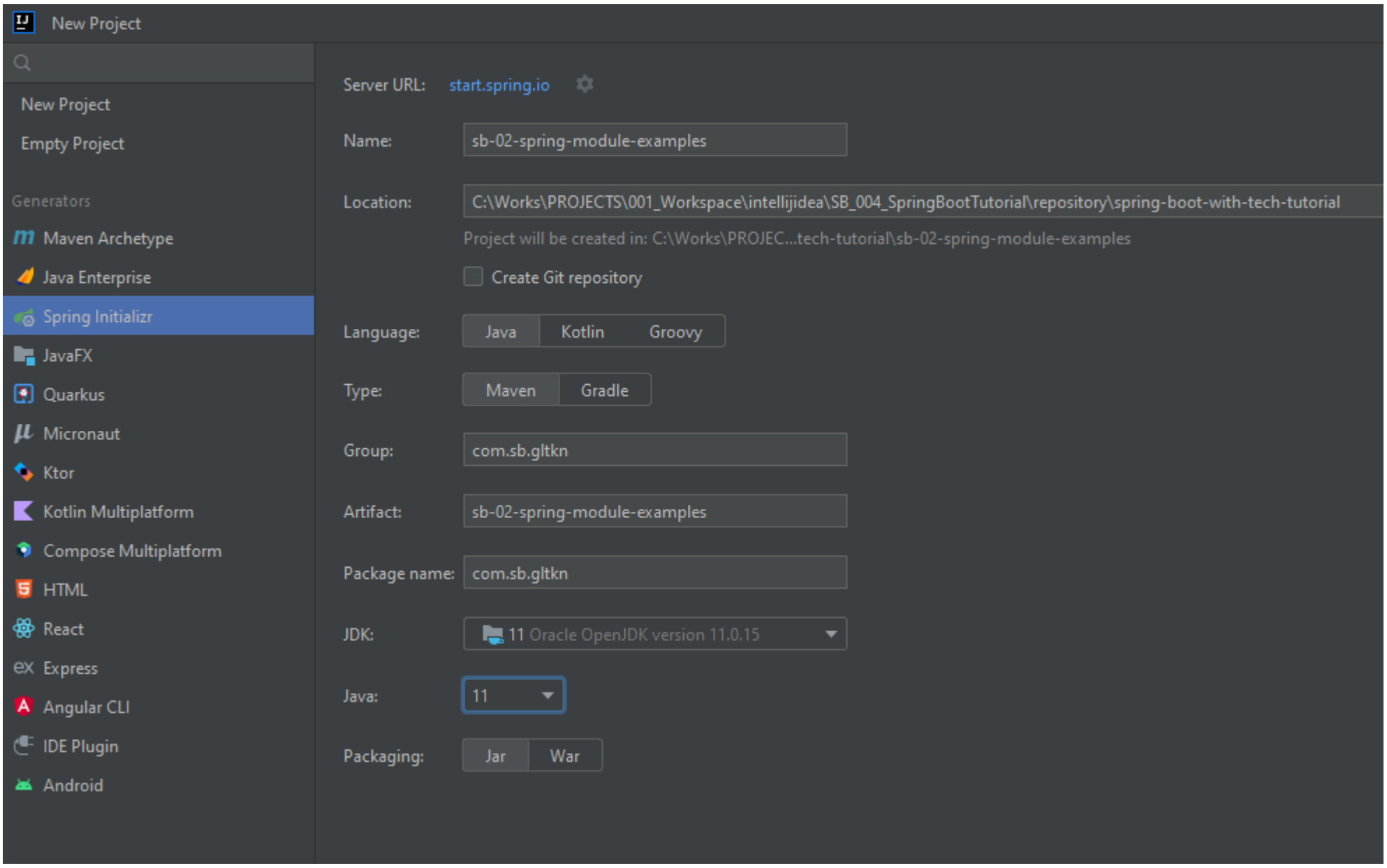
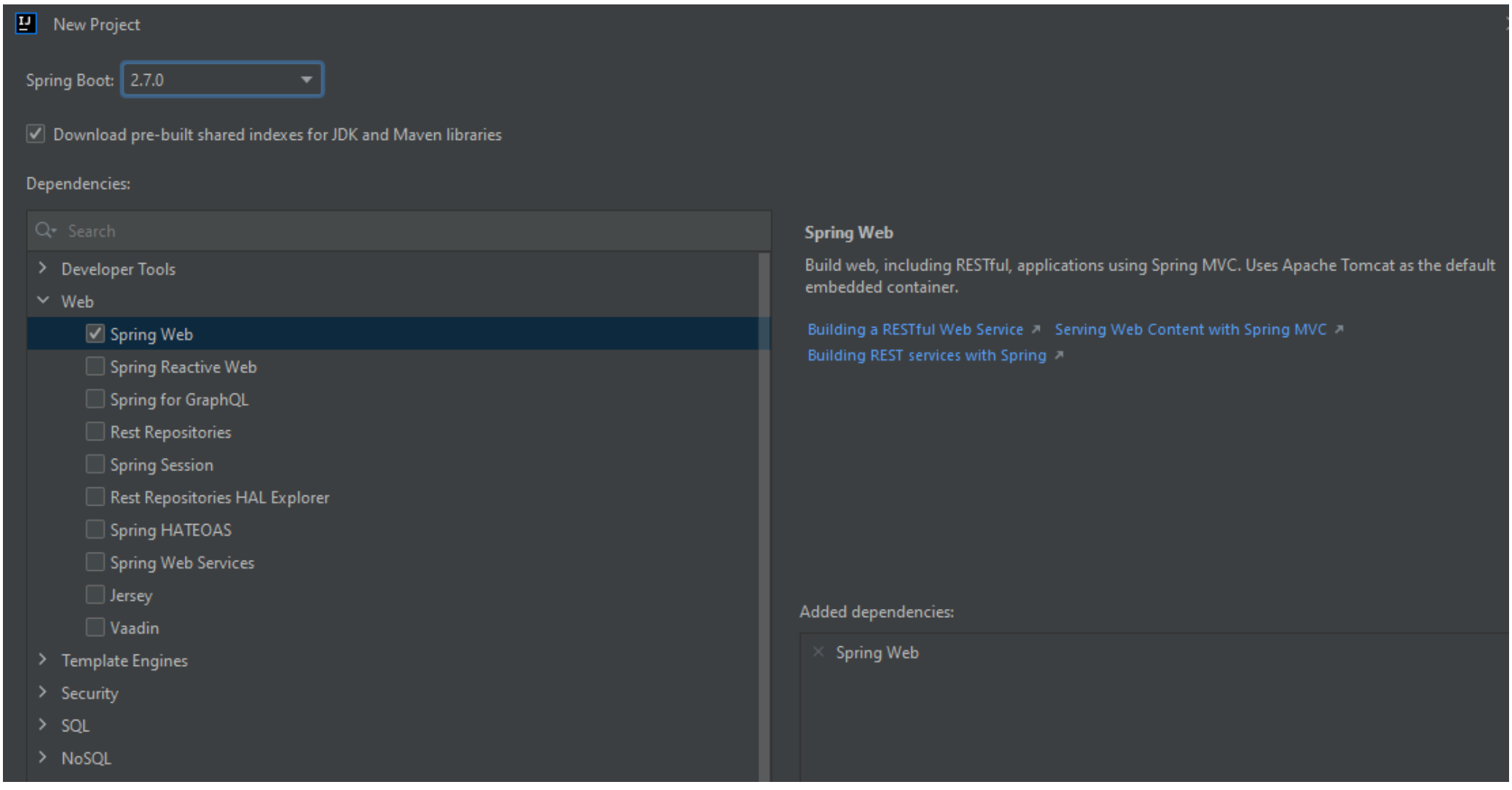
