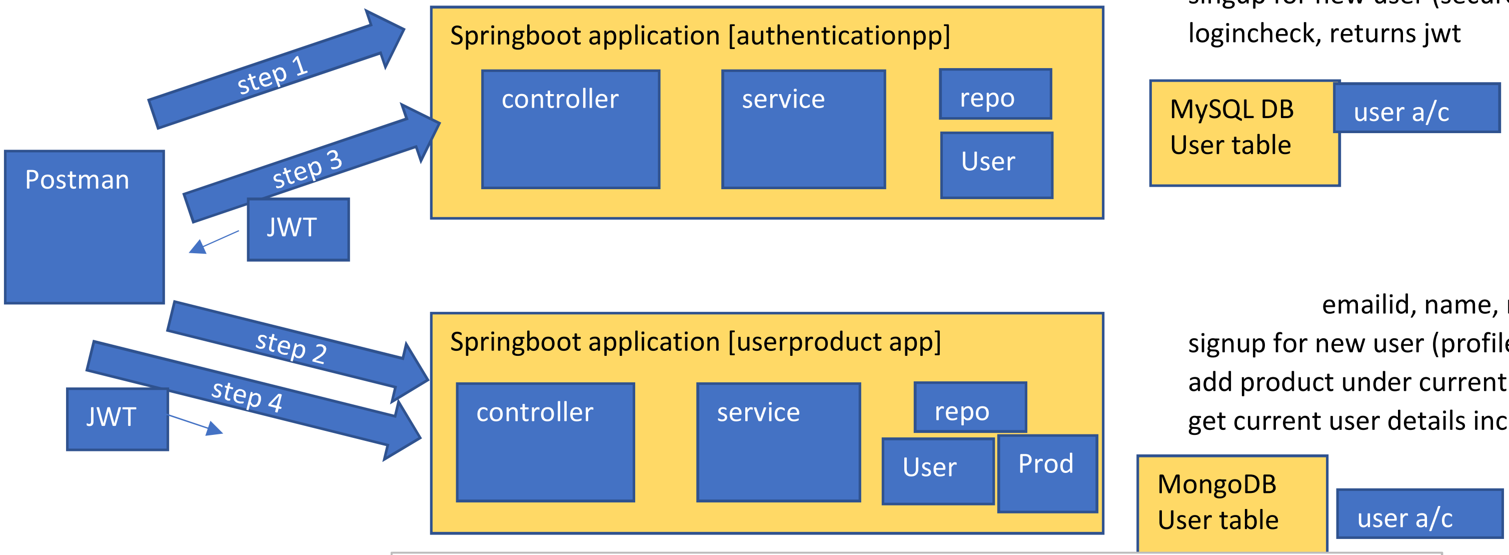
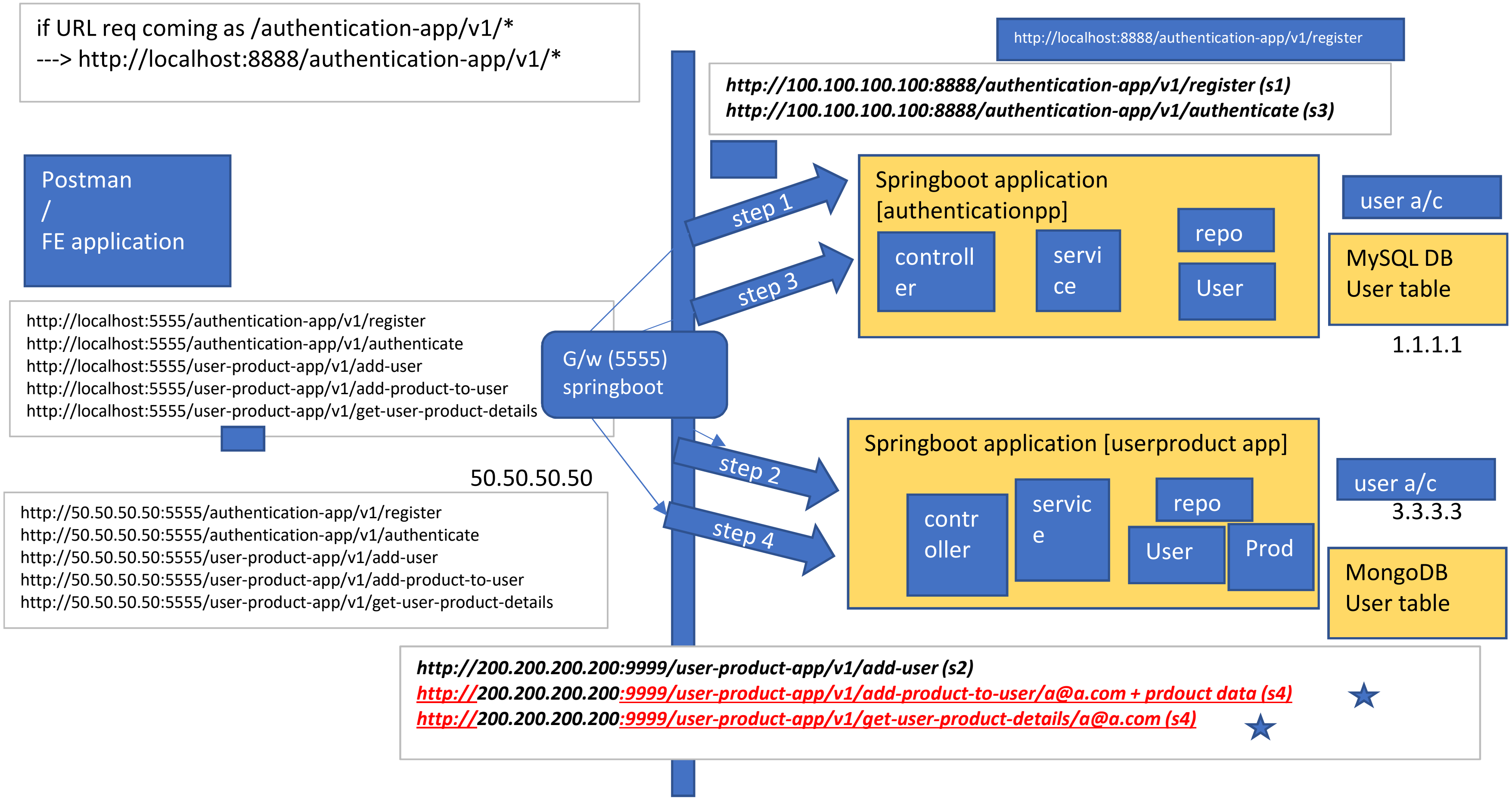
http://localhost:8888/authentication-app/v1/register (s1)
http://localhost:8888/authentication-app/v1/authenticate (s3)



