22. 리팩토링 - signIn

1. signIn 서비스

- 이것저것 로직이 너무 꼬여있어서 보기 힘들다.
- 리펙토링해서 뭐가 뭔지 한눈에 파악하게끔 바꾸자.

```
async signIn(username: string, pass: string): Promise<{ access_token: string,
refresh_token: string }> {
    const user = await this.usersService.findByUsername(username);
    if (!user) {
      throw new UnauthorizedException();
    }

    if (user.loginAttempts >= 5) {
      throw new UnauthorizedException('최대 로그인 시도 횟수를 초과하였습니다.');
    }

    if (!(await this.comparePasswords(pass, user.password))) {
      user.loginAttempts += 1;
      await this.userRepository.save(user);
      throw new UnauthorizedException();
    }


    // 로그인 성공 시, 로그인 시도 횟수 초기화
    user.loginAttempts = 0;
    await this.userRepository.save(user);


    const payload = { username: user.username, sub: user.id, role: user.role };
    const refresh_token = await this.jwtService.signAsync(payload, {
      expiresIn: '7d'  // Refresh 토큰 유효기간 설정
    });
    const access_token = await this.jwtService.signAsync(payload, {
      expiresIn: '60s'
    });
```

```
    // refresh token, access token 업데이트
    user.refreshToken = refresh_token;
    user.accessToken = access_token;
    await this.userRepository.save(user);

    return {
      access_token,
      refresh_token,
    };
  }
```

2. 리팩토링 코드

- 한 눈에 알아보기 쉽게 바뀌었다.

```
async signIn(username: string, pass: string): Promise<{ access_token: string,
refresh_token: string }> {
    const user = await this.usersService.findByUsername(username);
    if (!user) {
      throw new UnauthorizedException('사용자를 찾을 수 없습니다.');
    }

    if (user.loginAttempts >= 5) {
      throw new UnauthorizedException('최대 로그인 시도 횟수를 초과하였습니다.');
    }

    if (!(await this.comparePasswords(pass, user.password))) {
      await this.incrementLoginAttempts(user);
      throw new UnauthorizedException('잘못된 비밀번호입니다.');
    }

    await this.resetLoginAttempts(user);
    const { access_token, refresh_token } = await this.createTokens(user);

    return { access_token, refresh_token };
  }
```