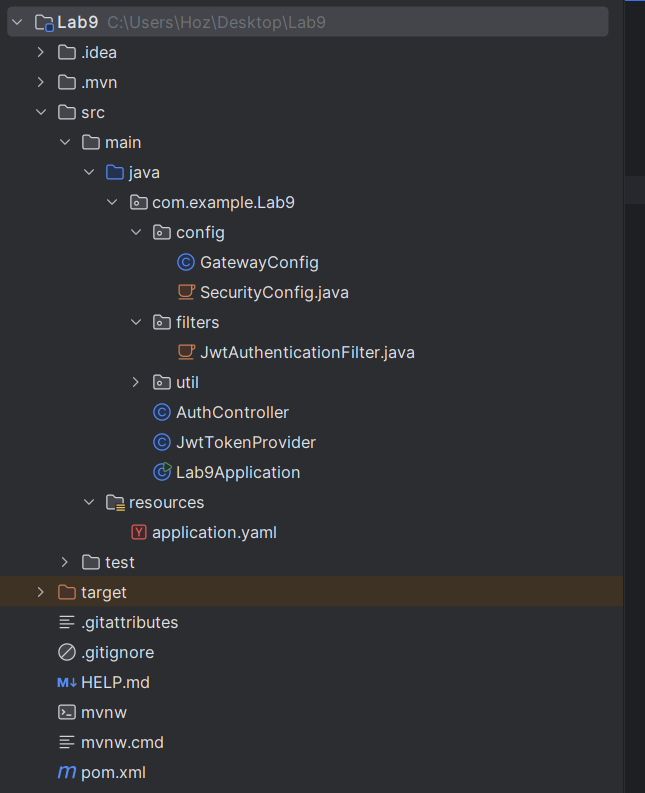
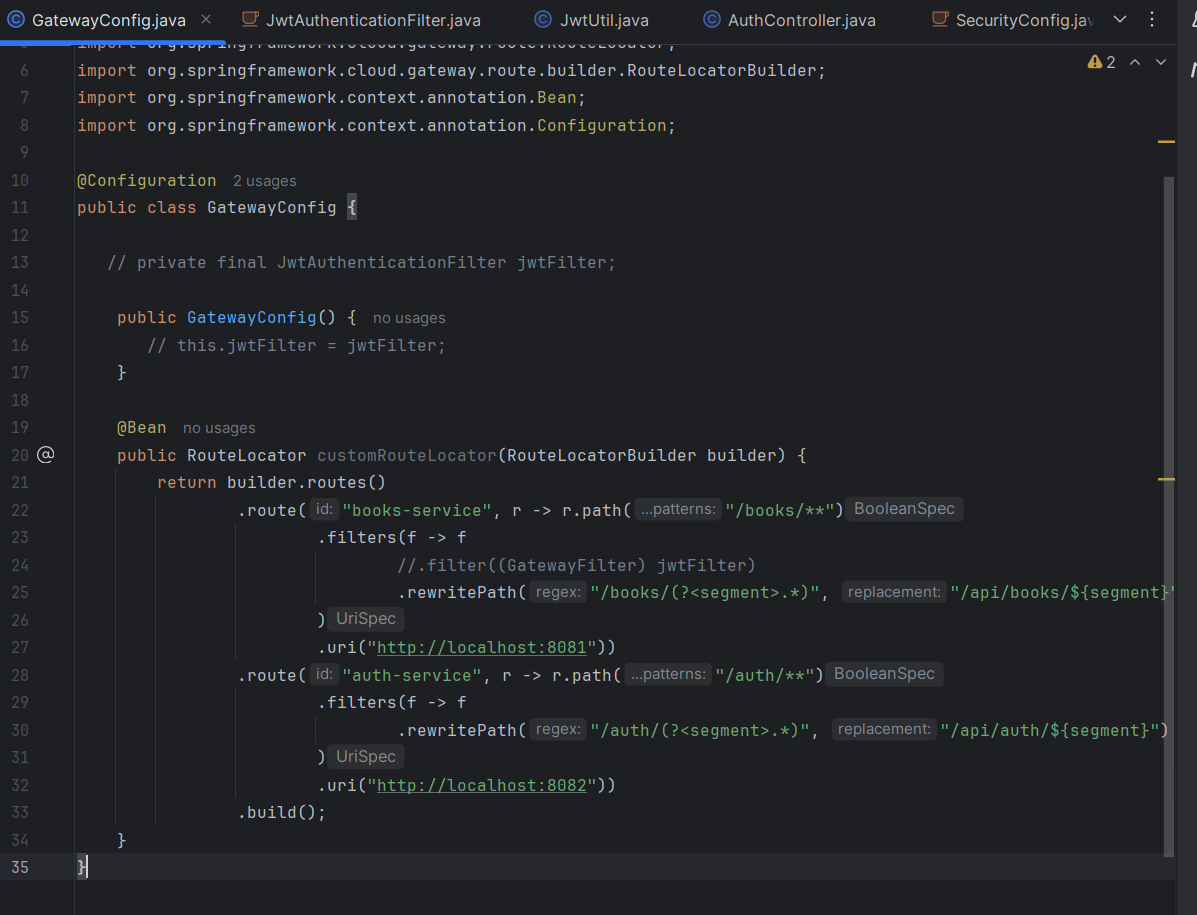