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

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

http://localhost:9999/user-product-app/v1/add-user (s2)
http://localhost:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data (s4) ★
http://localhost:9999/user-product-app/v1/get-user-product-details/a@a.com (s4) ★



if URL req coming as /authentication-app/v1/*
---> http://localhost:8888/authentication-app/v1/*

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

http://100.100.100.100:8888/authentication-app/v1/register (s1)
http://100.100.100.100:8888/authentication-app/v1/authenticate (s3)

Postman
/
FE application

http://localhost:5555/authentication-app/v1/register
http://localhost:5555/authentication-app/v1/authenticate
http://localhost:5555/user-product-app/v1/add-user
http://localhost:5555/user-product-app/v1/add-product-to-user
http://localhost:5555/user-product-app/v1/get-user-product-details

G/w (5555)
springboot

50.50.50.50

http://50.50.50.50:5555/authentication-app/v1/register
http://50.50.50.50:5555/authentication-app/v1/authenticate
http://50.50.50.50:5555/user-product-app/v1/add-user
http://50.50.50.50:5555/user-product-app/v1/add-product-to-user
http://50.50.50.50:5555/user-product-app/v1/get-user-product-details

Springboot application
[authenticationpp]

MySQL DB
User table

1.1.1.1

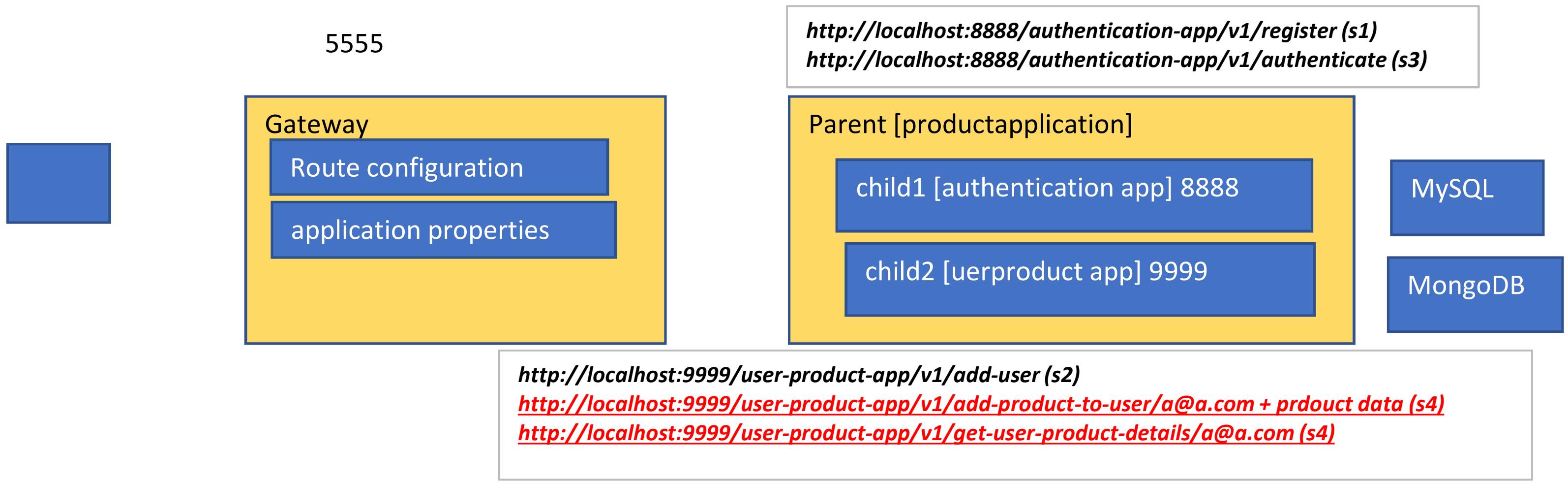
Springboot application [userproduct app]

