

Projemizi aşağıdaki gibi oluşturalım.

New Project

Empty Project

Generators

Maven Archetype

Java Enterprise

Spring Initializr

JavaFX

Quarkus

Micronaut

Ktor

Kotlin Multiplatform

Compose Multiplatform

HTML

React

Express

Angular CLI

IDE Plugin

Android

Server URL: start.spring.io

Name: sb-02-spring-module-examples

Location: C:\Works\PROJECTS\001_Workspace\intellijidea\SB_004_SpringBootTutorial\repository\spring-boot-with-tech-tutorial

Project will be created in: C:\Works\PROJEC...tech-tutorial\sb-02-spring-module-examples

Create Git repository

Language: Java Kotlin Groovy

Type: Maven Gradle

Group: com.sb.gltkn

Artifact: sb-02-spring-module-examples

Package name: com.sb.gltkn

JDK: 11 Oracle OpenJDK version 11.0.15

Java: 11

Packaging: Jar War

New Project

Spring Boot: 2.7.0

Download pre-built shared indexes for JDK and Maven libraries

Dependencies:

Search

Developer Tools

Web

Spring Web

Spring Reactive Web

Spring for GraphQL

Rest Repositories

Spring Session

Rest Repositories HAL Explorer

Spring HATEOAS

Spring Web Services

Jersey

Vaadin

Template Engines

Security

SQL

NoSQL

Spring Web

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Building a RESTful Web Service

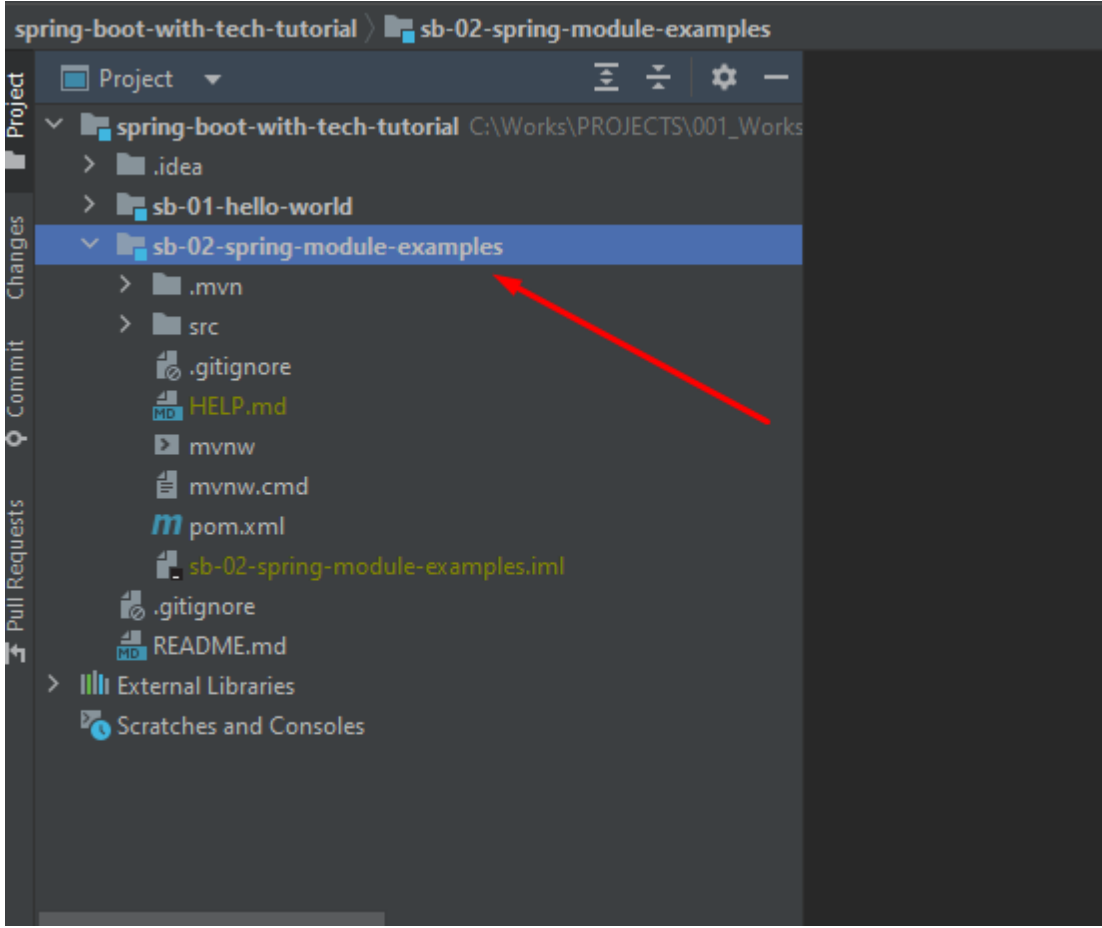
Serving Web Content with Spring MVC

Building REST services with Spring

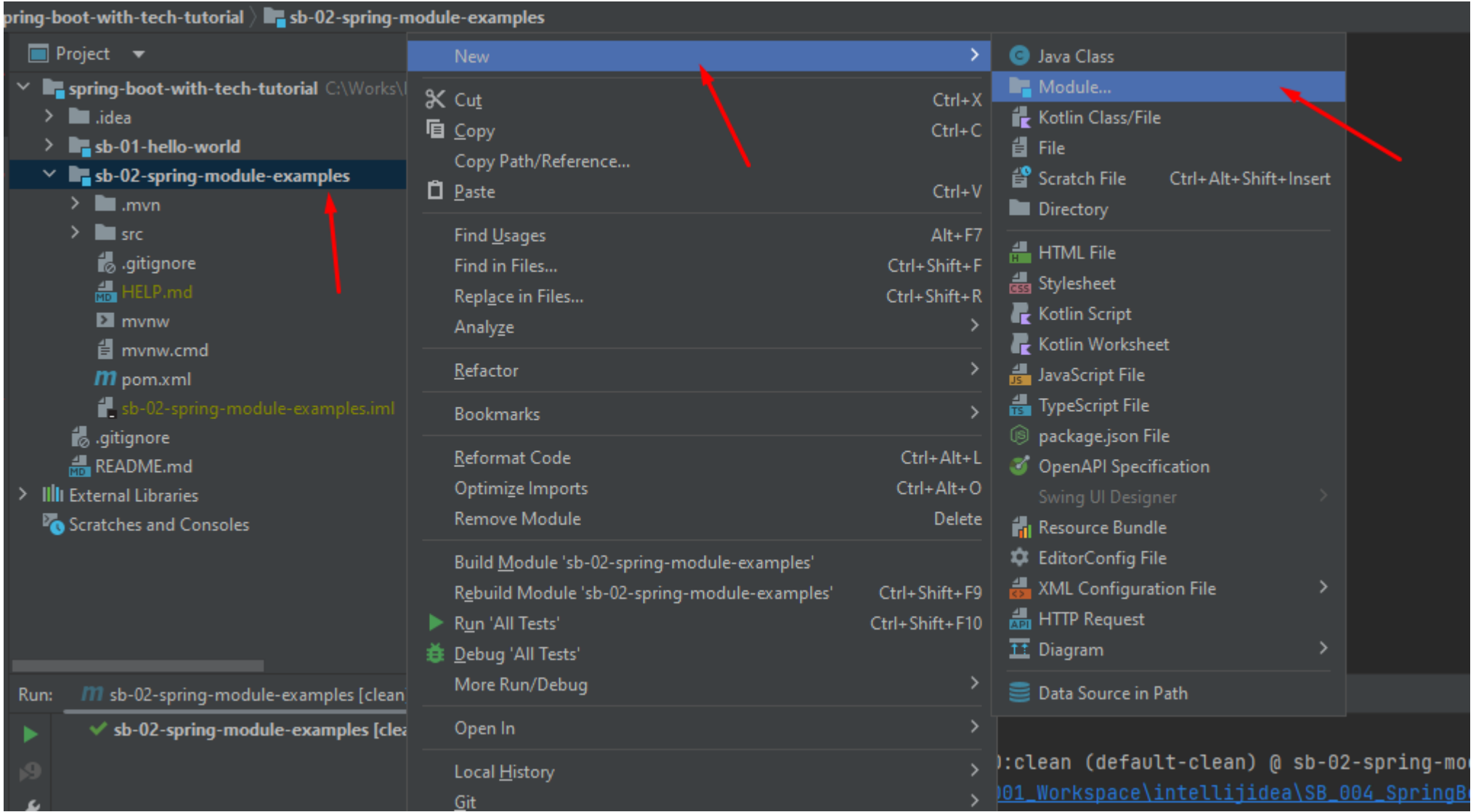
Added dependencies:

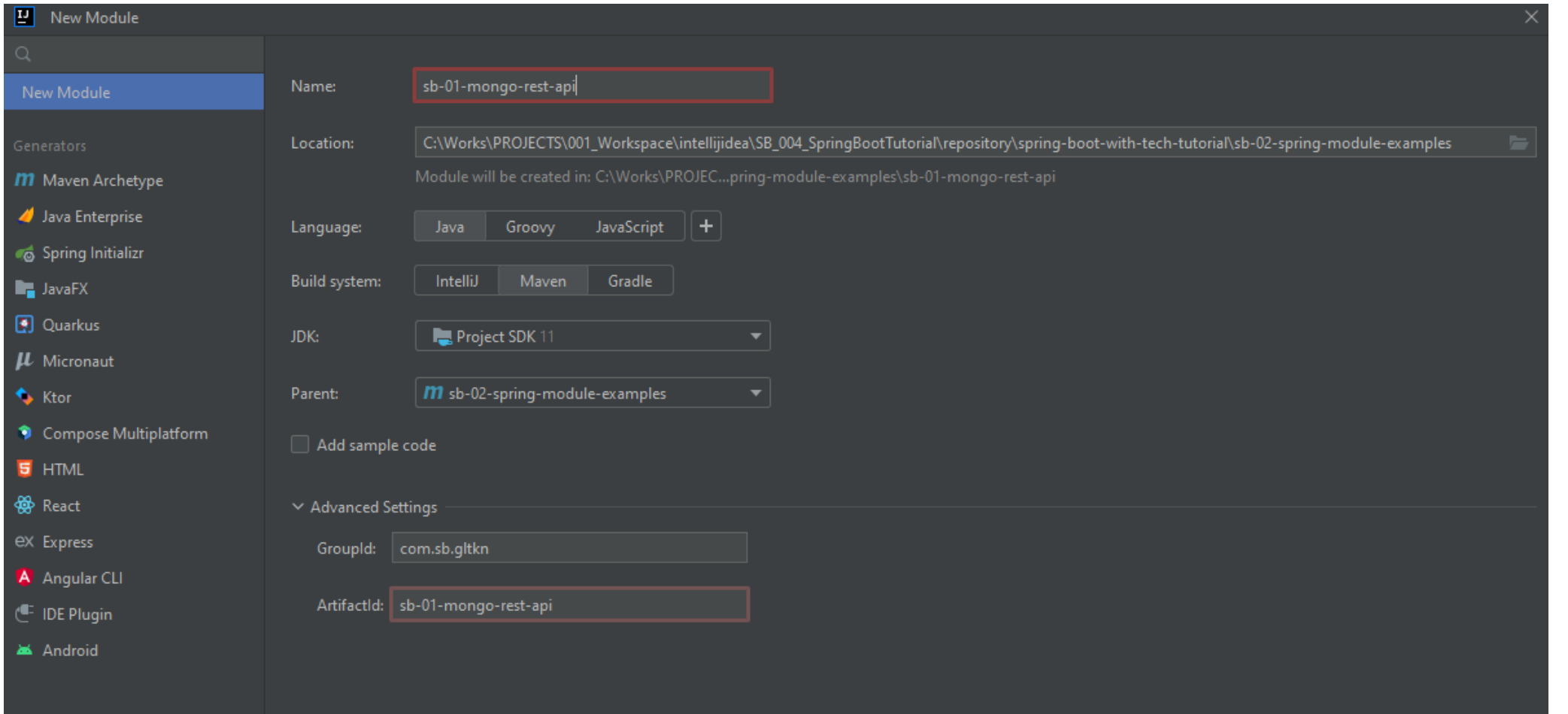
Spring Web

Altındaki proje içerisinde **src** klasörü, **.gitignore** ve **HELP.md** dosyalarını silebiliriz. Çünkü bu proje altına birazdan modüller oluşturacağız.



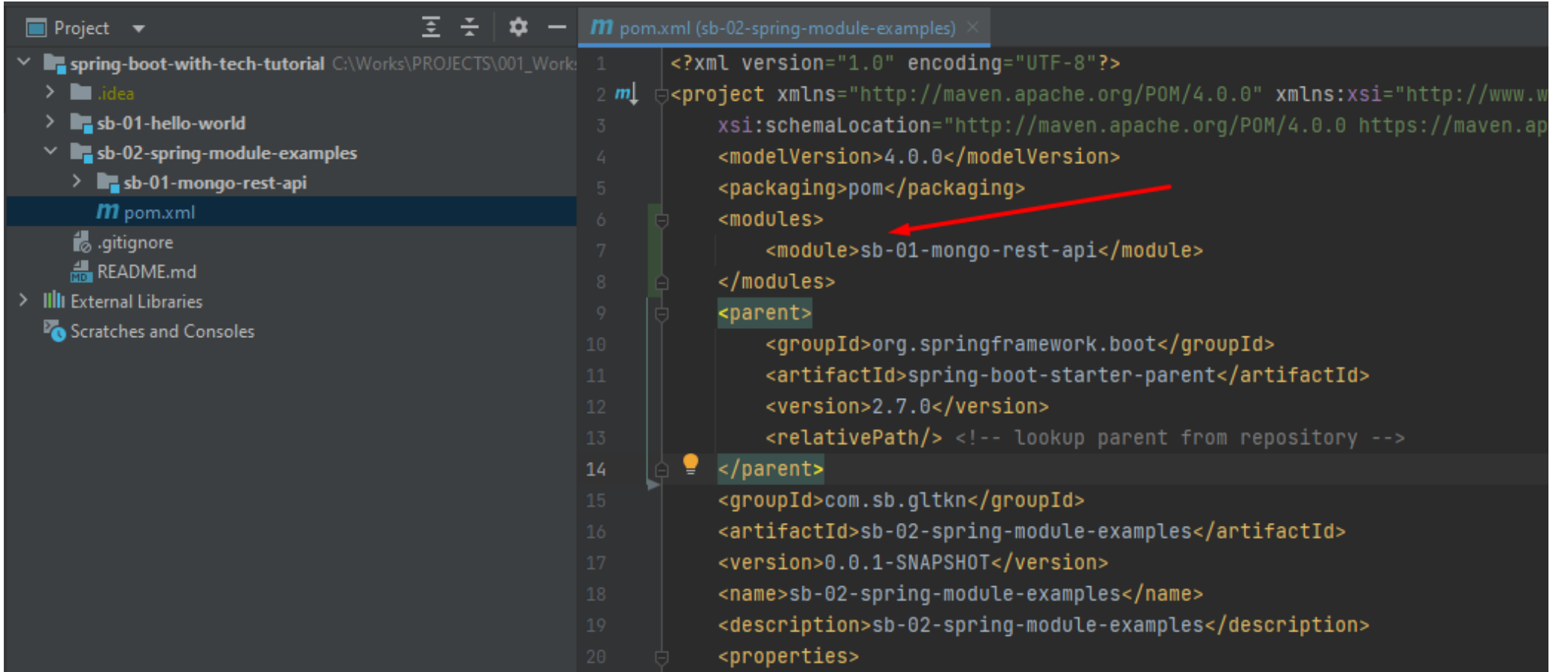
Projemize modül ekleyerek ilerleyeceğiz.



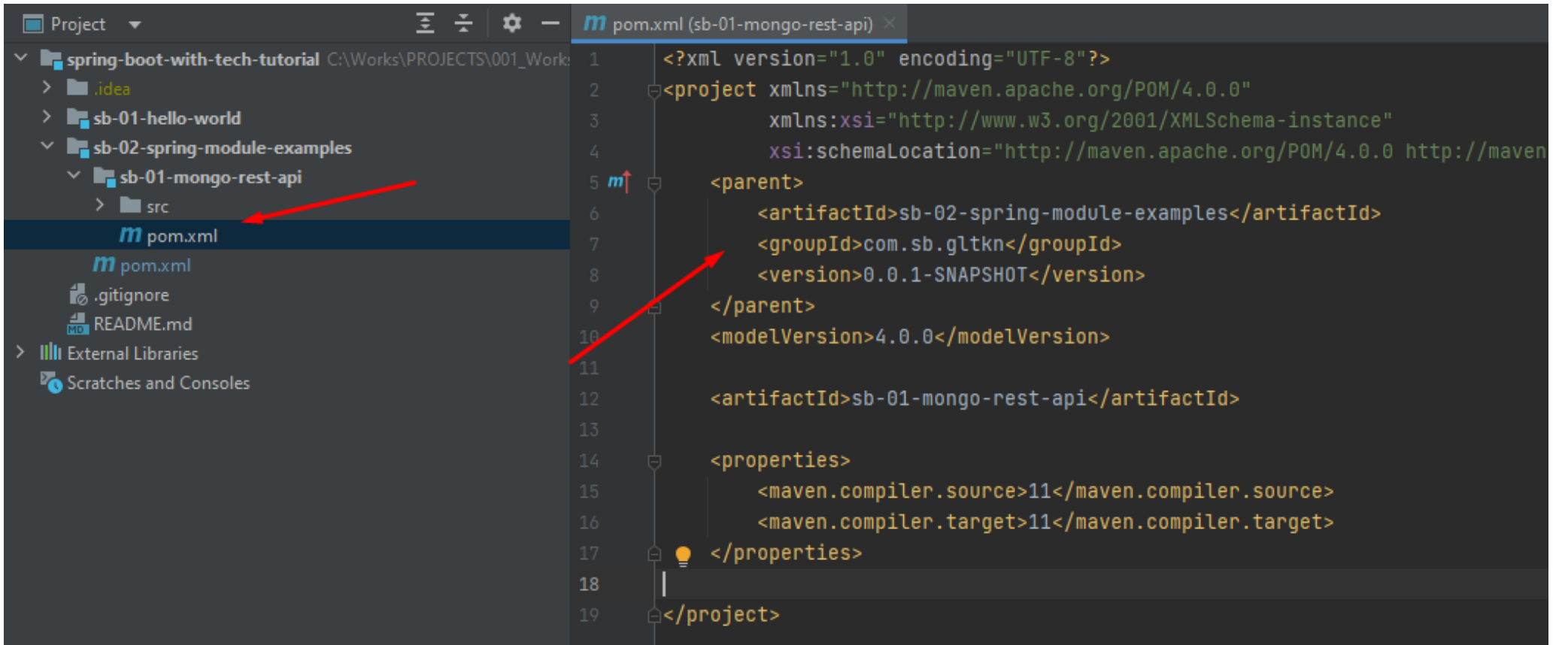


Parent projenin src dosyasını ve mvn dosyaları sildik ve pom.xml dosyasını da aşağıdaki gibi yaptık.

sb-02-spring-module-examples projesi'nin pom.xml'ine yeni eklediğimiz modül alttaki gibi otomatik eklenmiştir.



Bu yeni modül'ün pom.xml dosyasını alttaki gösterildiği gibi incelersek parent olarak **sb-02-spring-module-examples** projesinin otomatik verildiğini görebiliriz.



Mongo db kurulumu için docker hub'a gidelim.

Öncelikle docker'ı bilgisayarımıza kuralım.

Eğer Windows 10 Home kullanıyorsanız docker kurulumu esnasında hata alabilirsiniz.

Alttaki gibi kurulum gerçekleştirebilirsiniz.

İlk olarak oracle virtualBox indirelim ve bilgisayarımıza kuralım.

<https://www.oracle.com/tr/virtualization/technologies/vm/downloads/virtualbox-downloads.html>

<https://docker-docs.netlify.app/machine/get-started/#create-a-machine> adresinden yararlanıldı.