1 Сделали gateway сервис , который перенаправляет все запросы в нужные сервисы:

Структура проекта:

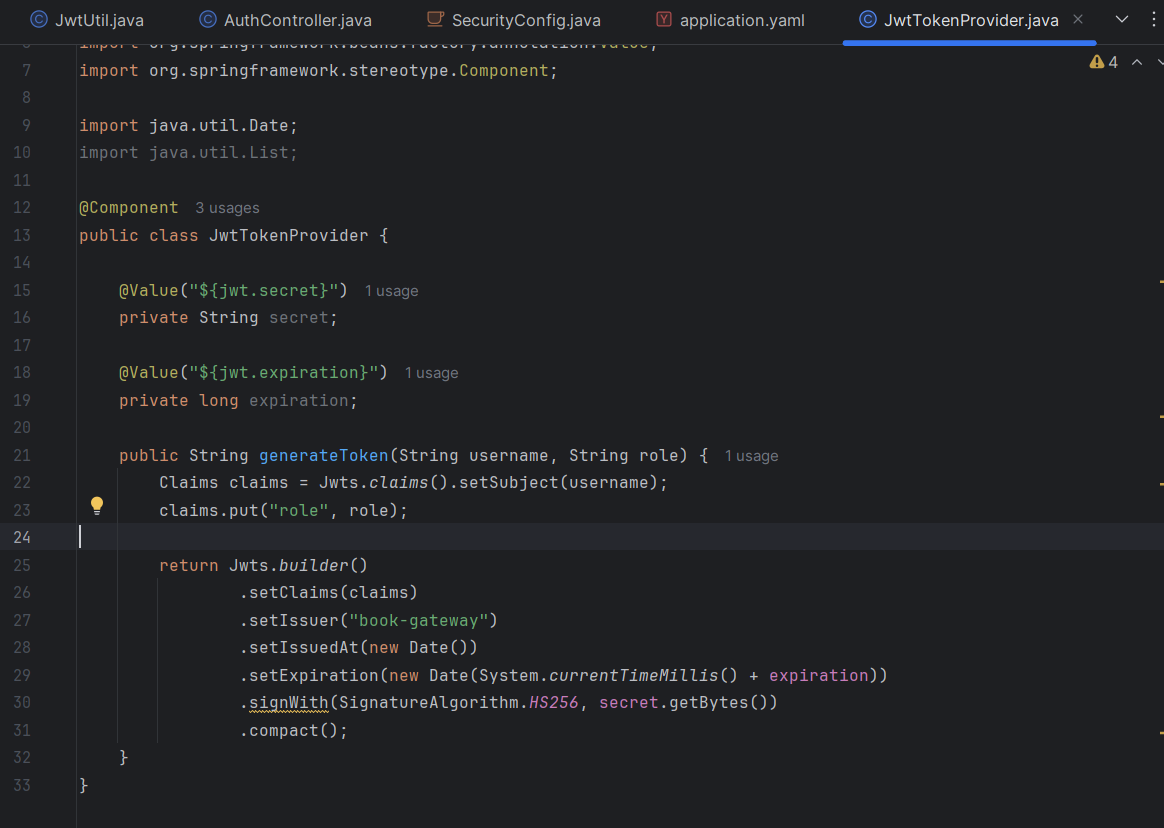


Gateway config – перенаправляет запросы.