3.3.3.3

MongoDB
User table

http://200.200.200.200:9999/user-product-app/v1/add-user (s2)
http://200.200.200.200:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data (s4) ★
http://200.200.200.200:9999/user-product-app/v1/get-user-product-details/a@a.com (s4) ★

Demo 1



5555

Gateway

Route configuration

application properties

http://localhost:8888/authentication-app/v1/register (s1)
http://localhost:8888/authentication-app/v1/authenticate (s3)

Parent [productapplication]

child1 [authentication app] 8888

child2 [uerproduct app] 9999

MySQL

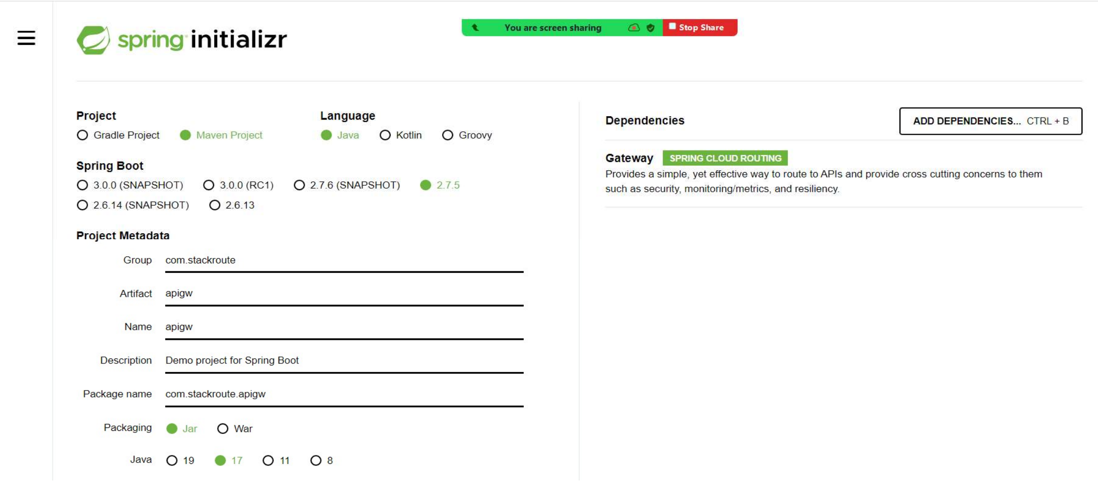
MongoDB

http://localhost:9999/user-product-app/v1/add-user (s2)
http://localhost:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data (s4)
http://localhost:9999/user-product-app/v1/get-user-product-details/a@a.com (s4)

Steps to create g/w springboot application

Step 1 Create springboot applciation in spring initializer adding required dependencies

Gateway



Project
○ Gradle Project ● Maven Project

Language
● Java ○ Kotlin ○ Groovy

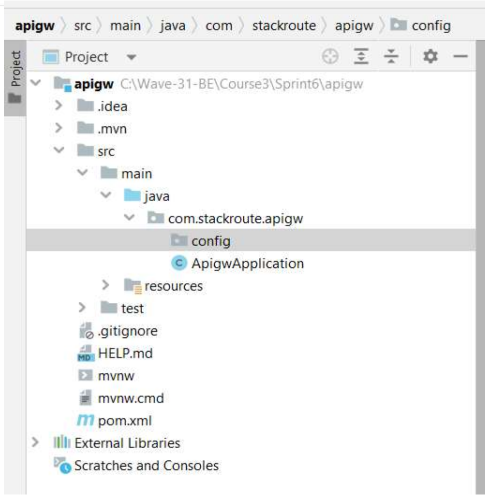
Spring Boot
○ 3.0.0 (SNAPSHOT) ○ 3.0.0 (RC1) ○ 2.7.6 (SNAPSHOT) ● 2.7.5
○ 2.6.14 (SNAPSHOT) ○ 2.6.13