Git Bash ile aşağıdaki gibi docker-machine'leri listeledik.

docker-machine ls

default ismindeki makinemizi kaldırıyoruz.

docker-machine rm default

MINGW64:/c/Users/EmreGltkn

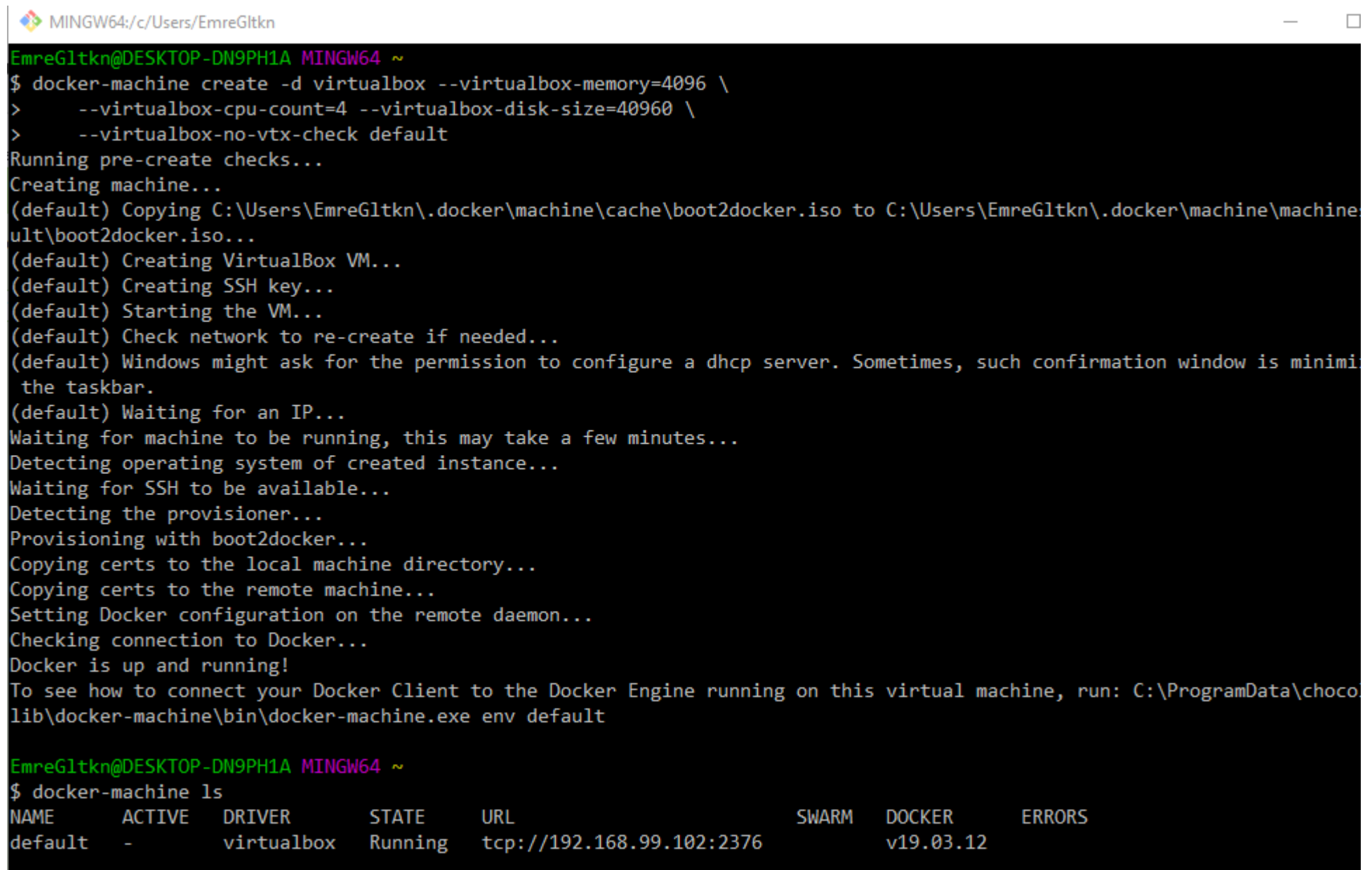
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine ls
NAME      ACTIVE  DRIVER        STATE   URL    SWARM   DOCKER  ERRORS
default   -       virtualbox    Error   URL    SWARM   DOCKER  machine does not exist

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine rm default
About to remove default
WARNING: This action will delete both local reference and remote instance.
Are you sure? (y/n): y
Successfully removed default
```

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine ls
NAME      ACTIVE  DRIVER        STATE   URL    SWARM   DOCKER  ERRORS
```

default isminde docker machine kurulumunu gerçekleştirelim.

```
docker-machine create -d virtualbox --virtualbox-memory=4096 \  
--virtualbox-cpu-count=4 --virtualbox-disk-size=40960 \  
--virtualbox-no-vtx-check default
```



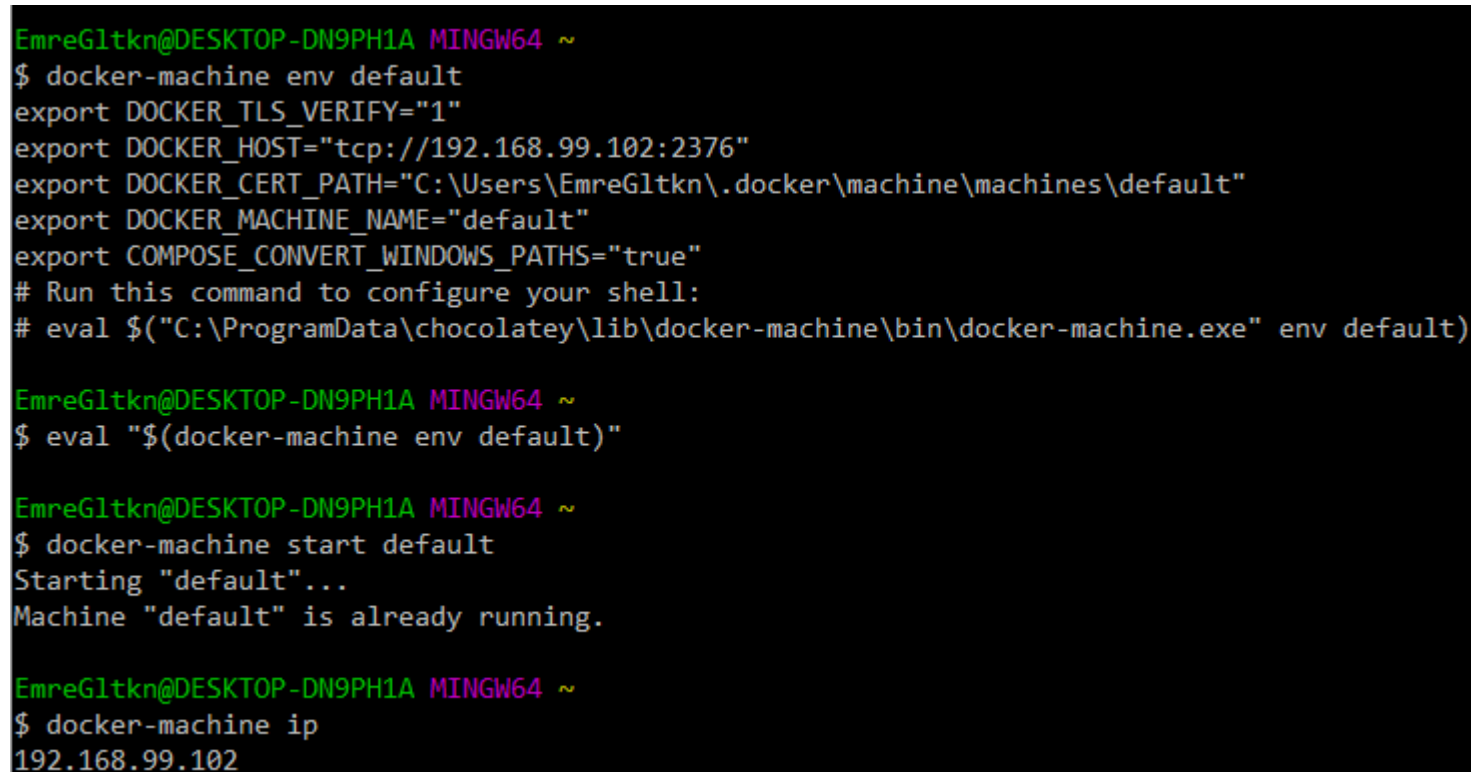
```
MINGW64:/c/Users/EmreGltkn  
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ docker-machine create -d virtualbox --virtualbox-memory=4096 \  
> --virtualbox-cpu-count=4 --virtualbox-disk-size=40960 \  
> --virtualbox-no-vtx-check default  
Running pre-create checks...  
Creating machine...  
(default) Copying C:\Users\EmreGltkn\.docker\machine\cache\boot2docker.iso to C:\Users\EmreGltkn\.docker\machine\machine-  
ult\boot2docker.iso...  
(default) Creating VirtualBox VM...  
(default) Creating SSH key...  
(default) Starting the VM...  
(default) Check network to re-create if needed...  
(default) Windows might ask for the permission to configure a dhcp server. Sometimes, such confirmation window is minimi  
the taskbar.  
(default) Waiting for an IP...  
Waiting for machine to be running, this may take a few minutes...  
Detecting operating system of created instance...  
Waiting for SSH to be available...  
Detecting the provisioner...  
Provisioning with boot2docker...  
Copying certs to the local machine directory...  
Copying certs to the remote machine...  
Setting Docker configuration on the remote daemon...  
Checking connection to Docker...  
Docker is up and running!  
To see how to connect your Docker Client to the Docker Engine running on this virtual machine, run: C:\ProgramData\chocol  
lib\docker-machine\bin\docker-machine.exe env default  
  
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ docker-machine ls  
NAME      ACTIVE   DRIVER      STATE     URL                          SWARM   DOCKER      ERRORS  
default   -        virtualbox   Running   tcp://192.168.99.102:2376      
v19.03.12
```