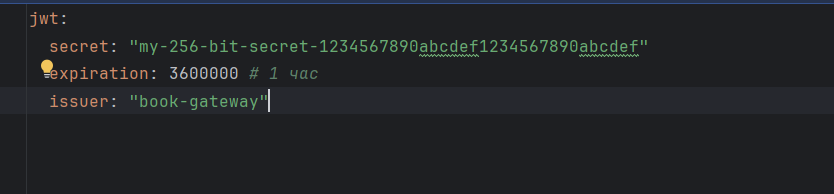


Контроллер авторизации:

2. Для аутентификации используется JWT provider.



Jwt данные в application.yaml



2.1 Реализовали jwt фильтр для аутентификации в основном сервисе:

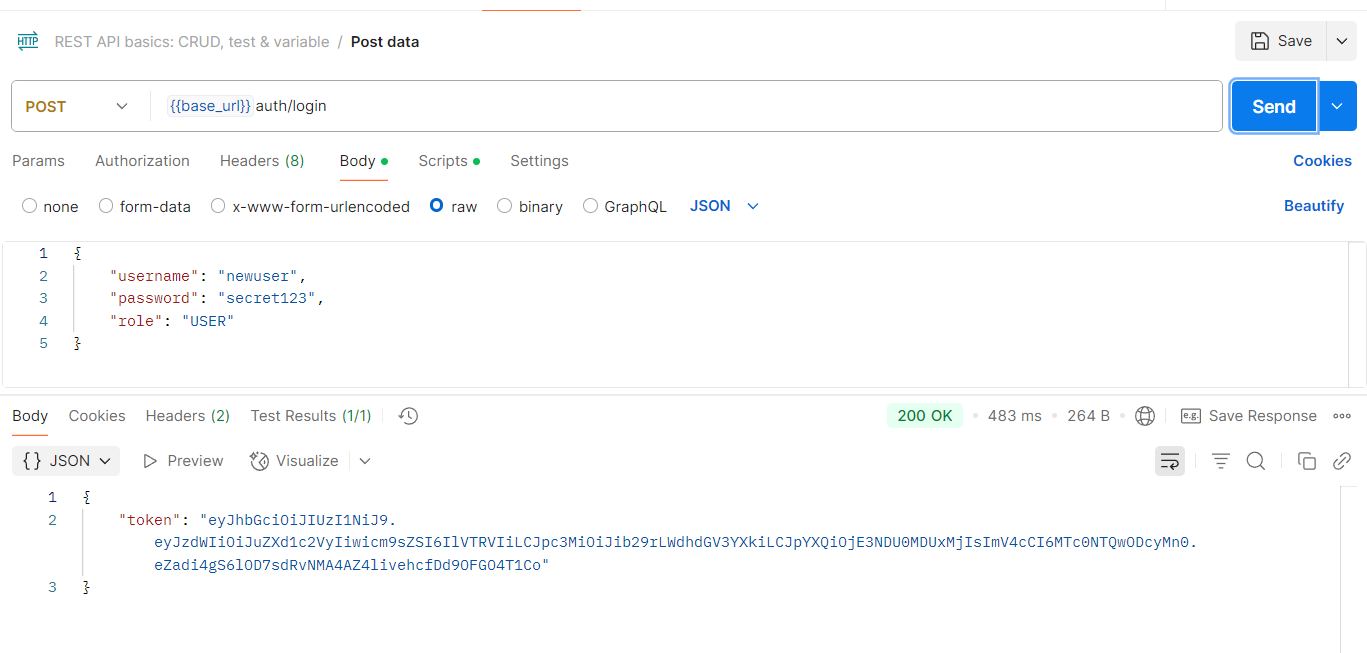
package com.example.demo.config;  
  
import io.jsonwebtoken.Claims;  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.security.Keys;  
import jakarta.servlet.FilterChain;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;  
import org.springframework.security.core.Authentication;  
import org.springframework.security.core.authority.SimpleGrantedAuthority;  
import org.springframework.security.core.context.SecurityContextHolder;  
import org.springframework.stereotype.Component;  
import org.springframework.web.filter.OncePerRequestFilter;  
import java.io.IOException;  
import java.nio.charset.StandardCharsets;  
import java.util.Collections;  
import java.util.List;  
  