Project Metadata

Group	com.stackroute
Artifact	apigw
Name	apigw
Description	Demo project for Spring Boot
Package name	com.stackroute.apigw
Packaging	● Jar ○ War
Java	○ 19 ● 17 ○ 11 ○ 8

Dependencies ADD DEPENDENCIES... CTRL + B

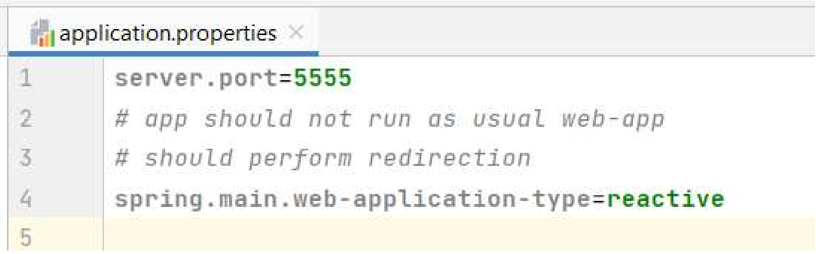
Gateway SPRING CLOUD ROUTING
Provides a simple, yet effective way to route to APIs and provide cross cutting concerns to them such as security, monitoring/metrics, and resiliency.



```
apigw 〉 src 〉 main 〉 java 〉 com 〉 stackroute 〉 apigw 〉 config

apigw C:\Wave-31-BE\Course3\Sprint6\apigw
  .idea
  .mvn
  src
    main
      java
        com.stackroute.apigw
          config
          ApigwApplication
      resources
    test
  .gitignore
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
External Libraries
Scratches and Consoles
```