docker-machine env default

```
eval "$(docker-machine env default)"
```

docker-machine start default

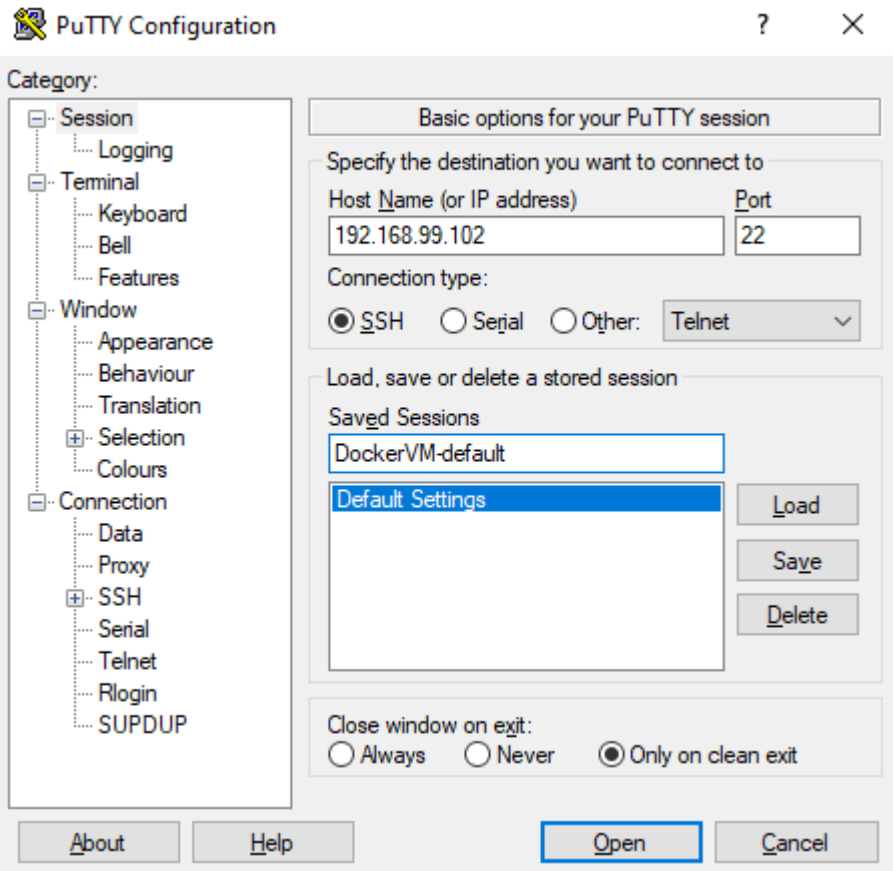
docker-machine ip



```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ docker-machine env default  
export DOCKER_TLS_VERIFY="1"  
export DOCKER_HOST="tcp://192.168.99.102:2376"  
export DOCKER_CERT_PATH="C:\Users\EmreGltkn\.docker\machine\machines\default"  
export DOCKER_MACHINE_NAME="default"  
export COMPOSE_CONVERT_WINDOWS_PATHS="true"  
# Run this command to configure your shell:  
# eval "$(C:\ProgramData\chocolatey\lib\docker-machine\bin\docker-machine.exe" env default)  
  
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ eval "$(docker-machine env default)"  
  
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ docker-machine start default  
Starting "default"...  
Machine "default" is already running.  
  
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~  
$ docker-machine ip  
192.168.99.102
```

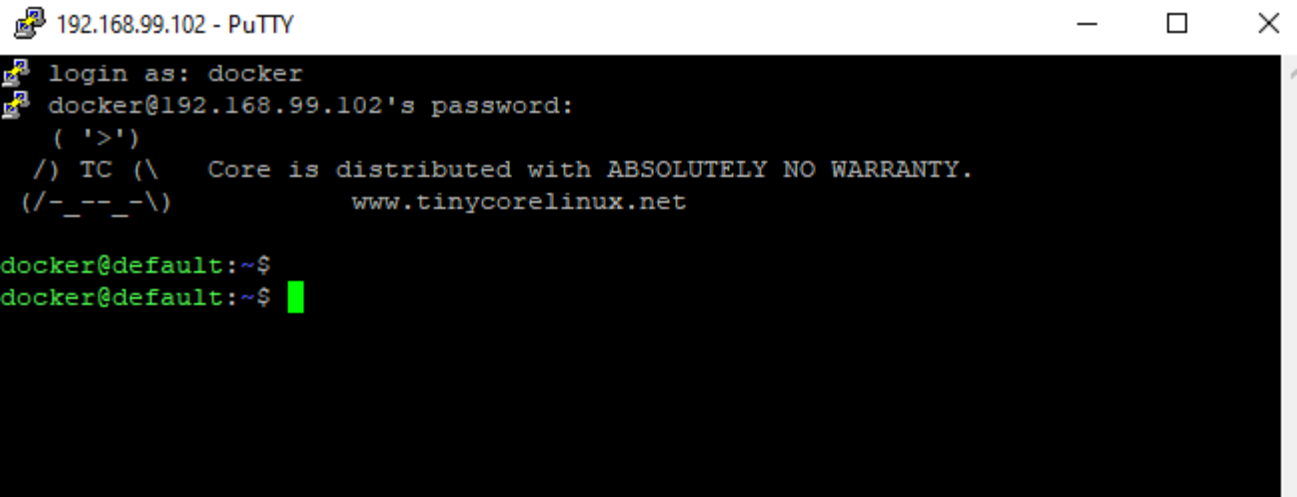
Docker Machine IP: 192.168.99.102

Şimdi docker makinemize putty üzerinden bağlanalım.



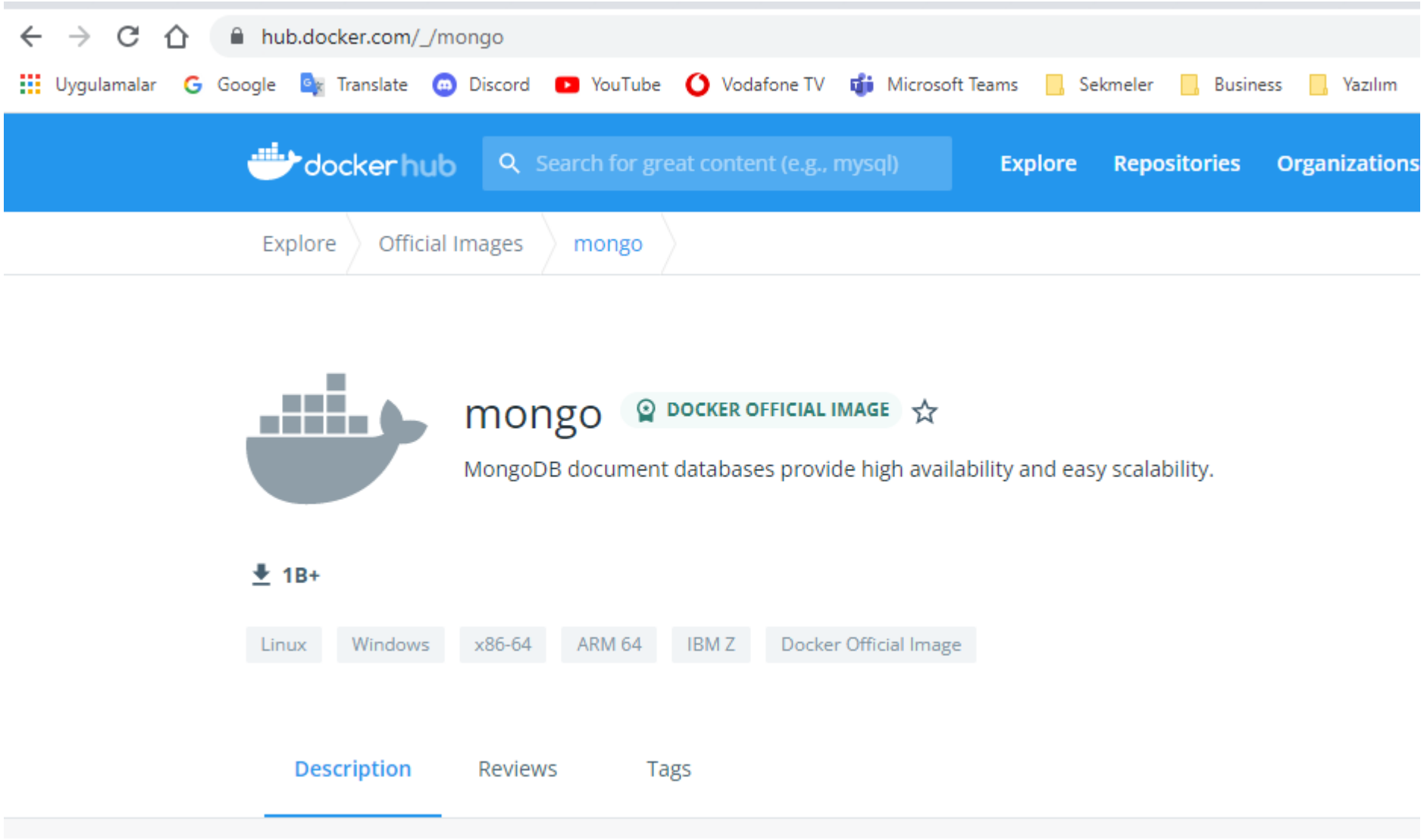
Default values;

Username:	docker
Password:	tcuser



Mongo db kurulumunu gerekleřtirelim.

https://hub.docker.com/_/mongo



docker run --name some-mongo -d mongo:tag

tag yerine versiyon bilgisi yazılır. Son versiyonu almak için ařağıdaki gibi latest yazılarak alıřtırılır.

