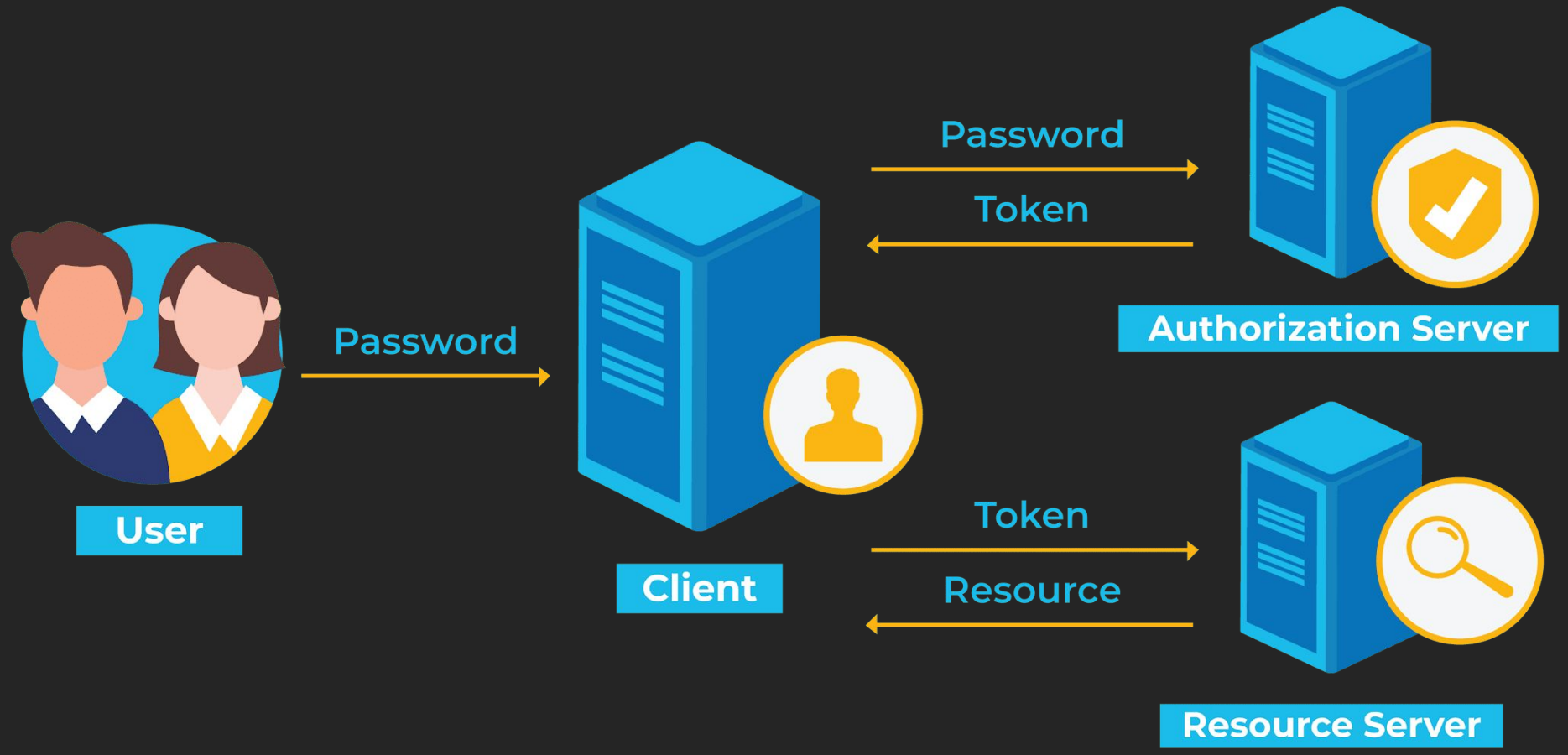
Autentikacija i autorizacija

Spring security 6.0 + keycloak demo

Autentikacija



okta

Json Web Token

- JSON objekat za bezbedan prenos informacija
- Podaci su bezbedni zato što su digitalno potpisani
- Nakon što se korisnik uspešno prijavi na sistem, generiše mu se token koji će se koristiti u svakom narednom zahtevu tog korisnika
- Tri Base64-URL enkodovana stringa, razdvojena tačkom
- Validacija tokena https://www.jwt.io/

Json Web Token

- Zaglavlje tokena definiše algoritme, tip tokena i identifikator ključa koji se koristio za digitalni potpis

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "3X_z3DnWtZzwVf3RY5auXQTI_VrGgcd3Mi-g7JFrVfE"
}
```

- RSA i SHA-256 su korišteni za enkripciju zaglavlja i tela tokena
- Rezultat ove enkripcije se nalazi u poslednjem delu tokena

Oauth 2.0

- Standard koji definiše na koji način servisi ili korisnici mogu da budu autorizovani kako bi pristupili resursima drugih servisa, bez da podele svoje kredencijale sa tim servisima
- Umesto kredencijala, za autorizaciju se koriste access tokeni (obično JWT)
- OAuth podržava RBAC (role based access control) – dodela prava pristupa korisnicima na osnovu njihove uloge u sistemu

Keycloak

- Open-source IAM (identity and access management) alat
- Podržava *SingleSignOn*, RBAC model autorizacije, autentikaciju putem naloga sa društvenih mreža, itd.
- Realm – security kontekst za upavljanje korisnicima, grupama, rolama, itd.
  - Na nivou *realm*-a se prave podešavanja za korisnike, role, prijavu na sistem, način autorizacije itd.
- Prepuštamo logiku registracije, prijave na sistem i autorizacije *keycloak*-u, a korisnicima i rolama upravljamo pomoću *keycloak* admin konzole

Instalacija i pokretanje

Keycloak možete preuzeti sa https://www.keycloak.org/downloads

Pokretanje pomoću docker-a:

```
docker run -p 127.0.0.1:8080:8080 -e
  KC_BOOTSTRAP_ADMIN_USERNAME=admin -e
  KC_BOOTSTRAP_ADMIN_PASSWORD=admin
  quay.io/keycloak/keycloak:26.3.3 start-dev
```

Spring boot konfiguracija

- Potrebne zavisnosti za `pom.xml`:
  Spring Security
  OAuth2 Resource Server
- `application.properties`:
  `spring.security.oauth2.resourceserver.jwt.issuer-uri`
  - bazni url koji identifikuje server za autorizaciju (u ovom slučaju *keycloak*)
  - bazni url je sadržan u `iss` claim-u u tokenu

Spring boot konfiguracija

```
http.oauth2ResourceServer(oauth -> oauth.jwt(jwt ->
jwt.jwtAuthenticationConverter(jwtRoleConverter)));
```

- konfigurišemo našu aplikaciju da prihvata i validira JWT
  - čita token iz Authorization: Bearer <token>
  - validira pomoću `issuer-uri` parametra
- Kako ćemo pročitati role iz tokena i konvertovati ih u *GrantedAuthority* koji odgovara *SpringSecurity* formatu?
  - pravimo *custom* converter

Dodatni resursi

https://aaronparecki.com/oauth-2-simplified/

https://docs.spring.io/spring-security/reference/servlet/oauth2/resource-server/jwt.html

https://www.keycloak.org/securing-apps/overview

https://www.keycloak.org/securing-apps/javascript-adapter