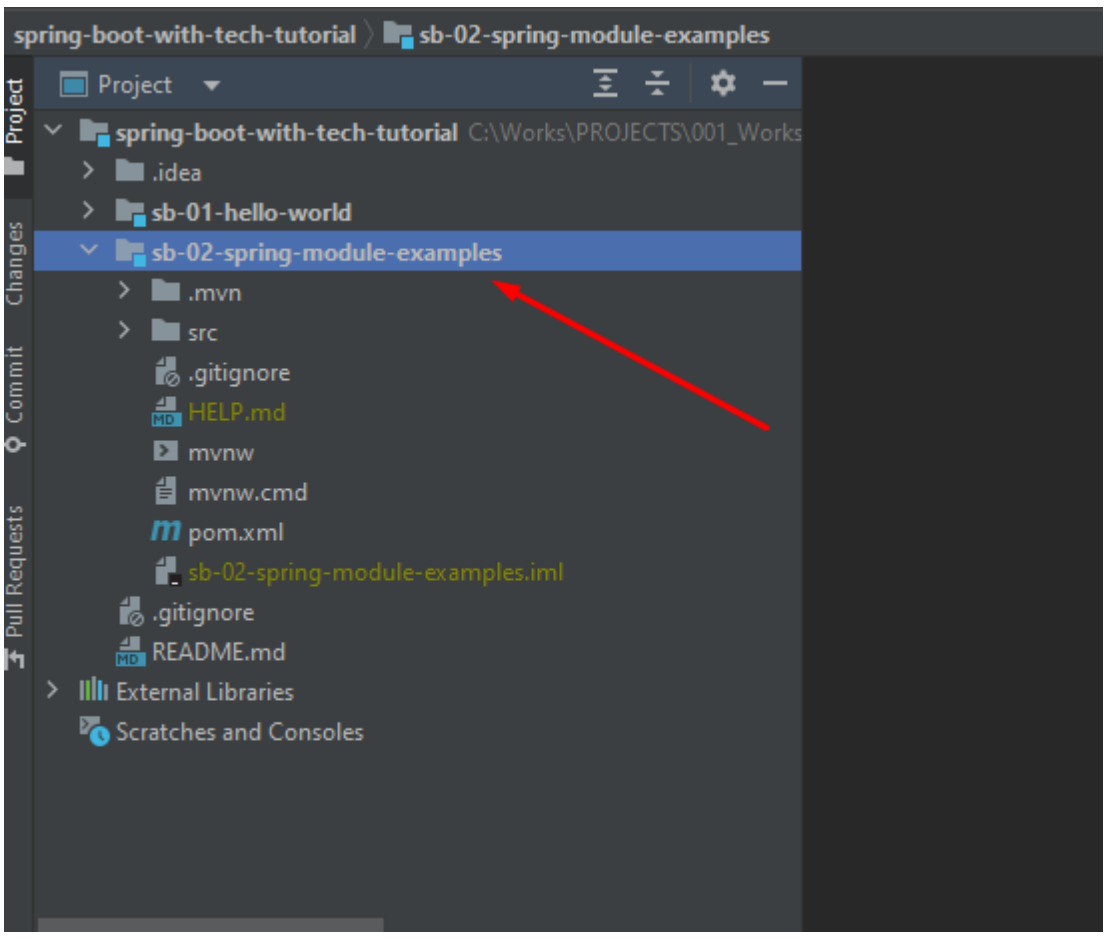
Projemizi aşağıdaki gibi oluşturalım.

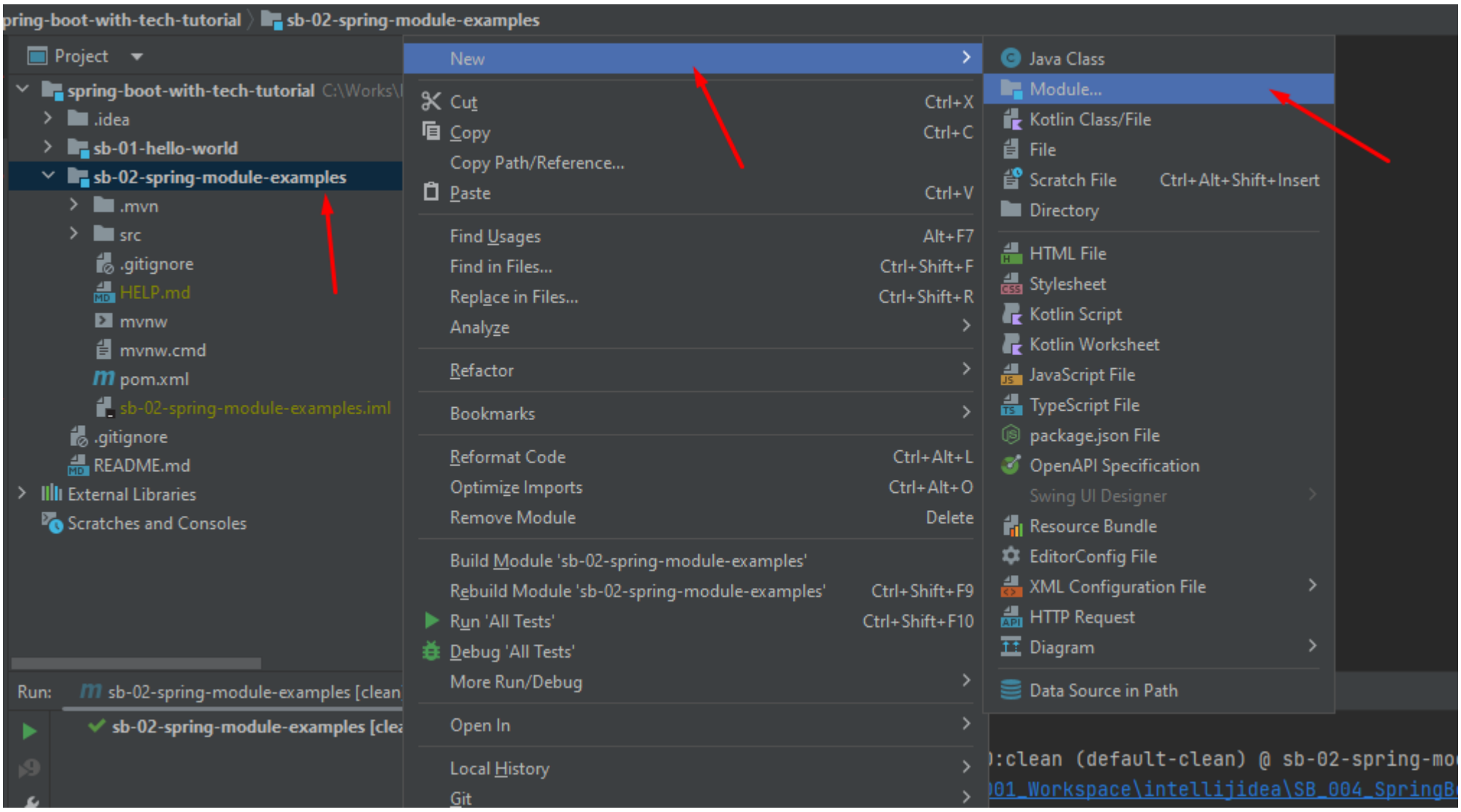


