# 11. 요구사항-1(2)

## 1. 기본예제 상황

- `auth.module.ts`

```
import { Module } from '@nestjs/common';
import { APP_GUARD } from '@nestjs/core';
import { JwtModule } from '@nestjs/jwt';
import { UsersModule } from '../users/users.module';
import { AuthController } from './auth.controller';
import { AuthGuard } from './auth.guard';
import { AuthService } from './auth.service';
import { jwtConstants } from './constants';

@Module({
  imports: [
    UsersModule,
    JwtModule.register({
      global: true,
      secret: jwtConstants.secret,
      signOptions: { expiresIn: '600s' },
    }),
  ],
  providers: [
    AuthService,
    {
      provide: APP_GUARD,
      useClass: AuthGuard,
    },
  ],
  controllers: [AuthController],
  exports: [AuthService],
})
export class AuthModule {}
```

- `users.module.ts`

```
import { Module } from '@nestjs/common';
import { UsersService } from './users.service';
import { AuthModule } from 'src/auth/auth.module';
import { TypeOrmModule } from '@nestjs/typeorm';
import { User } from 'src/entities/user.entity';
import { PassportModule } from '@nestjs/passport';
import { JwtModule } from '@nestjs/jwt';
import { UsersController } from './users.controller';

@Module({
  imports: [
    AuthModule,
    TypeOrmModule.forFeature([User]),
    PassportModule,
    JwtModule.register({
      secret: '1234',
      signOptions: { expiresIn: '600s' },
    }),
  ],
  providers: [UsersService],
  exports: [UsersService],
  controllers: [UsersController]
})
export class UsersModule {}
```

- `app.module.ts`

```
// app.module.ts
import { Module } from '@nestjs/common';
import { TypeOrmModule } from '@nestjs/typeorm';
import { AppController } from './app.controller';
import { AppService } from './app.service';
import { User } from './entities/user.entity'; // User 엔티티 임포트
import { AuthService } from './auth.service';
import { JwtModule } from '@nestjs/jwt';
import { PassportModule } from '@nestjs/passport';
import { JwtStrategy } from './jwt.strategy';
import { UserService } from './user.service';
import { AuthModule } from './auth/auth.module';


@Module({
  imports: [
    AuthModule,
    TypeOrmModule.forFeature([User]),
    TypeOrmModule.forRoot({
      type: 'mysql',
      host: 'localhost',
      port: 3306,
      username: 'root',
      password: '1234',
      database: 'opstest',
      entities: [User], // User 엔티티 등록
      synchronize: true,
    }),
    PassportModule,
    JwtModule.register({
      secret: '1234',
      signOptions: { expiresIn: '600s' },
    }),
  ],
  controllers: [AppController],
  providers: [AppService, AuthService, JwtStrategy, UserService],
})
export class AppModule {}
```

- 오류

```
[Nest] 10888  - 2024. 01. 25. 오후 2:03:44   ERROR [ExceptionHandler] Nest cannot
create the UsersModule instance.
The module at index [0] of the UsersModule "imports" array is undefined.

Potential causes:
- A circular dependency between modules. Use forwardRef() to avoid it. Read more:
https://docs.nestjs.com/fundamentals/circular-dependency
- The module at index [0] is of type "undefined". Check your import statements and
the type of the module.
```

```
Scope [AppModule -> AuthModule]
Error: Nest cannot create the UsersModule instance.
The module at index [0] of the UsersModule "imports" array is undefined.

Potential causes:
- A circular dependency between modules. Use forwardRef() to avoid it. Read more:
https://docs.nestjs.com/fundamentals/circular-dependency
- The module at index [0] is of type "undefined". Check your import statements and
the type of the module.

Scope [AppModule -> AuthModule]
    at DependenciesScanner.scanForModules (C:\gits\ops-
test\node_modules\@nestjs\core\scanner.js:60:23)
    at DependenciesScanner.scanForModules (C:\gits\ops-
test\node_modules\@nestjs\core\scanner.js:68:32)
    at DependenciesScanner.scanForModules (C:\gits\ops-
test\node_modules\@nestjs\core\scanner.js:68:32)
    at DependenciesScanner.scan (C:\gits\ops-
test\node_modules\@nestjs\core\scanner.js:27:9)
    at C:\gits\ops-test\node_modules\@nestjs\core\nest-factory.js:107:17
    at Function.asyncRun (C:\gits\ops-
test\node_modules\@nestjs\core\errors\exceptions-zone.js:22:13)
    at NestFactoryStatic.initialize (C:\gits\ops-
test\node_modules\@nestjs\core\nest-factory.js:106:13)
    at NestFactoryStatic.create (C:\gits\ops-test\node_modules\@nestjs\core\nest-
factory.js:42:9)
    at bootstrap (C:\gits\ops-test\src\main.ts:5:15)
PS C:\gits\ops-test>
```

## 2. register 동작 확인

```
import {
    Body,
    Controller,
    Get,
    HttpCode,
    HttpStatus,
    Post,
    Request,
  } from '@nestjs/common';
```

```
import { UsersService } from './users.service';
import { RegisterDto } from 'src/dto/register.dto';
import { AuthService } from 'src/auth/auth.service';
import { Public } from 'src/auth/decorators/public.decorator';

@Controller('users')
export class UsersController {
constructor(private usersService: UsersService,
            private authService: AuthService) {}

    @Public()
    @Post('register')
    async register(@Body() registerDto: RegisterDto) {
        // 이메일 중복 검사
        const existingUser = await
this.usersService.findByEmail(registerDto.email);
        if (existingUser) {
        throw new Error('이미 존재하는 이메일입니다.');
        }

        // 비밀번호 해싱
        const hashedPassword = await
this.authService.hashPassword(registerDto.password);

        // 사용자 정보 저장
        await this.usersService.createUser({
        ...registerDto,
        password: hashedPassword,
        });

        // 응답 반환 (비밀번호 정보는 제외)
        return { email: registerDto.email };
}

}
```

```
mysql> SELECT * FROM opstest.user;
+----+----------------------+------------------------------------------------------------
--------+--------+
| id | email                | password
    | role   |
+----+----------------------+------------------------------------------------------
--------+--------+
|  1 | example@example.com  |
$2b$10$J11c5tJv3UW0ZAPvXC2Xz.1JOSl.7tv4g3WjvNKUFeoQT9hiSVgAW | MEMBER |
|  2 | example2@example.com |
$2b$10$NSVNvtlv7qe2OEf.VScQwu89T0.I6AUlXQZmB47n.2GHU7RxjVzZS | MEMBER |
|  3 | example3@example.com |
$2b$10$XLr3CDeAdqVgW4pdlo8jneL/NzO2BwflDl9jnO2/E1TTZzDJruT9K | MEMBER |
|  4 | test@gmail.com       |
$2b$10$jfvKRScCq2zSvGmTgNqtVOoMb/.5jvzMCnwIXiof3uj/ZEtBBsA2. | MEMBER |
|  5 | test2@gmail.com      |
$2b$10$1ix8ntEP0U1Q/9m7H9Jg/.nziAAKB6w/0F88NyrR/5I13wF4Cp5Oy | MEMBER |
+----+----------------------+------------------------------------------------------
--------+--------+
5 rows in set (0.00 sec)
```

- 다음과 같이 정상적으로 동작하는 것을 확인할 수 있다.