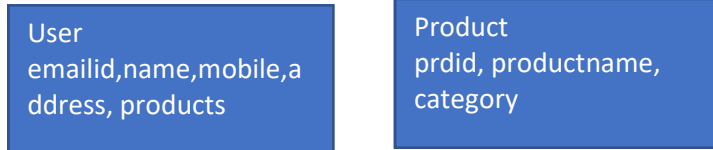


Product application



- add useraccount
- user can add product under his account
- user can view products added under his account

Springboot application [authenticationpp]

controller

service

repo

User

id,pwd,role,emailid
signup for new user (secure data)
logincheck, returns jwt

MySQL DB
User table

Springboot application [userproduct app]

controller

service

repo

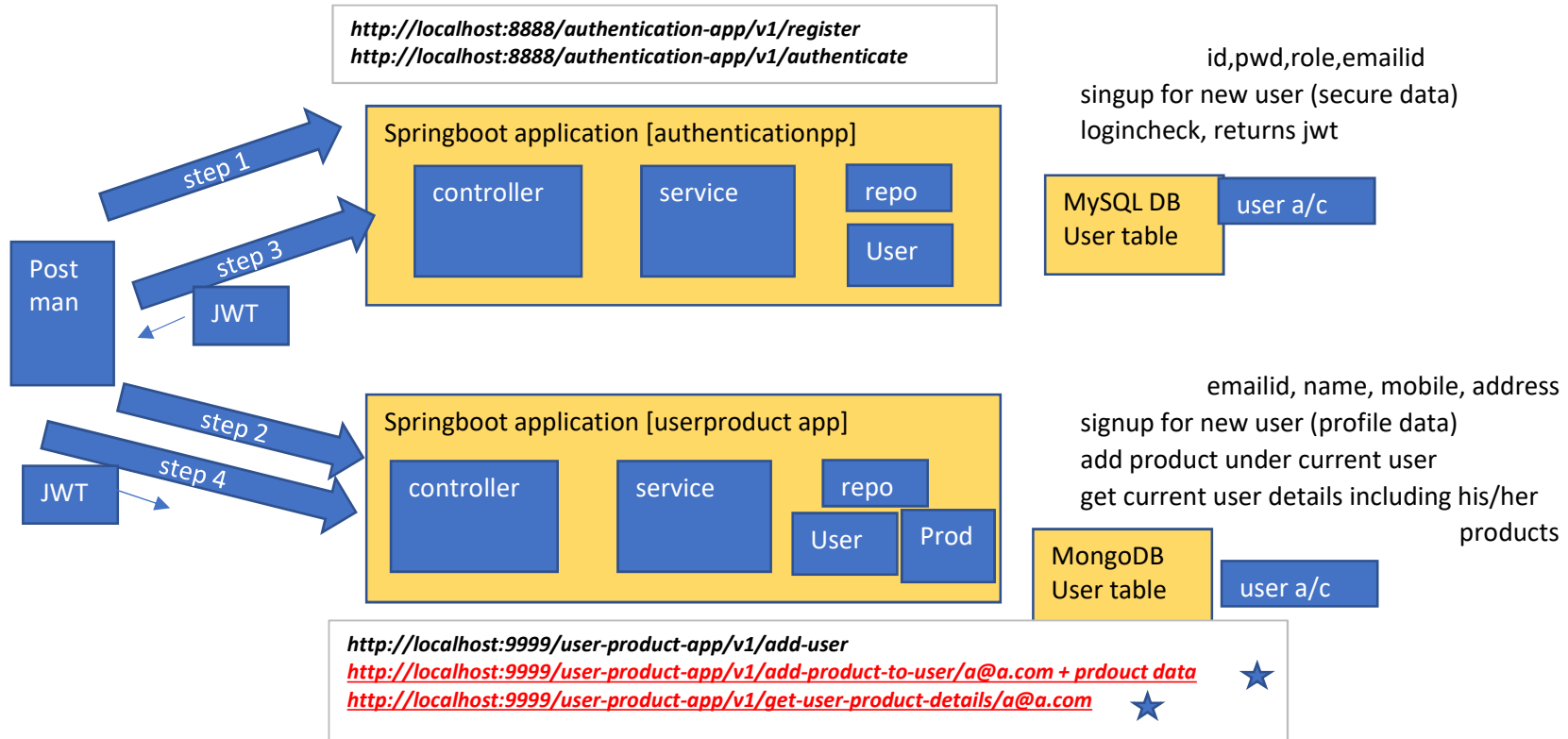
User

emailid, name, mobile, address
signup for new user (profile data)
add product under current user
get current user details including his/her products

MongoDB
User table

Stage 1

This both applications run in two different intellij windows



Note : signup process has below two steps

Step 1

Create new user account in auth app

<http://localhost:8888/authentication-app/v1/register>

emailId
password

ex1

raghu@gmail.com

12345

Step 2

Create same user account in product app

<http://localhost:9999/user-product-app/v1/add-user>

emailId mobileNo
userName address

Note : By completing above two steps user becomes member, he/she can login

Step 3 User login to app

<http://localhost:8888/authentication-app/v1/authenticate>

userId

password

Note: if authentication success, auth app returns valid JWT token as response

Step 4 Try to access any intercepted resource of product application by carrying token along with request

<http://localhost:9999/user-product-app/v1/get-user-product-details/a@a.com>

<http://localhost:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data>

Step 1

The screenshot displays a REST client interface for a POST request to `http://localhost:8888/authentication-app/v1/register`. The 'Body' tab is selected, showing a JSON payload with `emailId` and `password`. Below the request, the 'Body' tab of the response is shown, displaying a JSON object with `userId`, `password`, `role`, and `emailId`. The status is 200 OK.