docker run --name some-mongo -d mongo:latest

docker ps

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 /c/Program Files/Docker Toolbox
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
EmreGltkn@DESKTOP-DN9PH1A MINGW64 /c/Program Files/Docker Toolbox
$ _
```

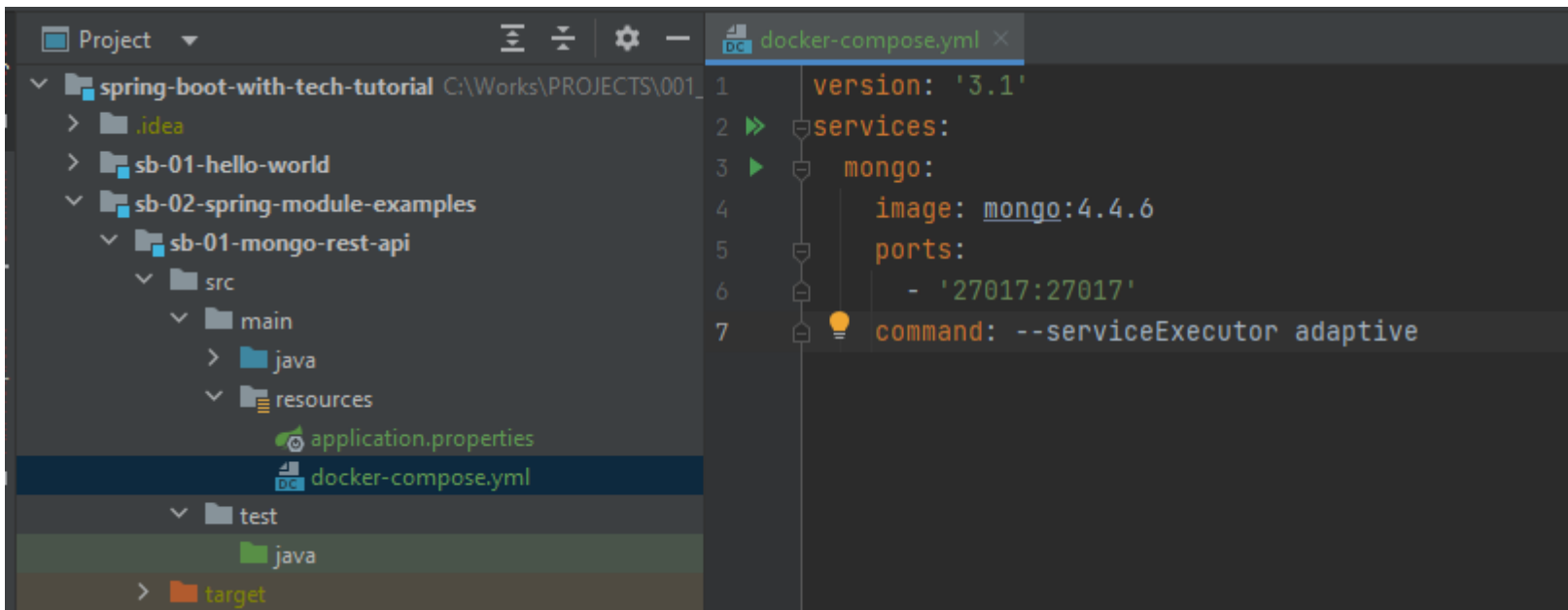
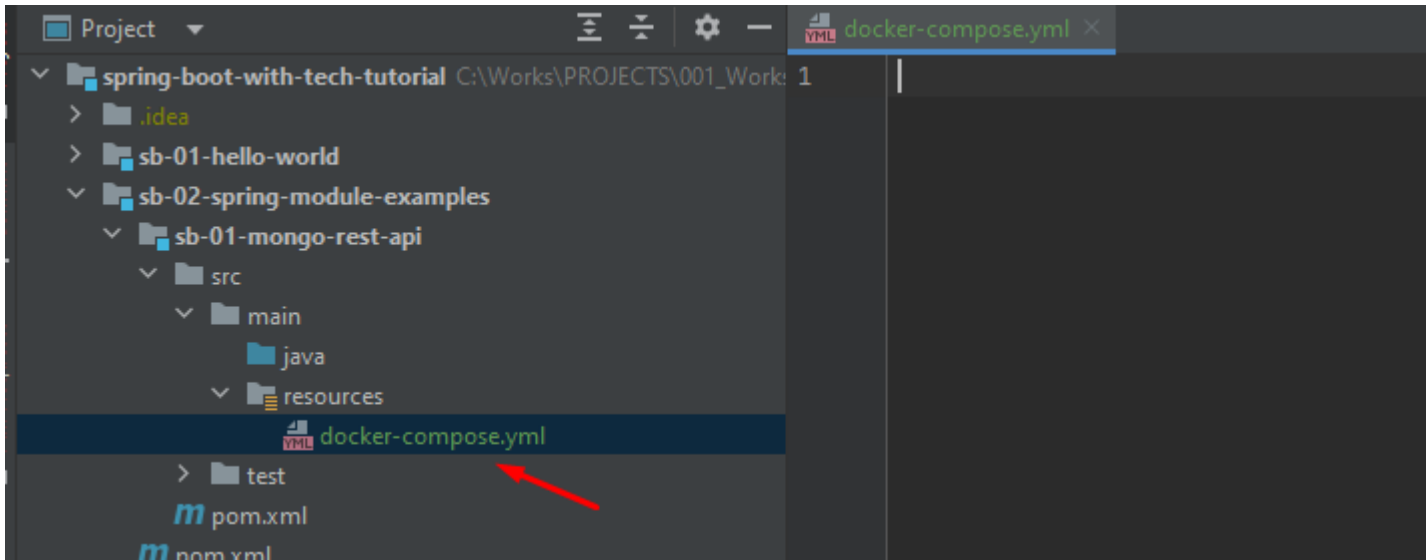
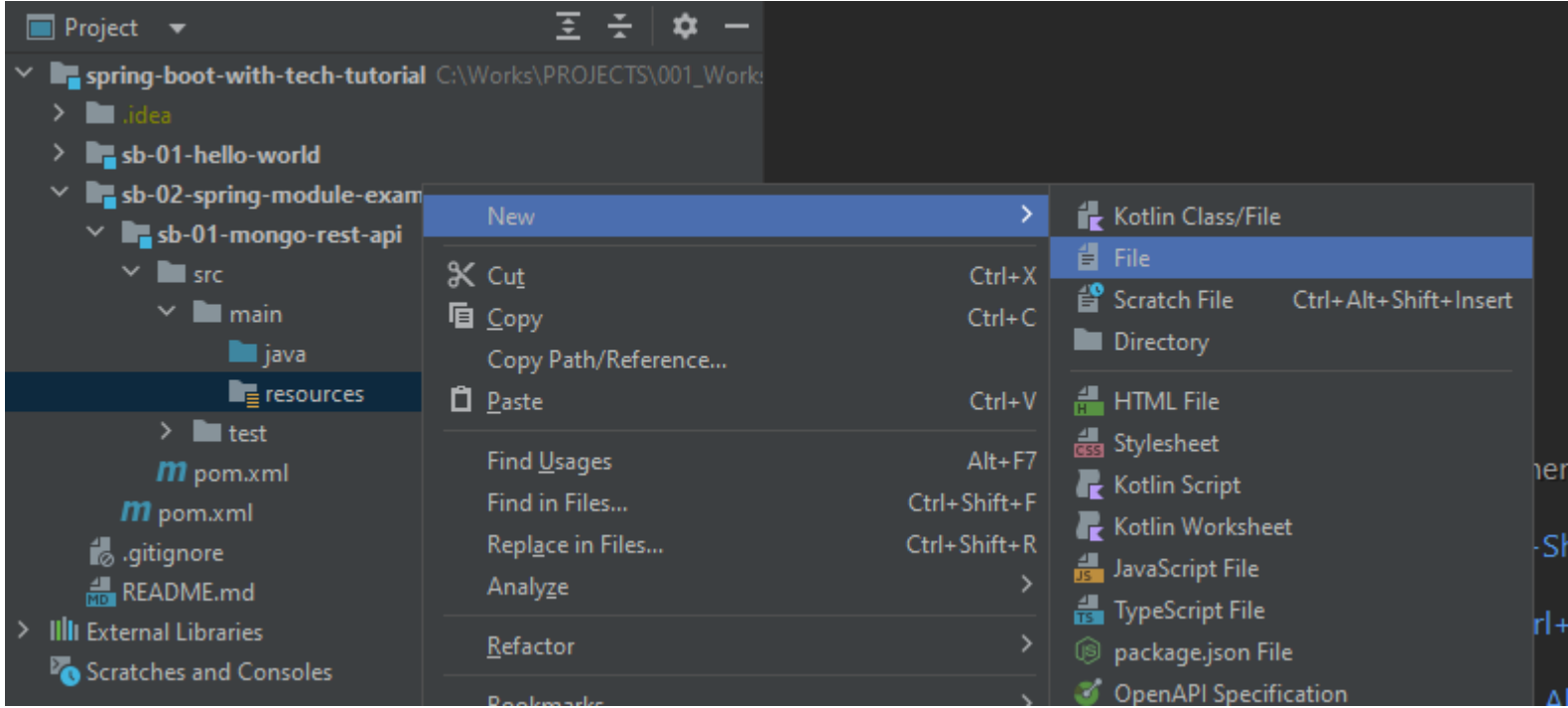
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 /c/Program Files/Docker Toolbox
$
EmreGltkn@DESKTOP-DN9PH1A MINGW64 /c/Program Files/Docker Toolbox
$ docker run --name some-mongo -d mongo:latest_
```

řimdi git bash ile docker image'larına bakalım.

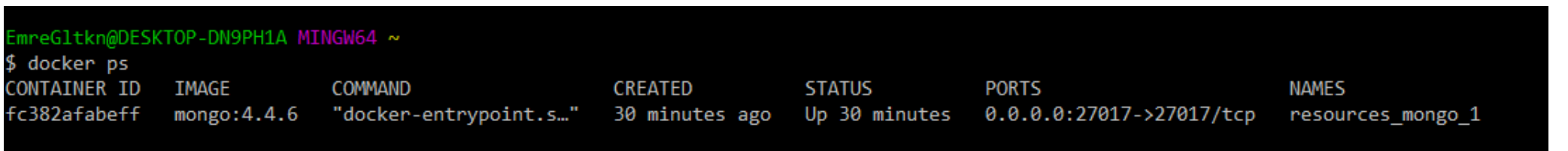
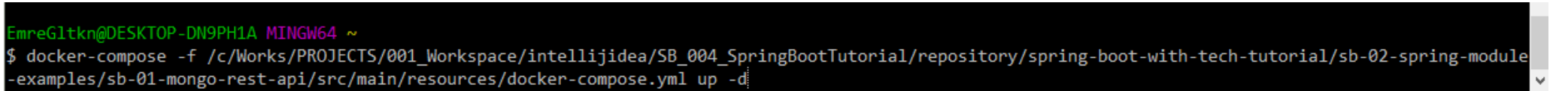
docker image ls

