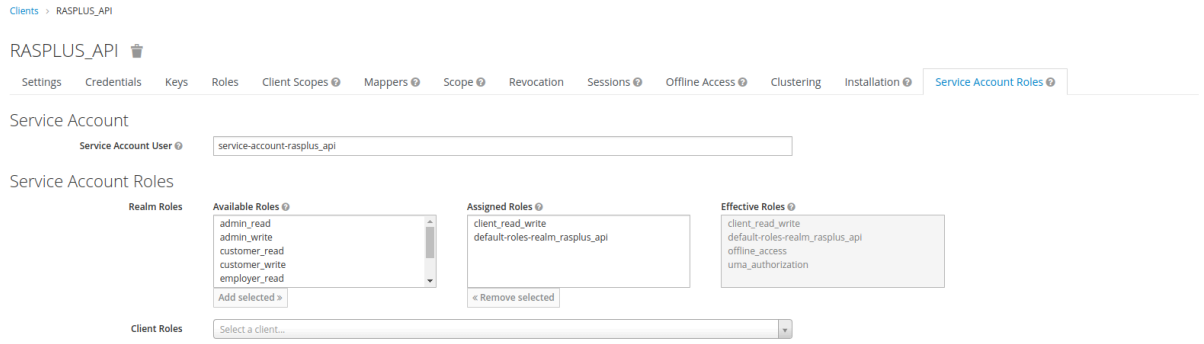




Crie a role client_read_write

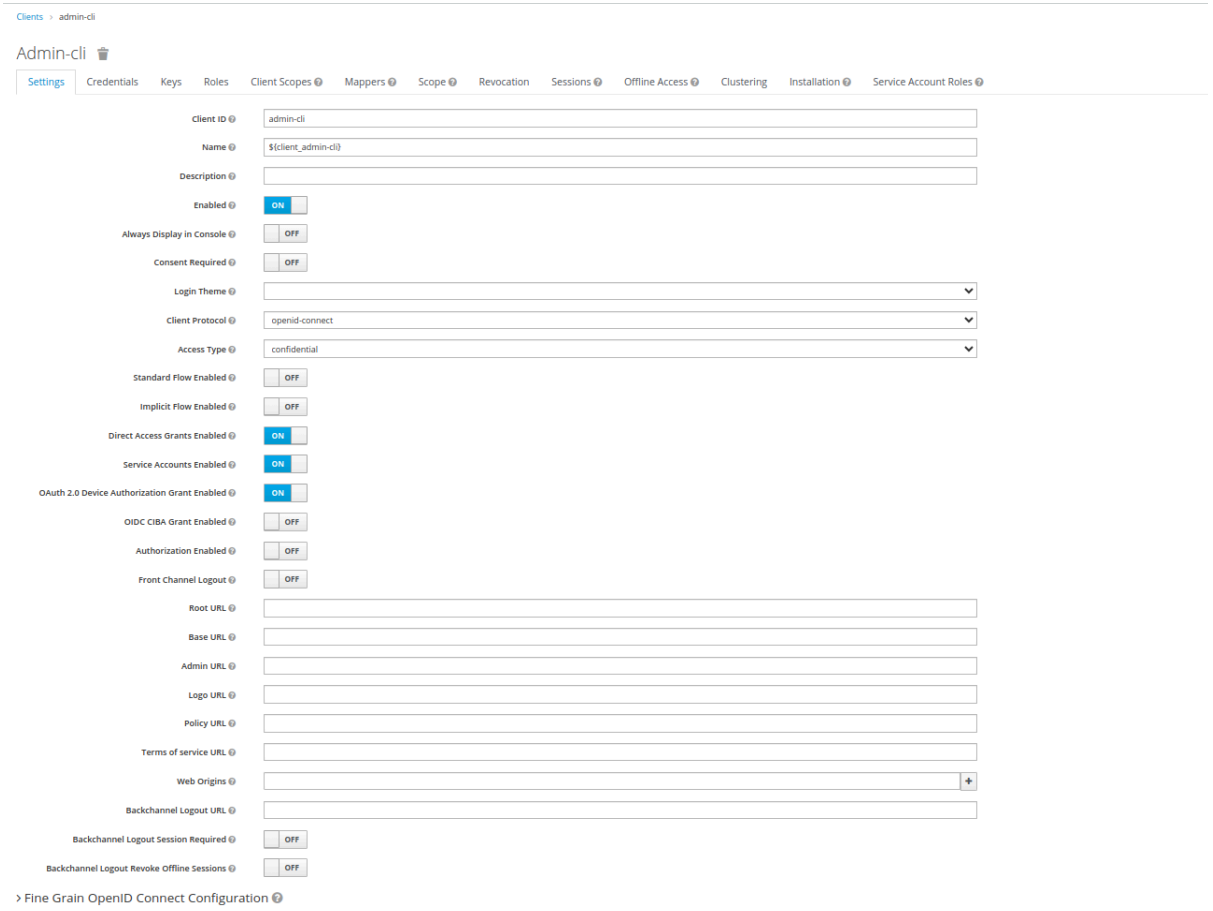
*Não esqueça de habilitar o novo scope ao client scope.