```
POST http://localhost:8888/authentication-app/v1/register

{
  "emailId": "raghu@gmail.com",
  "password": "12345"
}
```

```
{
  "userId": 6,
  "password": "12345",
  "role": "ROLE_USER",
  "emailId": "raghu@gmail.com"
}
```

Step 2

POST ▼ http://localhost:9999/user-product-app/v1/add-user

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1  {
2    "emailId": "raghu@gmail.com",
3    "userName": "Raghu",
4    "mobileNo": "12423423",
5    "address": "Chennai"
6  }
```

Body Cookies Headers (5) Test Results 🌐 Status: 200 OK

Pretty Raw Preview Visualize **JSON** ▼ ≡

```
1  {
2    "emailId": "raghu@gmail.com",
3    "userName": "Raghu",
4    "mobileNo": "12423423",
5    "address": "Chennai",
6    "products": []
7  }
```

Signup is done for new user

Step 3 login

POST ▼ http://localhost:8888/authentication-app/v1/authenticate Send ▼

Params Auth Headers (8) **Body** ● Pre-req. Tests Settings Cookie:

raw ▼ **JSON** ▼ Beautify

```
1  {
2    "userId": 6,
3    "password": 12345
4  }
```

Body ▼ 🌐 200 OK 197 ms 464 B Save Response ▼

Pretty Raw Preview Visualize **JSON** ▼ ≡ 🔍

```
1  {
2    "message": "User successfully logged in",
3    "token": "eyJhbGciOiJIUzUxMiJ9.eyJ1c2VyT2JqZWNoIjp7InVzZXJJZCI6NiwiwGFzc3dvcmQiOiIiLCJyb2x1Ijo1Uk9MRV9VU0VSIIiwiaWZlhaWwJCI6InJhZ2h1QGdtYWlsLmNvbSJ9LCJpYXQiOiE2NjY5NDUzNTB9. r7hp0TSi8FAHxsV8fv8Qet0jxvz09HTyluvJUF9kxvEPeqSgvjfjJvWHptv17f4u7Qmg4wnsx0HEuq5JRZr7YQ"
4  }
```

Step 4 try to get access of intercepted urls attaching token in header

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://localhost:9999/user-product-app/v1/get-user-product-details/raghu@gmail.com`
- Authorization:** Bearer Token. The token value is `eyJhbGciOiJIUzUxMiJ9.eyJ1c2VyT2JqZWN...`
- Status:** 200 OK, Time: 360 ms, Size: 312 B
- Response Body (JSON):**