extract project and open in intellij
create required packages

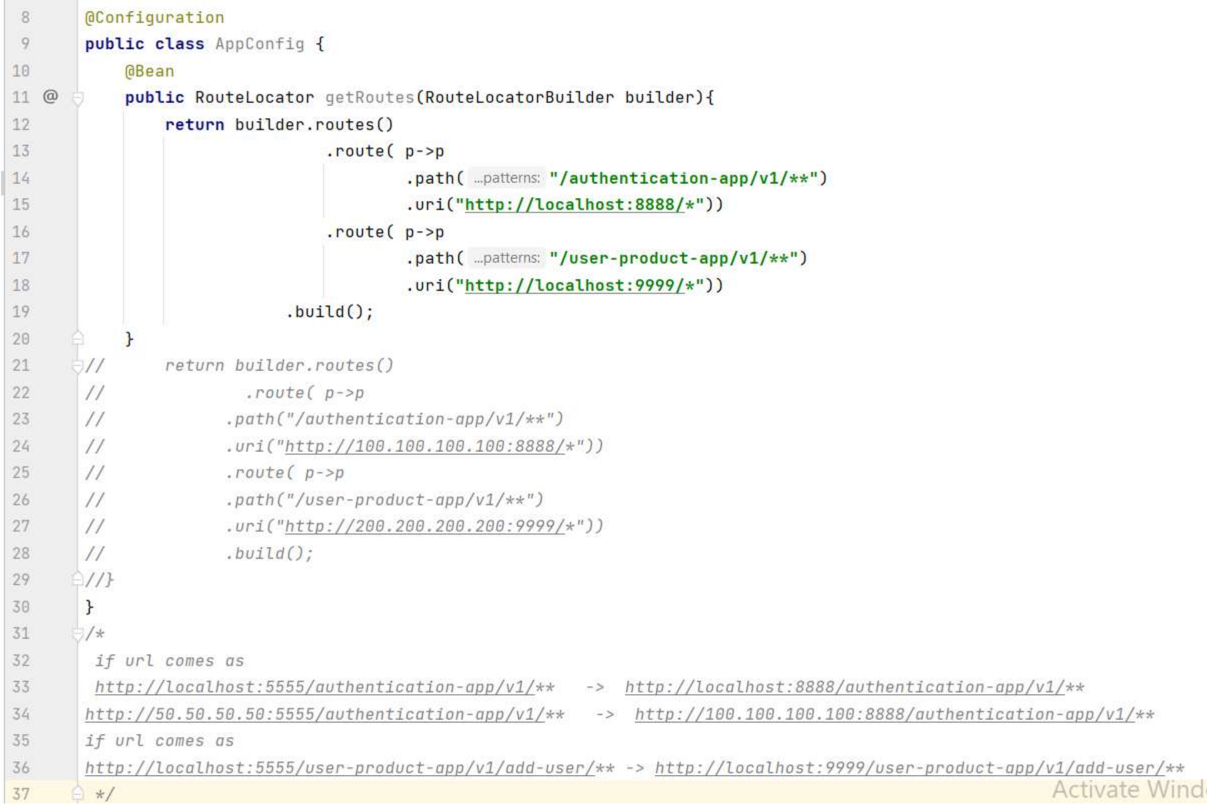
Step 2 application properties



application.properties

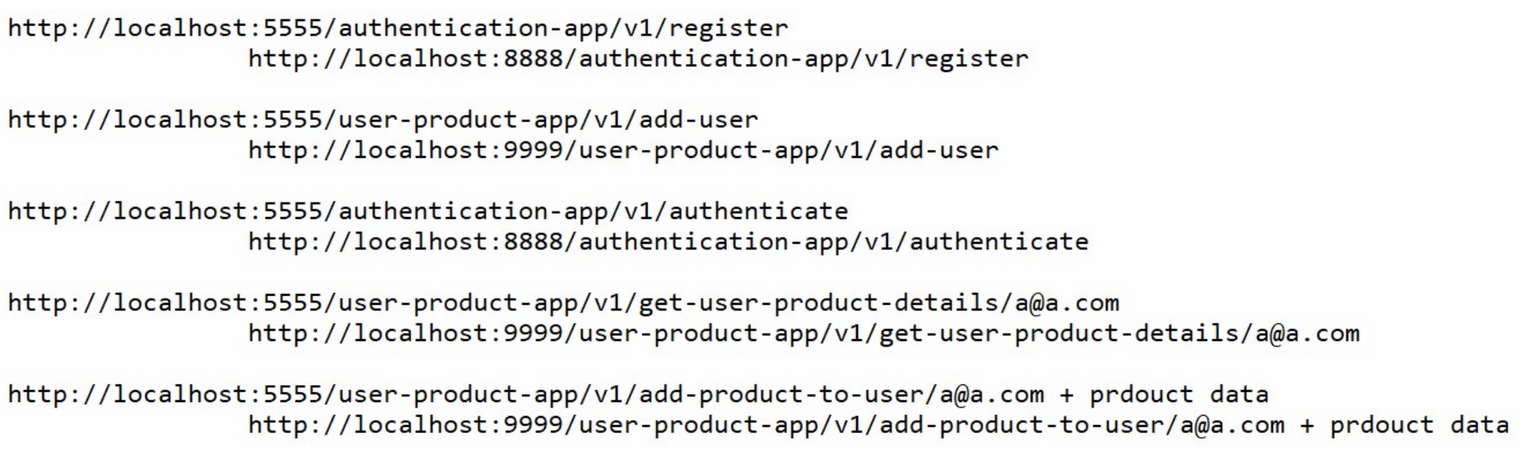
```
server.port=5555
# app should not run as usual web-app
# should perform redirection
spring.main.web-application-type=reactive

```

Step 3 Configure routes in configuration file



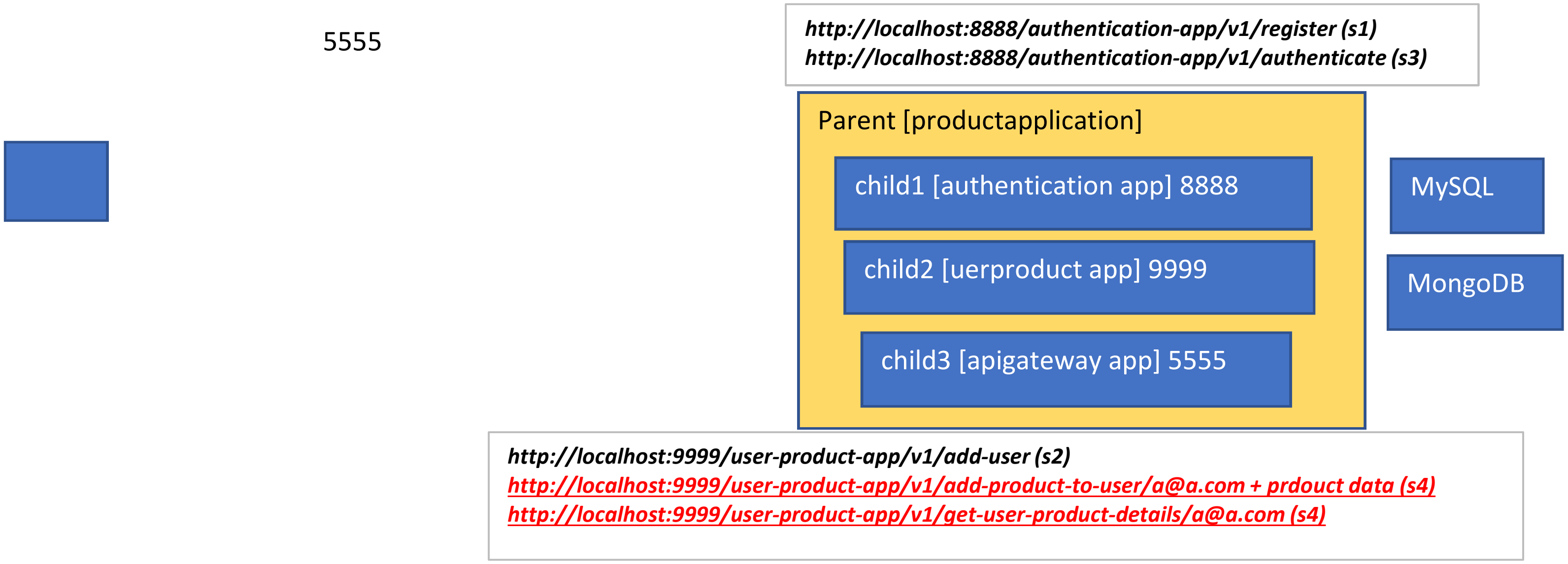
```
@Configuration
public class AppConfig {
    @Bean
    public RouteLocator getRoutes(RouteLocatorBuilder builder){
        return builder.routes()
                        .route( p->p
                                .path( ...patterns: "/authentication-app/v1/**")
                                .uri("http://localhost:8888/*"))
                        .route( p->p
                                .path( ...patterns: "/user-product-app/v1/**")
                                .uri("http://localhost:9999/*"))
                    .build();
    }
//      return builder.routes()
//              .route( p->p
//            .path("/authentication-app/v1/**")
//            .uri("http://100.100.100.100:8888/*"))
//            .route( p->p
//            .path("/user-product-app/v1/**")
//            .uri("http://200.200.200.200:9999/*"))
//            .build();
//}
}
/*
 if url comes as
 http://localhost:5555/authentication-app/v1/**   ->  http://localhost:8888/authentication-app/v1/**
http://50.50.50.50:5555/authentication-app/v1/**   ->  http://100.100.100.100:8888/authentication-app/v1/**
if url comes as
http://localhost:5555/user-product-app/v1/add-user/** -> http://localhost:9999/user-product-app/v1/add-user/**
 */
```