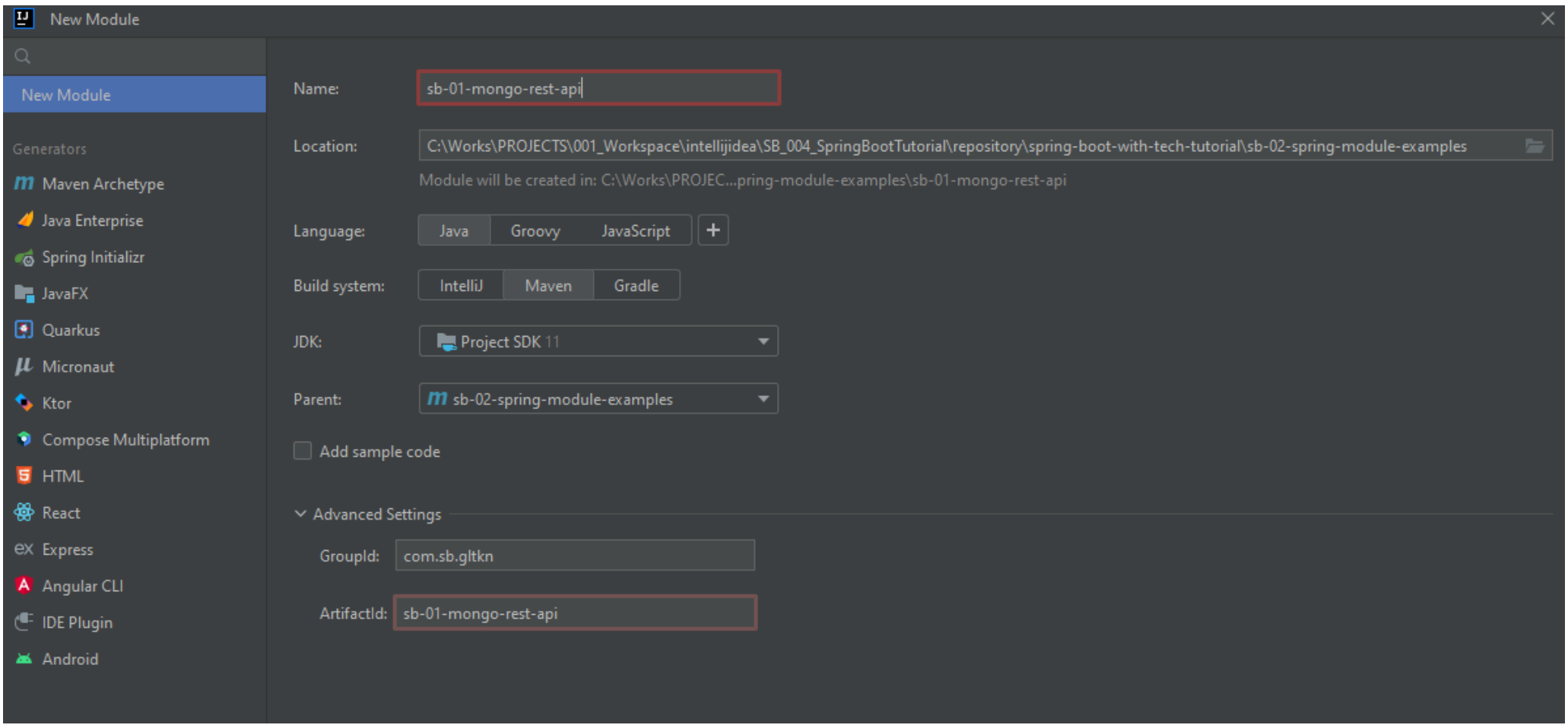


Alttaki proje içerisinden **src** klasörü, **.gitignore** ve **HELP.md** dosyalarını silebiliriz. Çünkü bu proje altına birazdan modüller oluşturacağız.