```
1  {
2    "emailId": "raghu@gmail.com",
3    "userName": "Raghu",
4    "mobileNo": "12423423",
5    "address": "Chennai",
6    "products": []
7  }
```

POST

http://localhost:9999/user-product-app/v1/add-product-to-user/raghu@gmail.com

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

```
1  {
2    ... "productname": "pencil",
3    ... "category": "stationary"
4  }
```

Body

Cookies

Headers (6)

Test Results

Status: 200 OK

Pretty

Raw

Preview

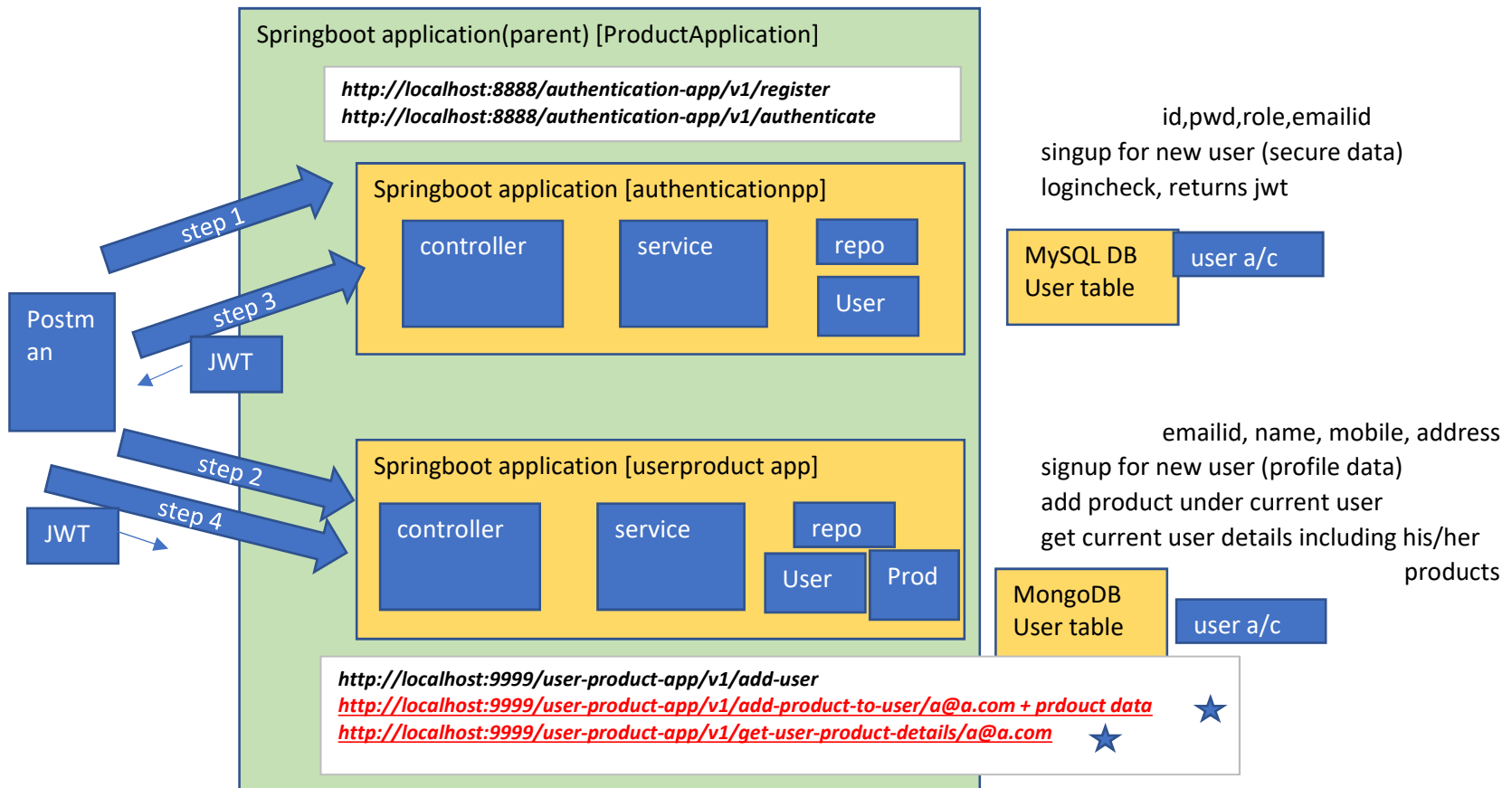
Visualize

JSON