Adicione o client_read_write ao service account

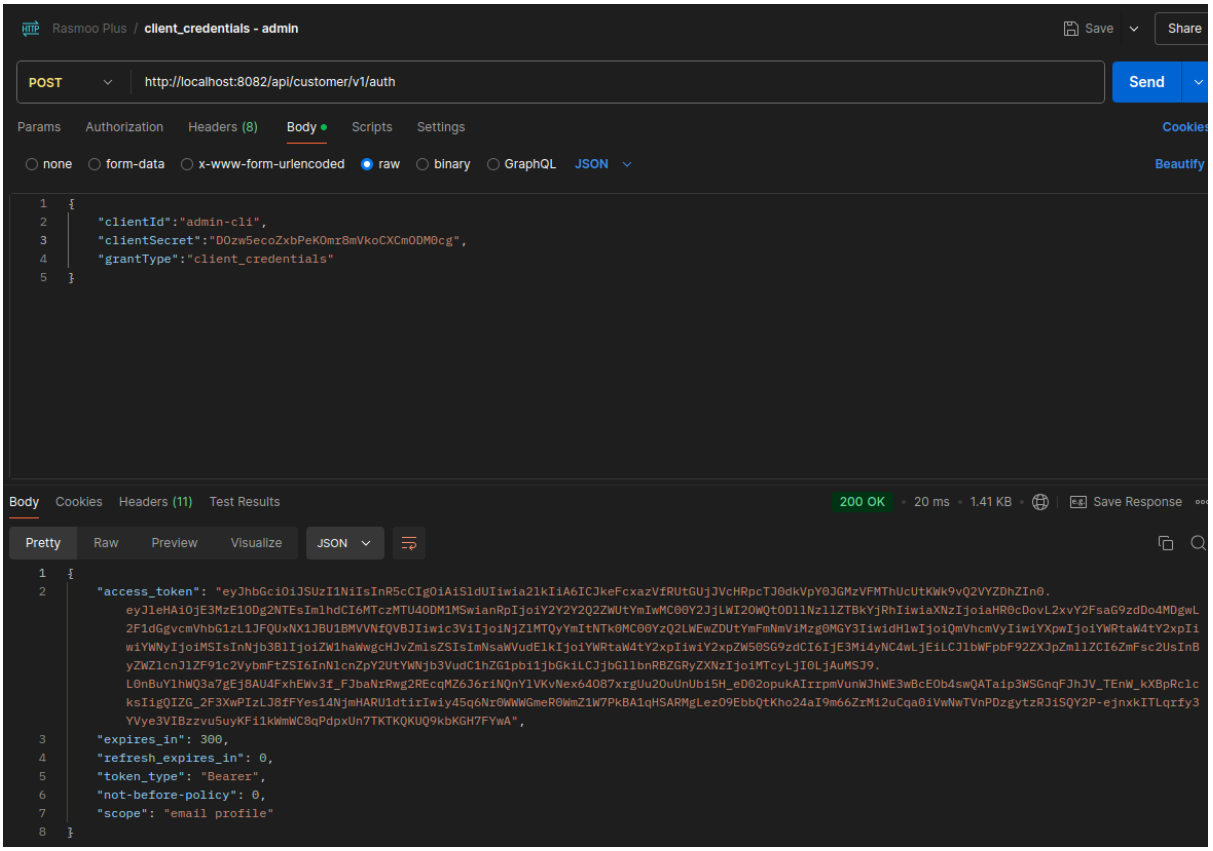


11 - Habilite o admin-cli para criar usuários

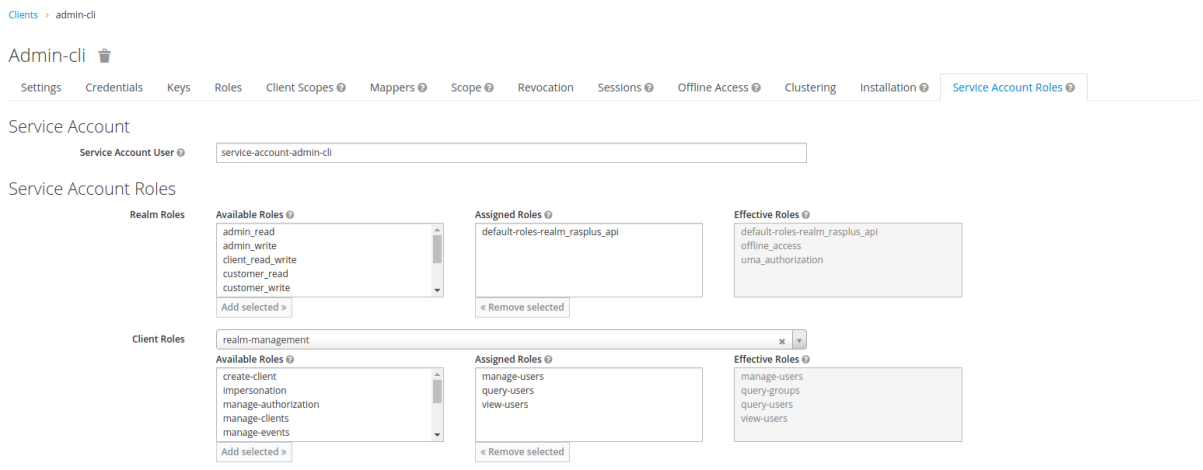
Em “clients>admin-cli”



Copie o token JWT e tente a autenticação



Em “Service Account Roles“ habilite o admin-cli para que ele possa gerenciar e visualizar usuários.



Configuração do Swagger

Crie uma classe “configuration > SwaggerConfig” e adicione o seguinte código para habitar a autenticação por Bearer Authorization

```java
package com.client.api.cutomer.configuration;

import io.swagger.v3.oas.annotations.OpenAPIDefinition;
import io.swagger.v3.oas.annotations.enums.SecuritySchemeType;
import io.swagger.v3.oas.annotations.info.Contact;
import io.swagger.v3.oas.annotations.info.Info;
import io.swagger.v3.oas.annotations.info.License;
import io.swagger.v3.oas.annotations.security.SecurityRequirement;
import io.swagger.v3.oas.annotations.security.SecurityScheme;
import org.springframework.context.annotation.Configuration;

@Configuration
@OpenAPIDefinition(
        info =@Info(
                title = "User API",
                version = "${api.version}",
                contact = @Contact(
                        name = "Rasmoo", email = "midias@rasmoo.com", url =
"https://www.rasmoo.com"
                ),
                license = @License(
                        name = "Apache 2.0", url =
"https://www.apache.org/licenses/LICENSE-2.0"
                ),
                description = "API para para atender ao frontend do RASMOO
PLUS"
        ),
        security = {@SecurityRequirement(name = "Bearer Authentication")
}
)
@SecurityScheme(
        name = "Bearer Authentication",
        type = SecuritySchemeType.HTTP,
        bearerFormat = "JWT",
        scheme = "bearer"
)
public class SwaggerConfig {
}
```