app	7 6 5	
os	4 3	ip port
hw	2 1	



```
http://localhost:5555/authentication-app/v1/register
                http://localhost:8888/authentication-app/v1/register

http://localhost:5555/user-product-app/v1/add-user
                http://localhost:9999/user-product-app/v1/add-user

http://localhost:5555/authentication-app/v1/authenticate
                http://localhost:8888/authentication-app/v1/authenticate

http://localhost:5555/user-product-app/v1/get-user-product-details/a@a.com
                http://localhost:9999/user-product-app/v1/get-user-product-details/a@a.com

http://localhost:5555/user-product-app/v1/add-product-to-user/a@a.com + prdouct data
                http://localhost:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data
```

Demo 2



5555
Parent [productapplication]
child1 [authentication app] 8888
child2 [uerproduct app] 9999
child3 [apigateway app] 5555
MySQL
MongoDB

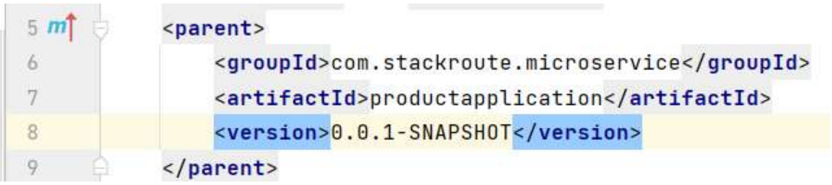
http://localhost:8888/authentication-app/v1/register (s1)
http://localhost:8888/authentication-app/v1/authenticate (s3)

http://localhost:9999/user-product-app/v1/add-user (s2)
http://localhost:9999/user-product-app/v1/add-product-to-user/a@a.com + prdouct data (s4)
http://localhost:9999/user-product-app/v1/get-user-product-details/a@a.com (s4)

Steps to make apigw application as child in existing microservice application

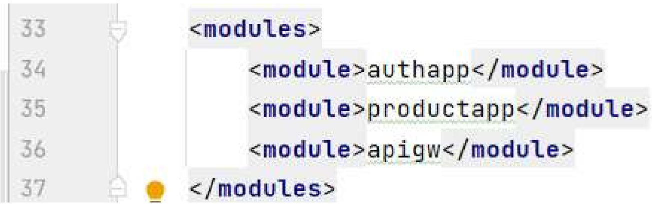
Step 1 copy apigw application to parent application folder

Step 2 edit pom in child (apigw)