```
1  {
2    "emailId": "raghu@gmail.com",
3    "userName": "Raghu",
4    "mobileNo": "12423423",
5    "address": "Chennai",
6    "products": [
7      {
8        "productname": "pencil",
9        "category": "stationary"
10     }
11   ]
12 }
```

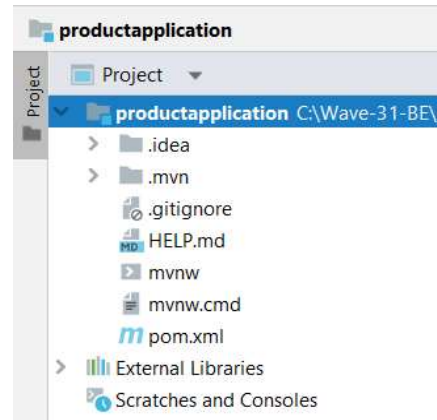
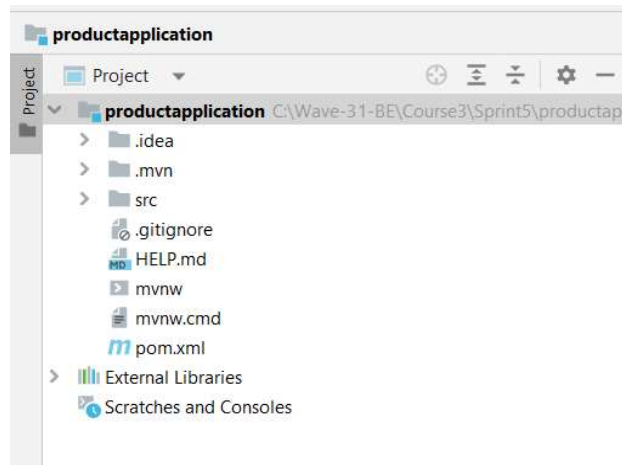
Stage 2

make both applications as child applications under a parent application
run these applications as microservices



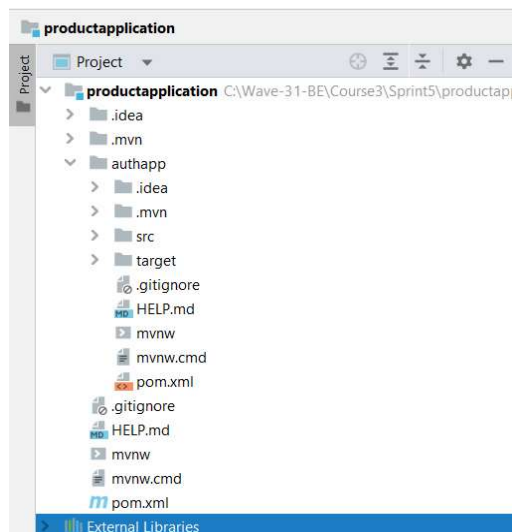
Here, prodcutapplication is parent
it holds other two applications as child applications
userproduct
authentication

- Step 1 Create parent application (springboot application)
Open in intelliJ
remove src folder



- Step 2 Copy child application into parent application folder

authentication

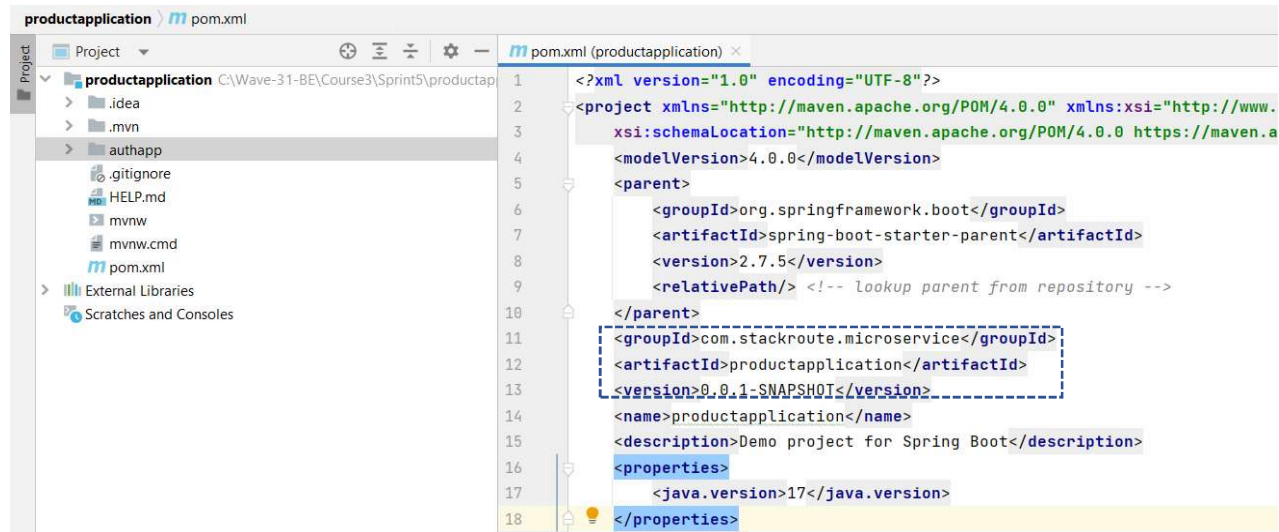


Step 3 Create parent child relation between 'productapplication' and 'authapp'

- a copy parent application details into child application under parent section

parent pom	child pom
groupId	<parent>
artifactId	
version	
	</parent>

parent pom.xml