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mongo         4.4.6     61ea24dc52c6  11 months ago  423MB
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
```

Şimdi bir docker compose dosyası oluşturalım.



docker-compose -f /c/Works/PROJECTS/001_Workspace/intellijidea/SB_004_SpringBootTutorial/repository/spring-boot-with-tech-tutorial/sb-02-spring-module-examples/sb-01-mongo-rest-api/src/main/resources/docker-compose.yml up -d

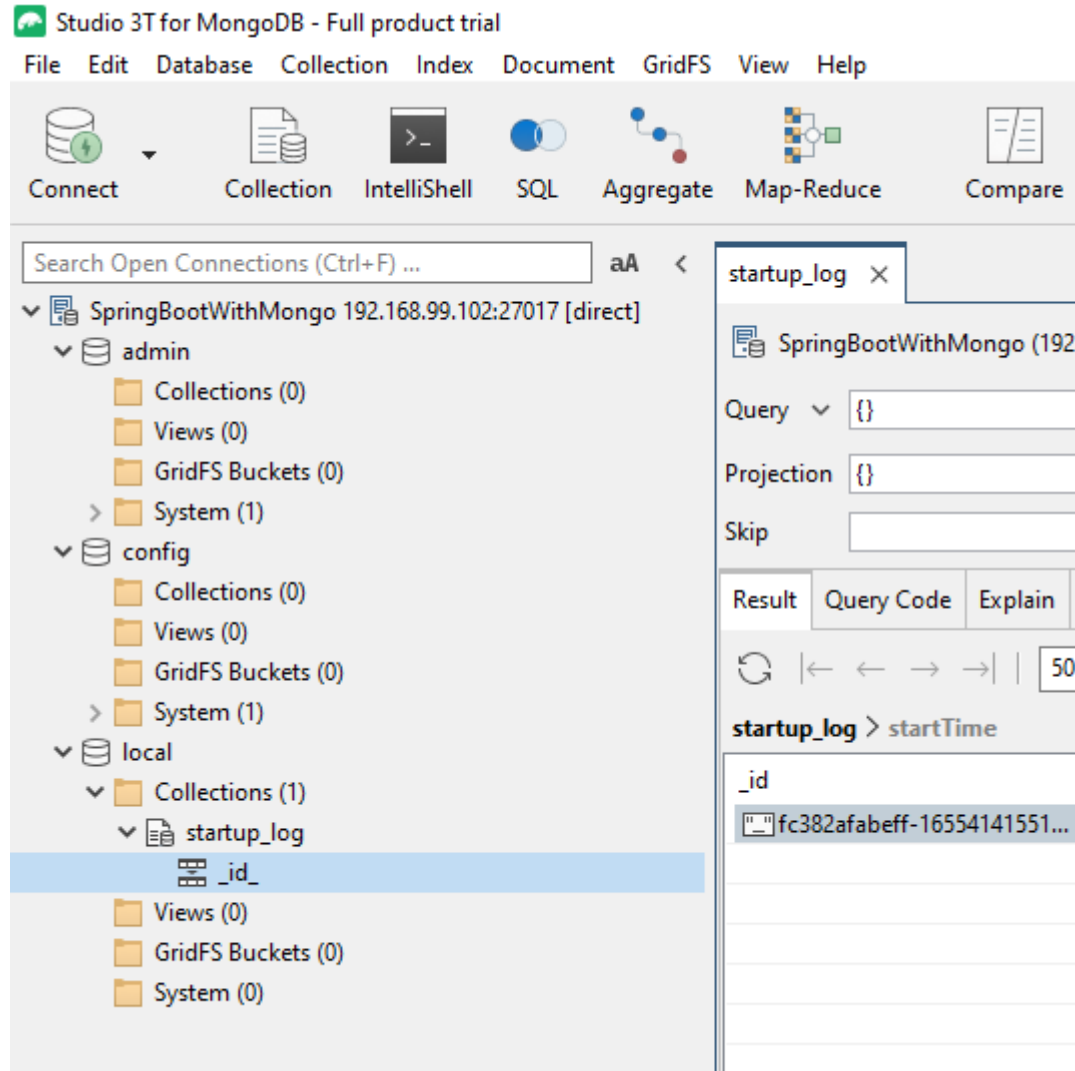
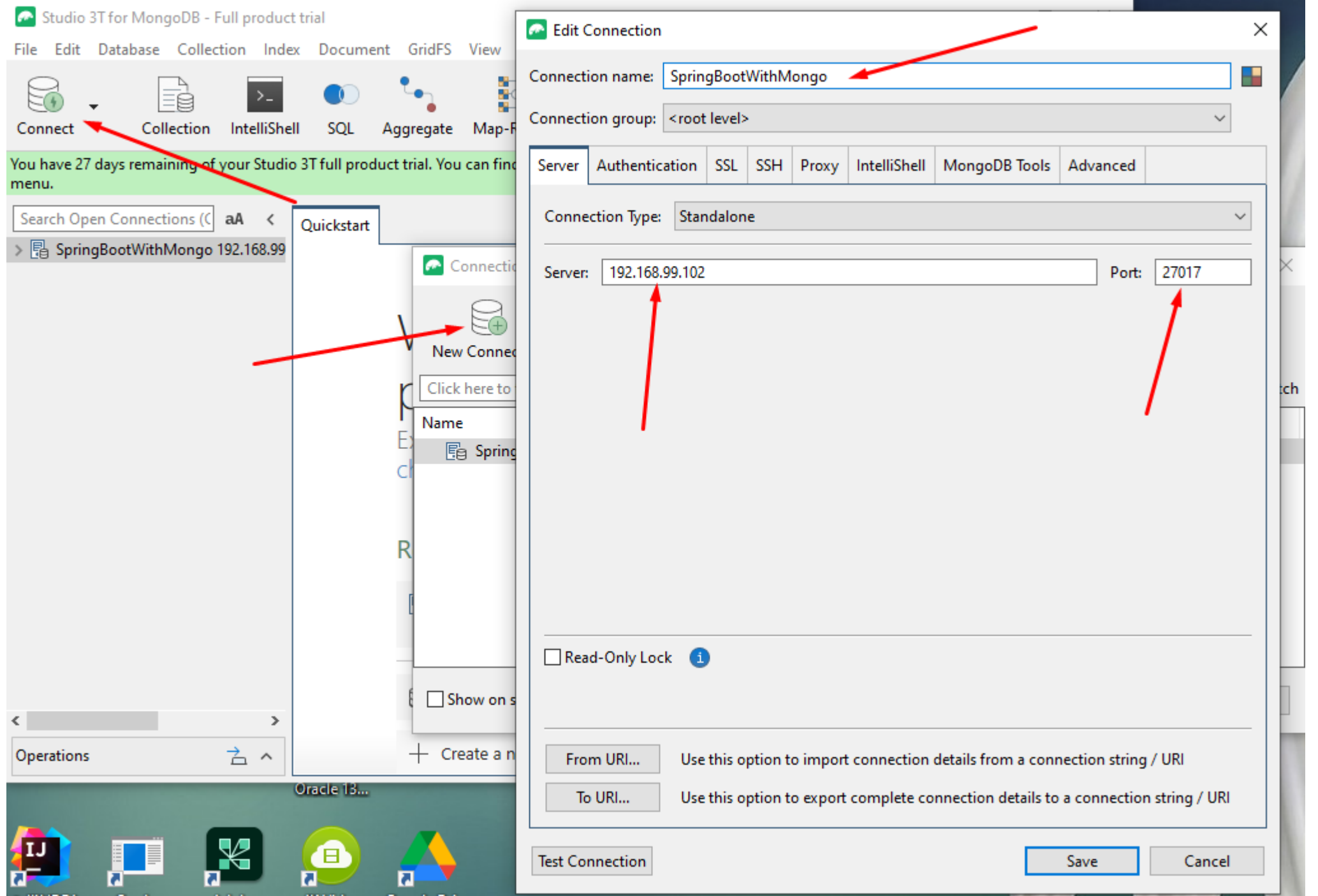


Şu komut ile istersek docker'da çalışan tüm image'ları durdurabiliriz.

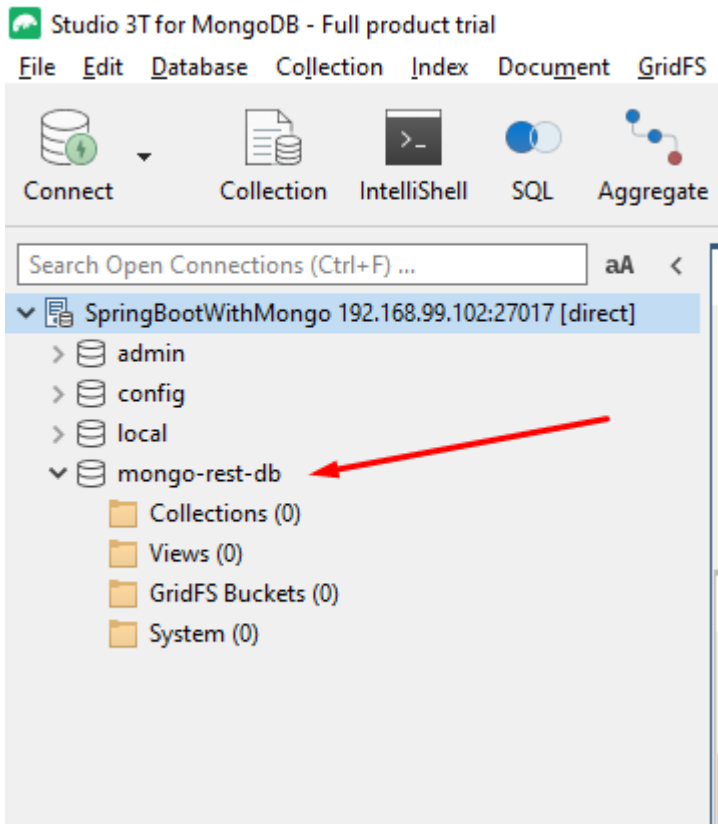
docker stop \$(docker ps -qa)

Mongo db arayüzüne bağlanmak için robomongo'yu kullanabiliriz. Alttaki link üzerinden indirip kurabiliriz.

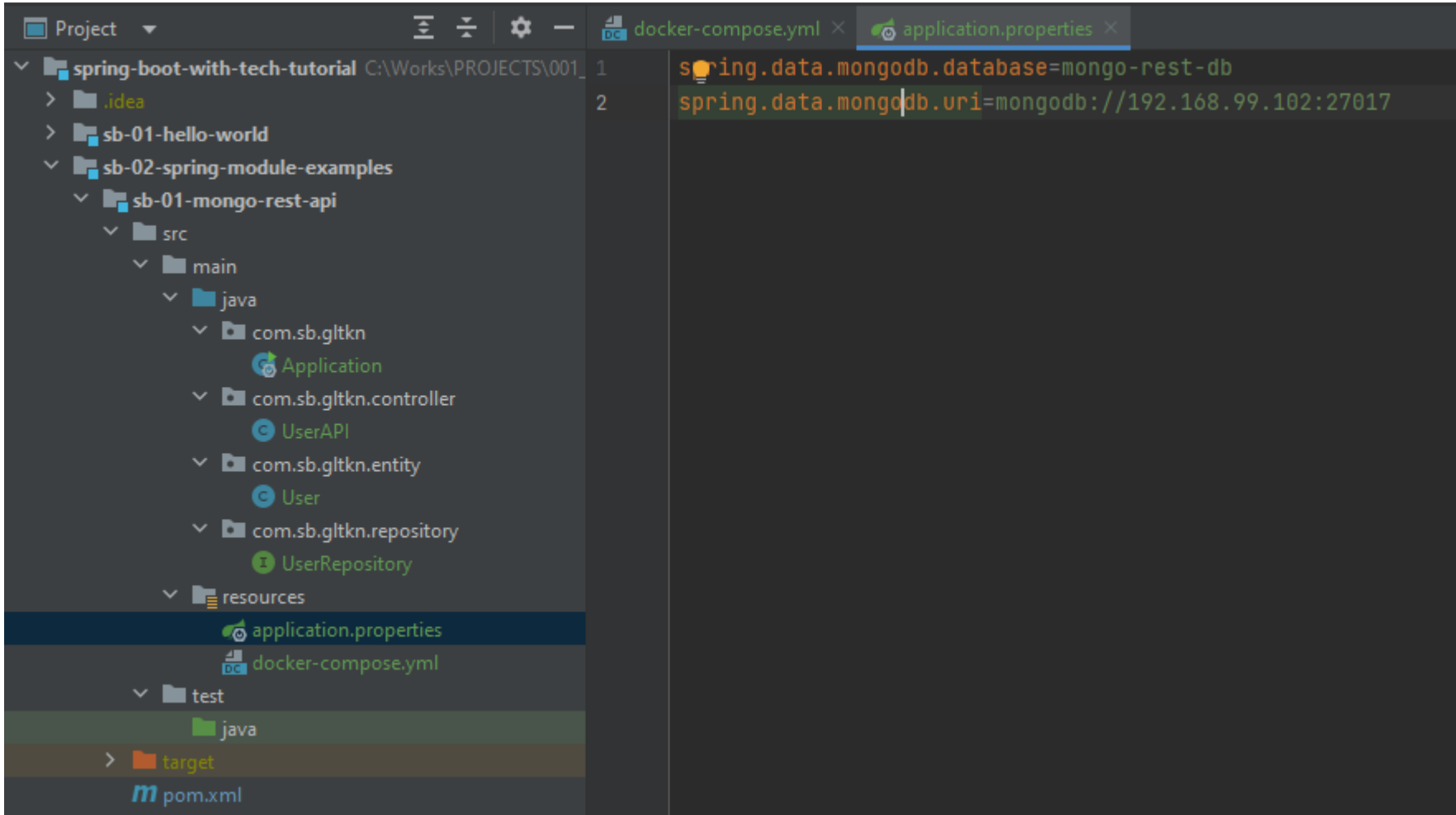