@Component  
public class JwtAuthFilter extends OncePerRequestFilter {  
  
 @Value("${jwt.secret}")  
 private String secret;  
  
 @Override  
 protected void doFilterInternal(HttpServletRequest request,  
 HttpServletResponse response,  
 FilterChain filterChain)  
 throws ServletException, IOException {  
  
 try {  
 String token = resolveToken(request);  
 if (token != null && validateToken(token)) {  
 Authentication auth = getAuthentication(token);  
 SecurityContextHolder.*getContext*().setAuthentication(auth);  
 }  
 } catch (Exception e) {  
 logger.error("JWT validation error", e);  
 response.sendError(HttpServletResponse.*SC\_UNAUTHORIZED*, "Invalid JWT");  
 return;  
 }  
  
 filterChain.doFilter(request, response);  
 }  
  
 private String resolveToken(HttpServletRequest request) {  
 String bearerToken = request.getHeader("Authorization");  
 if (bearerToken != null && bearerToken.startsWith("Bearer ")) {  
 return bearerToken.substring(7);  
 }  
 return null;  
 }  
  
 private boolean validateToken(String token) {  
 try {  
 Jwts.*parser*()  
 .setSigningKey(Keys.*hmacShaKeyFor*(secret.getBytes(StandardCharsets.*UTF\_8*)))  
 .build()  
 .parseClaimsJws(token);  
 return true;  
 } catch (Exception e) {  
 logger.error("JWT validation failed", e);  
 return false;  
 }  
 }  
  
 private Authentication getAuthentication(String token) {  
 Claims claims = Jwts.*parser*()  
 .setSigningKey(Keys.*hmacShaKeyFor*(secret.getBytes(StandardCharsets.*UTF\_8*)))  
 .build()  
 .parseClaimsJws(token)  
 .getBody();  
  
 String username = claims.getSubject();  
 String role = claims.get("role", String.class);  
  
 // Проверяем, что роль не null и не пустая  
 if (role == null || role.trim().isEmpty()) {  
 throw new IllegalArgumentException("Role cannot be empty");  
 }  
  
 // Добавляем префикс ROLE\_, если его нет (это стандартное требование Spring Security для ролей)  
 if (!role.startsWith("ROLE\_")) {  
 role = "ROLE\_" + role;  
 }  
  
 List<SimpleGrantedAuthority> authorities = Collections.*singletonList*(  
 new SimpleGrantedAuthority(role)  
 );  
  
 return new UsernamePasswordAuthenticationToken(  
 username,  
 null,  
 authorities  
 );  
 }  
}

3. Добавили фильтр в SecurityConfig в основном сервисе :

package com.example.demo.config;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;  
import org.springframework.security.web.SecurityFilterChain;  
import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;  
  
@Configuration  
@EnableWebSecurity  
public class SecurityConfig {  
  
 private final JwtAuthFilter jwtAuthFilter;  
  
 public SecurityConfig(JwtAuthFilter jwtAuthFilter) {  
 this.jwtAuthFilter = jwtAuthFilter;  
 }  
  
 @Bean  
 public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
 return http  
 .csrf(csrf -> csrf.disable())  
 .addFilterBefore(jwtAuthFilter, UsernamePasswordAuthenticationFilter.class)  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/api/public/\*\*").permitAll()  
 .anyRequest().authenticated()  
 )  
 .build();  
 }  
}

4. Протестировали работу gateway при помощи Postman

Получение токена



Перенаправление на другой сервис через gateway с запросом Authorization заголовка с Bearer токеном.