```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.a
<modelVersion>4.0.0</modelVersion>
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.7.5</version>
  <relativePath/> <!-- lookup parent from repository -->
</parent>
<groupId>com.stackroute.microservice</groupId>
<artifactId>productapplication</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>productapplication</name>
<description>Demo project for Spring Boot</description>
<properties>
  <java.version>17</java.version>
</properties>
```

copy the above info from parent pom to child pom

productapplication \ authapp \ pom.xml

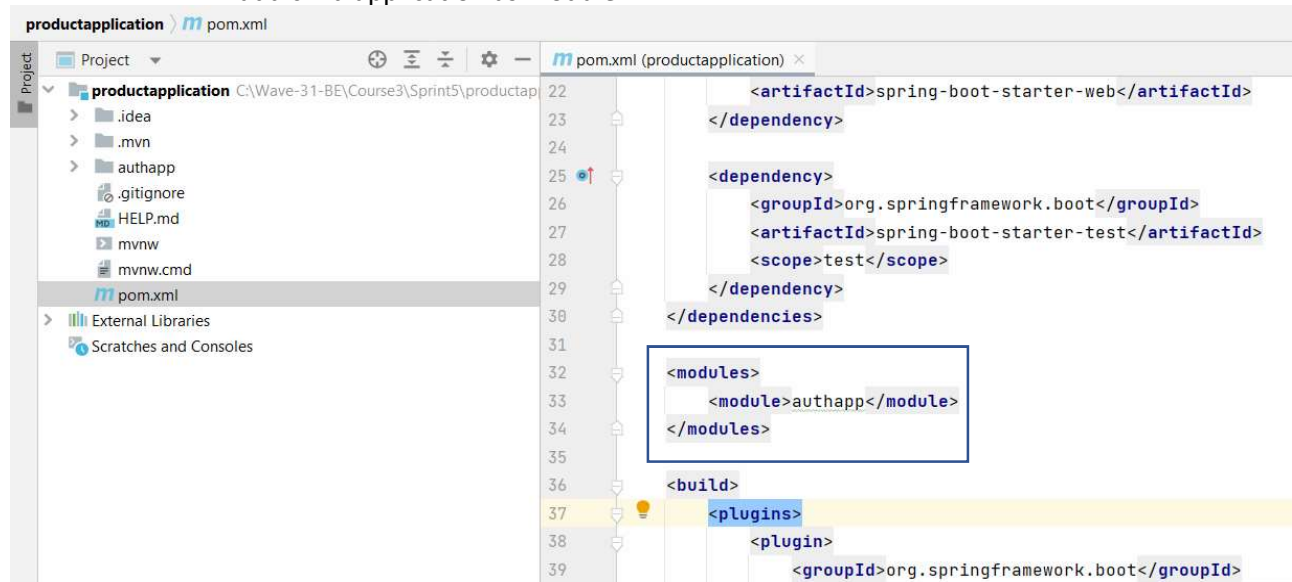
```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.7.4</version>
<relativePath/> <!-- Lookup parent from repository -->
</parent>
<groupId>com.stackroute.jwt</groupId>