<https://robomongo.org/>



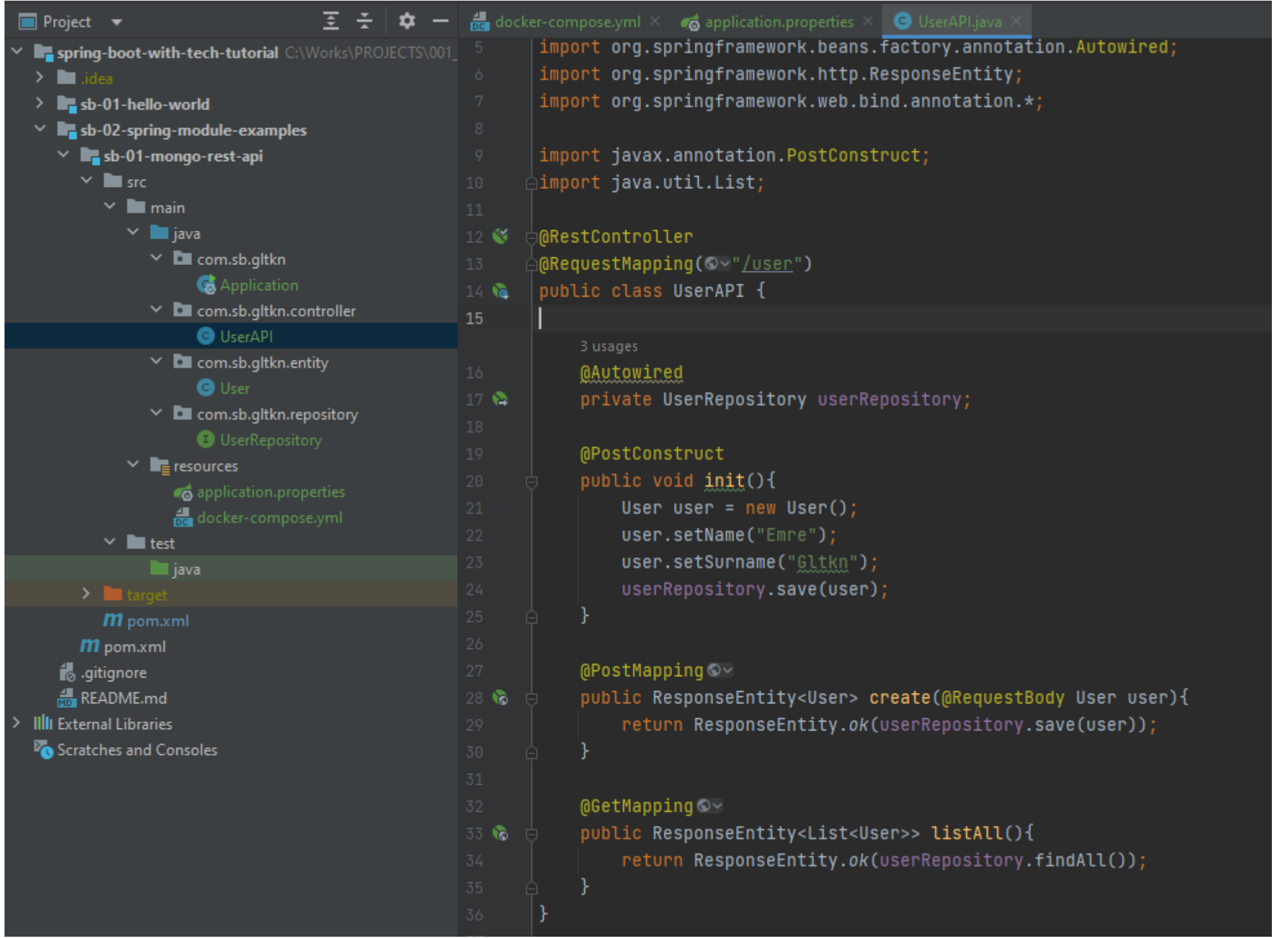
mongo-rest-db ismindeki database'imizi oluşturduk.



Spring boot ile **sb-01-mongo-rest-api** uygulamamızı da yazdık.



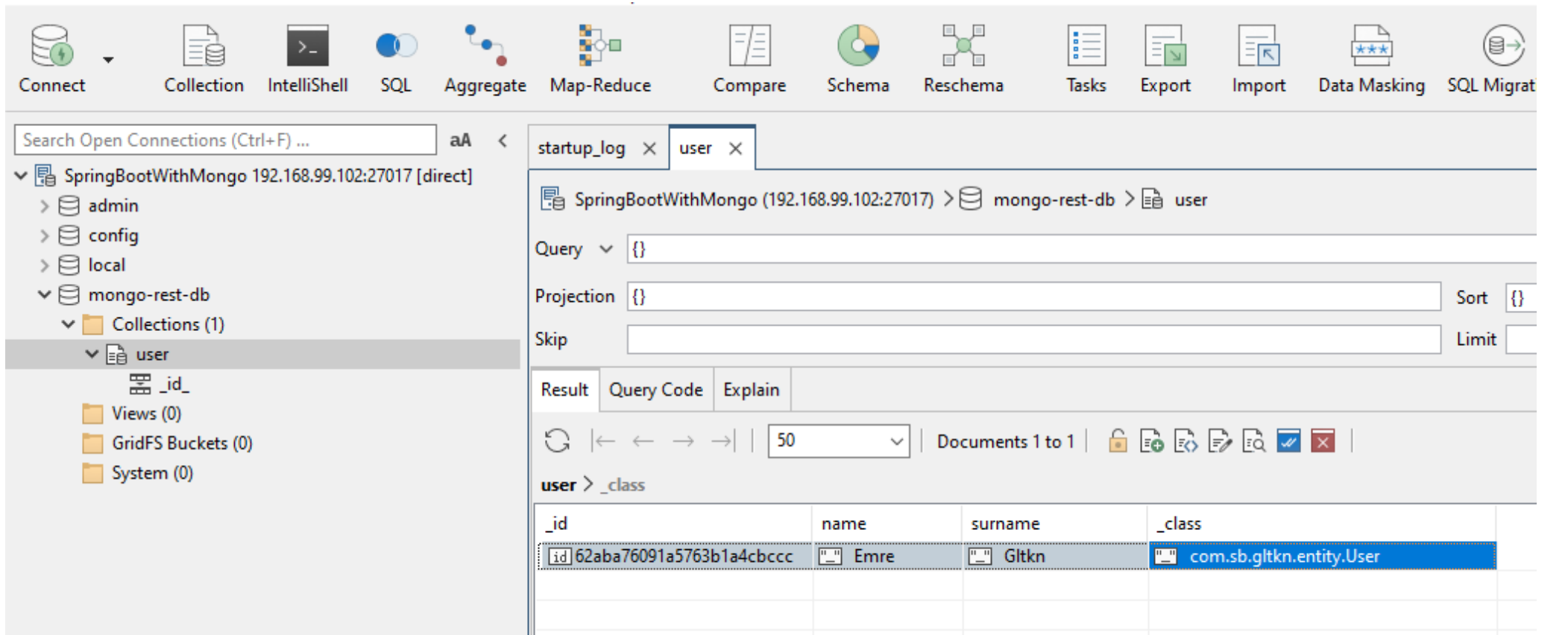
@PostConstruct ile ilk verinin eklenmesini sağladık.



The screenshot shows the IntelliJ IDEA IDE. On the left, the Project Explorer displays the project structure for 'spring-boot-with-tech-tutorial'. The 'src/main/java/com.sb.gltkn.controller' package contains the 'UserAPI' class. The 'resources' folder contains 'application.properties' and 'docker-compose.yml'. The 'test' folder contains a 'java' sub-folder. The 'target' folder contains 'pom.xml' files. The 'External Libraries' and 'Scratches and Consoles' are also visible.

The main editor shows the 'UserAPI.java' file. The code is as follows:

```
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.http.ResponseEntity;
7 import org.springframework.web.bind.annotation.*;
8
9 import javax.annotation.PostConstruct;
10 import java.util.List;
11
12 @RestController
13 @RequestMapping("/user")
14 public class UserAPI {
15
16     3 usages
17     @Autowired
18     private UserRepository userRepository;
19
20     @PostConstruct
21     public void init(){
22         User user = new User();
23         user.setName("Emre");
24         user.setSurname("Gltkn");
25         userRepository.save(user);
26     }
27
28     @PostMapping
29     public ResponseEntity<User> create(@RequestBody User user){
30         return ResponseEntity.ok(userRepository.save(user));
31     }
32
33     @GetMapping
34     public ResponseEntity<List<User>> listAll(){
35         return ResponseEntity.ok(userRepository.findAll());
36     }
37 }
```



The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure, including the 'mongo-rest-db' database and the 'user' collection. The main panel shows the 'user' collection with a single document.

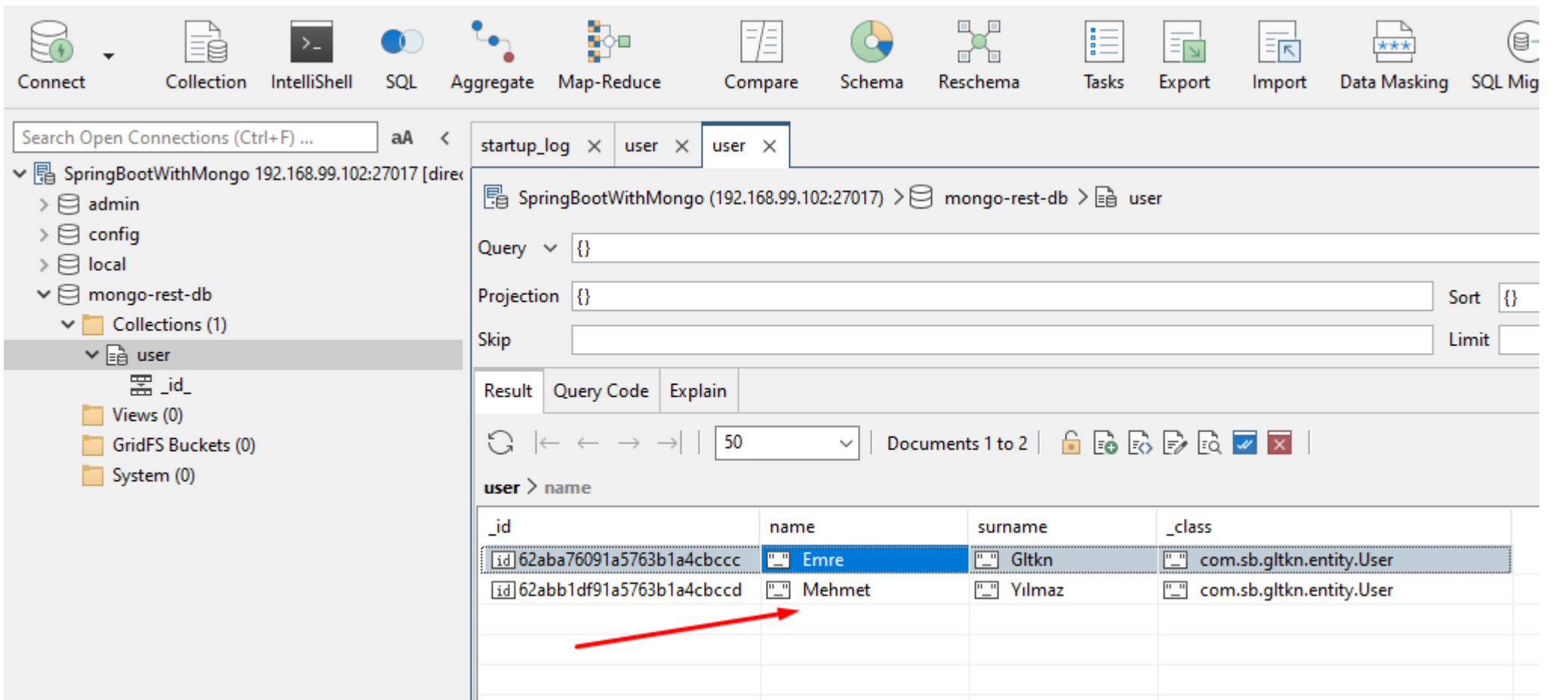
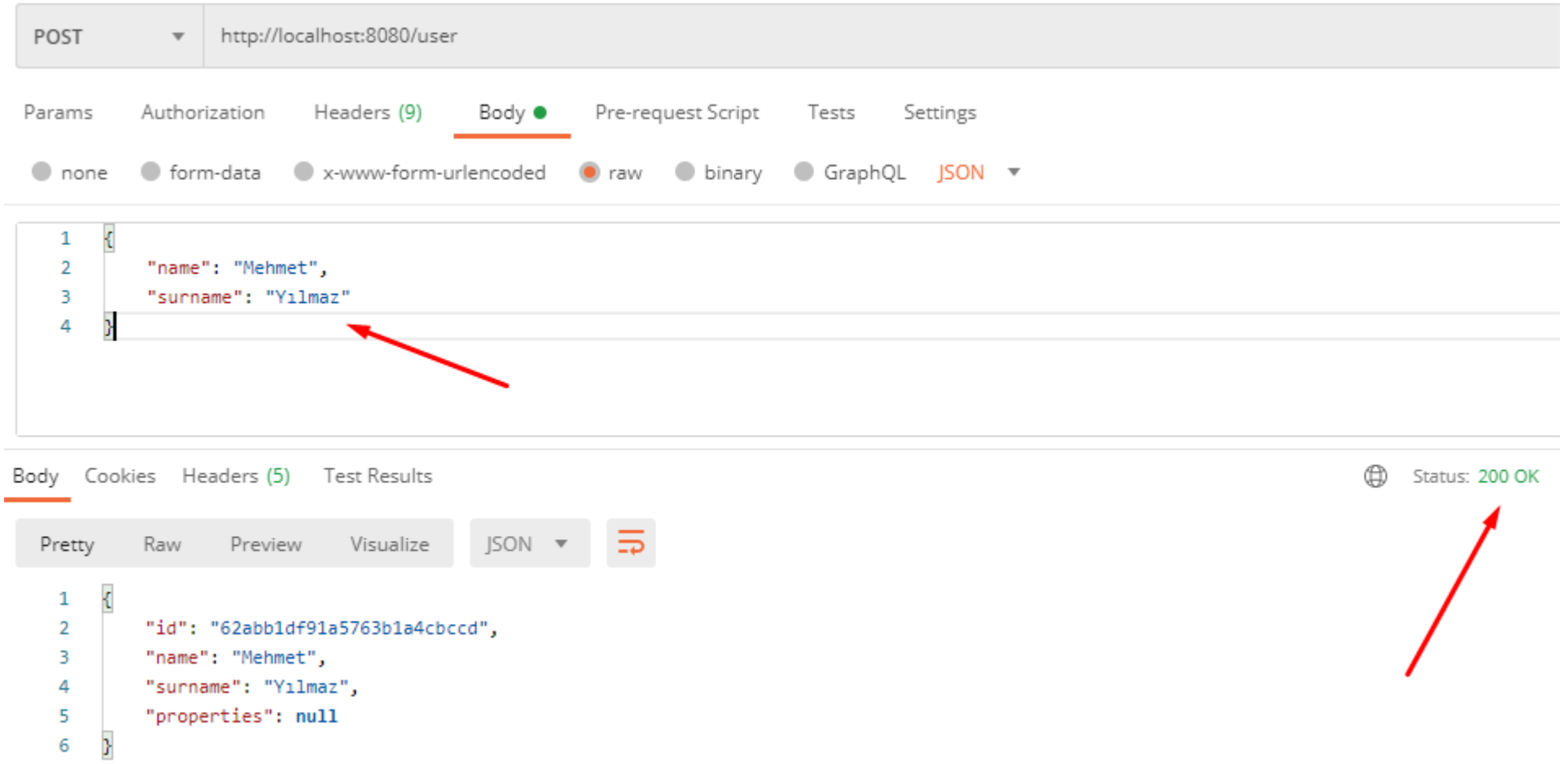
The document details are as follows:

_id	name	surname	_class
62aba76091a5763b1a4cbccc	Emre	Gltkn	com.sb.gltkn.entity.User

<http://localhost:8080/user>

```
1 // 20220617011753
2 // http://localhost:8080/user
3
4 [
5   {
6     "id": "62aba76091a5763b1a4cbccc",
7     "name": "Emre",
8     "surname": "Gltkn",
9     "properties": null
10  }
11 ]
```

Postman ile yeni bir data kayıt işlemi gerçekleştirelim.



Şimdi ise container'ımızı stop edelim.

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
fc382afabeff   mongo:4.4.6    "docker-entrypoint.s..." About an hour ago Up 3 minutes    0.0.0.0:27017->27017/tcp resources_mongo_1

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker stop fc382afabeff
fc382afabeff

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker ps
```

Tekrar container'ımızı başlatmak istersek alttaki komutu kullanabiliriz.

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-compose -f /c/Works/PROJECTS/001_Workspace/intellijidea/SB_004_SpringBootTutorial/repository/spring-boot-with-tech-tutorial/sb-02-spring-module
-examples/sb-01-mongo-rest-api/src/main/resources/docker-compose.yml up -d
```