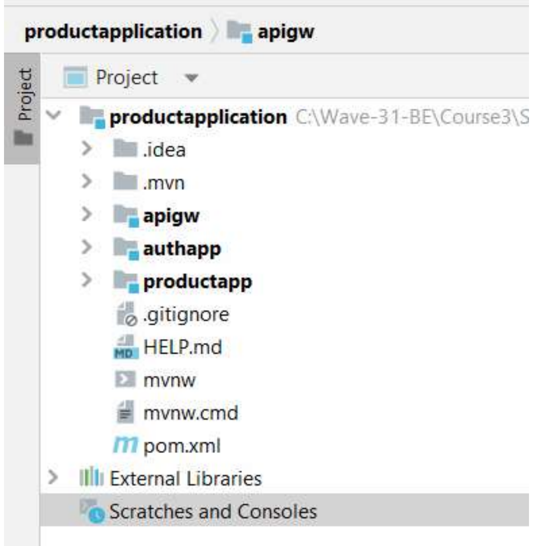


```
<parent>
    <groupId>com.stackroute.microservice</groupId>
    <artifactId>productapplication</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</parent>
```

Step 3 edit pom in parent (productapplication)



```
<modules>
    <module>authapp</module>
    <module>productapp</module>
    <module>apigw</module>
</modules>
```



productapplication 〉 apigw

Project

```
productapplication C:\Wave-31-BE\Course3\S
    .idea
    .mvn
    apigw
    authapp
    productapp
    .gitignore
    HELP.md
    mvnw
    mvnw.cmd
    pom.xml
External Libraries
Scratches and Consoles
```

Run all applications

Demo 3

Dockerizing Demo2 application

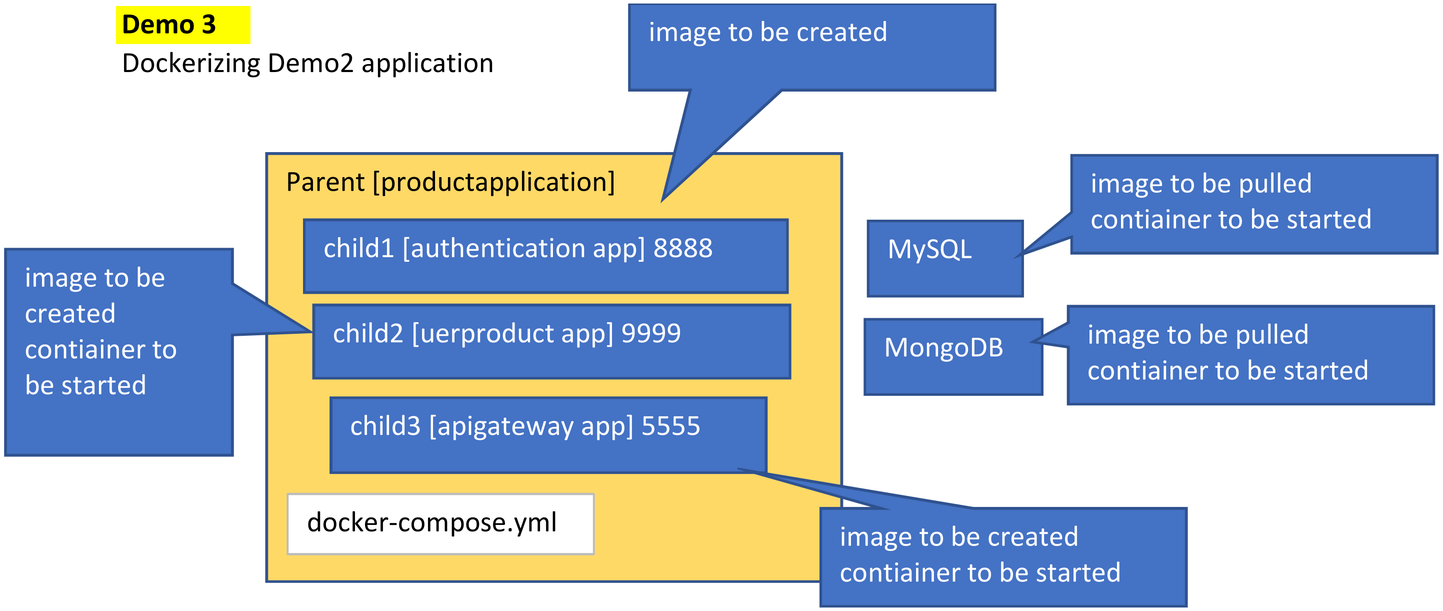


image to be created

Parent [productapplication]