<artifactId>authapp</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>authapp</name>
<description>Demo project for Spring Boot</description>
<properties>
<java.version>17</java.version>
</properties>
</project>
```

replace this content with parent application details

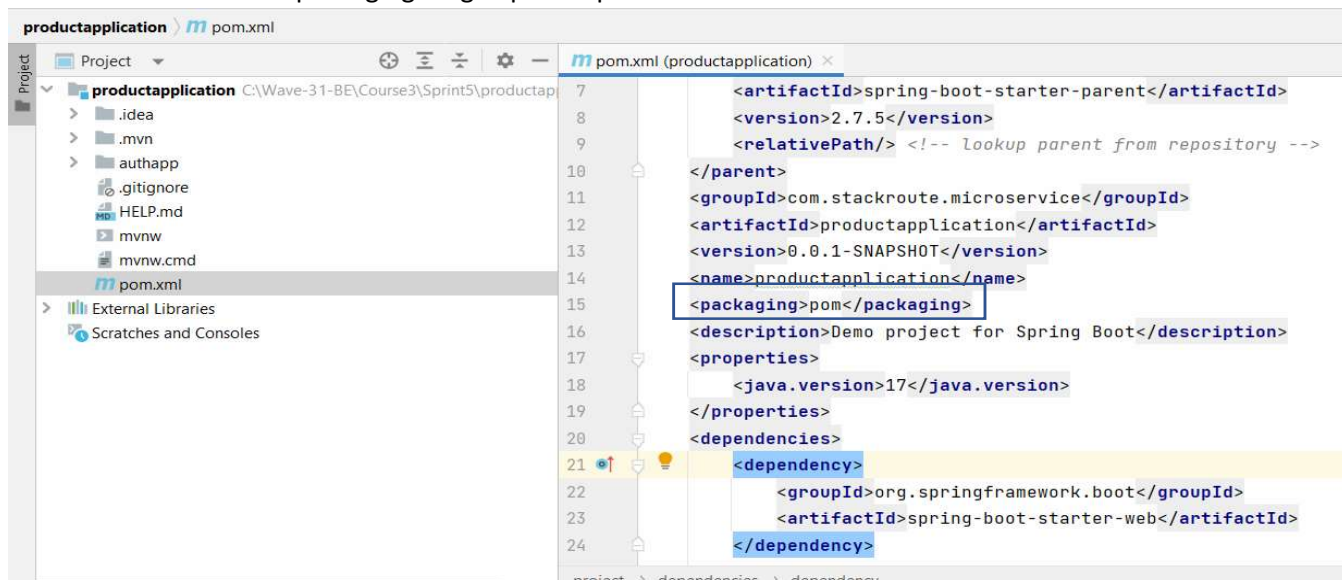
productapplication \ authapp \ pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>com.stackroute.microservice</groupId>
<artifactId>productapplication</artifactId>
<version>0.0.1-SNAPSHOT</version>
</parent>
<groupId>com.stackroute.jwt</groupId>
<artifactId>authapp</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>authapp</name>
<description>Demo project for Spring Boot</description>
<properties>
<java.version>17</java.version>
</properties>
<dependencies>
```

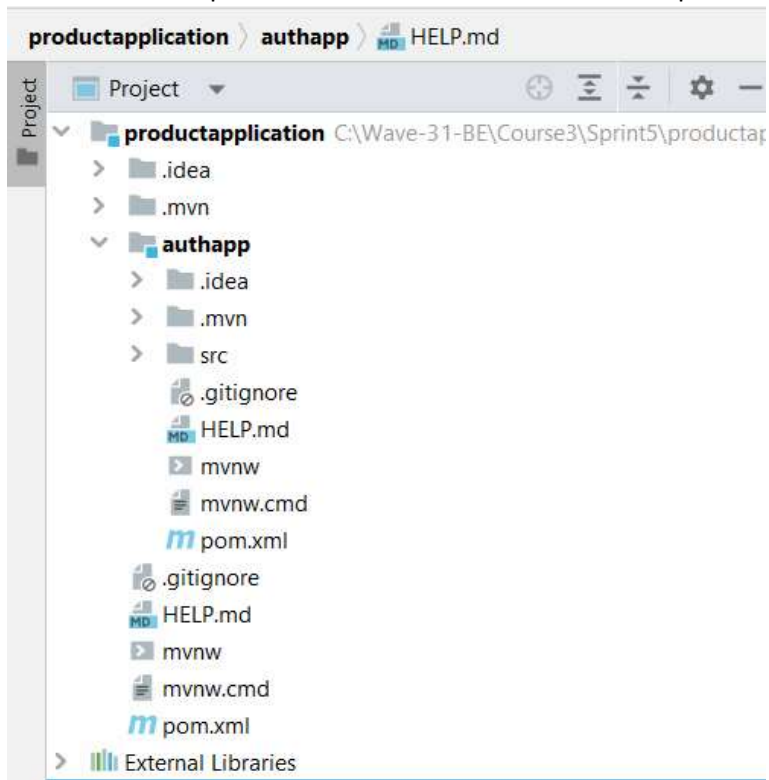
- b in parent pom
add child application as module



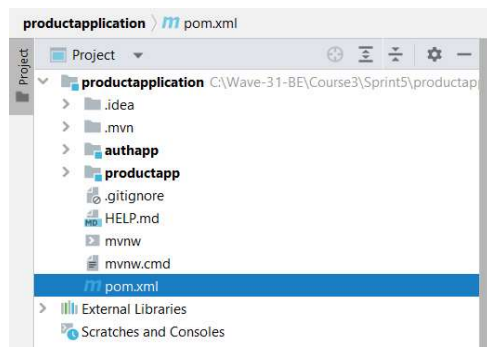
- c add <packaging> tag in parent pom



Make sure parent child relation created between 'productapplication' and 'authapp'



Step 4 Create parent child relation between 'productapplication' and 'productapp'



Start applications individual