child1 [authentication app] 8888

child2 [uerproduct app] 9999

child3 [apigateway app] 5555

docker-compose.yml

image to be created
contiainer to be started

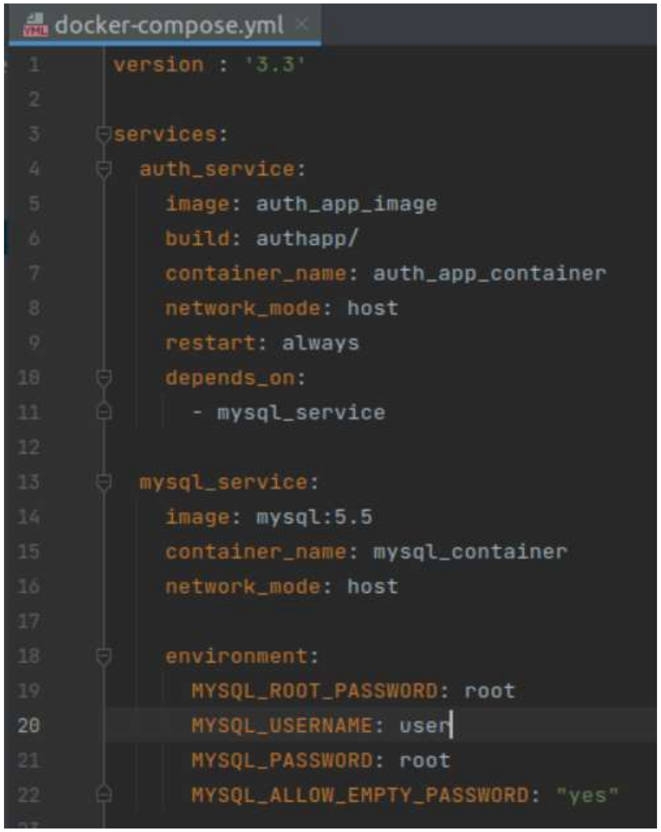
image to be created
contiainer to be started

MySQL

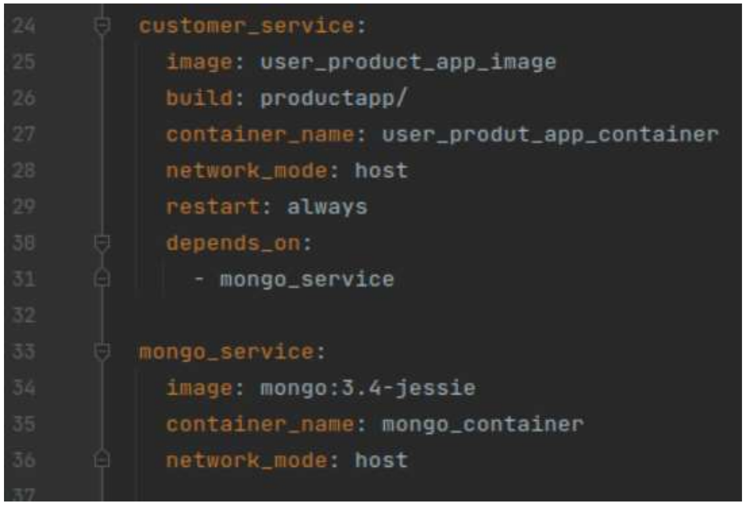
image to be pulled
contiainer to be started

MongoDB

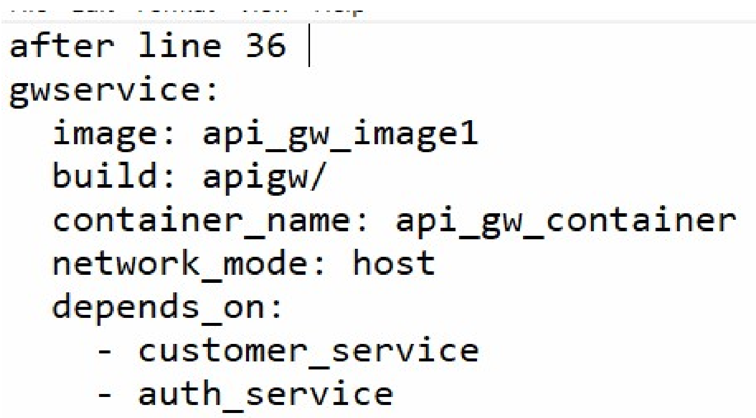
image to be pulled
contiainer to be started



```
1   version : '3.3'
2
3   services:
4     auth_service:
5       image: auth_app_image
6       build: authapp/
7       container_name: auth_app_container
8       network_mode: host
9       restart: always
10      depends_on:
11        - mysql_service
12
13    mysql_service:
14      image: mysql:5.5
15      container_name: mysql_container
16      network_mode: host
17
18      environment:
19        MYSQL_ROOT_PASSWORD: root
20        MYSQL_USERNAME: user
21        MYSQL_PASSWORD: root
22        MYSQL_ALLOW_EMPTY_PASSWORD: "yes"
```



```
24    customer_service:
25      image: user_product_app_image
26      build: productapp/
27      container_name: user_produt_app_container
28      network_mode: host
29      restart: always
30      depends_on:
31        - mongo_service
32
33    mongo_service:
34      image: mongo:3.4-jessie
35      container_name: mongo_container
36      network_mode: host
```



```
after line 36
gwservice:
  image: api_gw_image1
  build: apigw/
  container_name: api_gw_container
  network_mode: host
  depends_on:
    - customer_service
    - auth_service
```