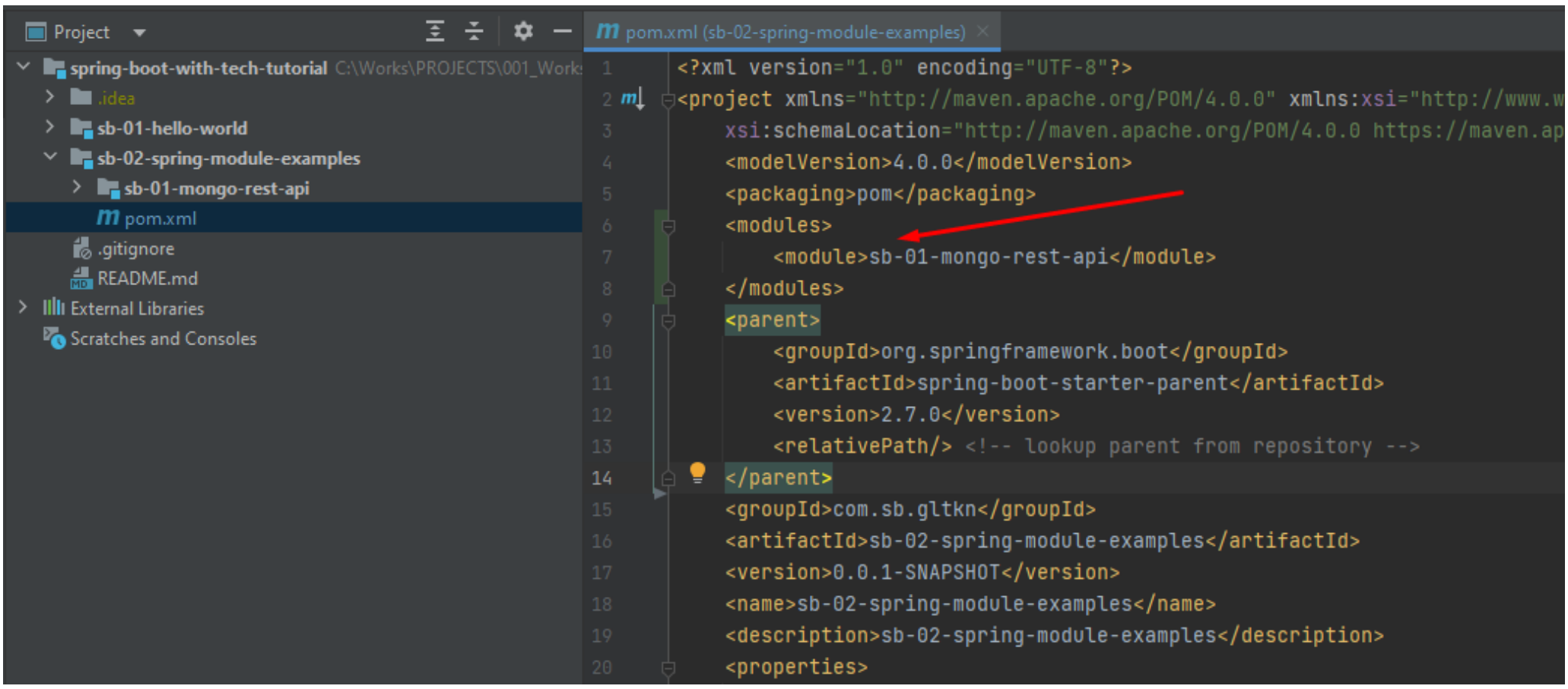
Projemize modül ekleyerek ilerleyeceğiz.



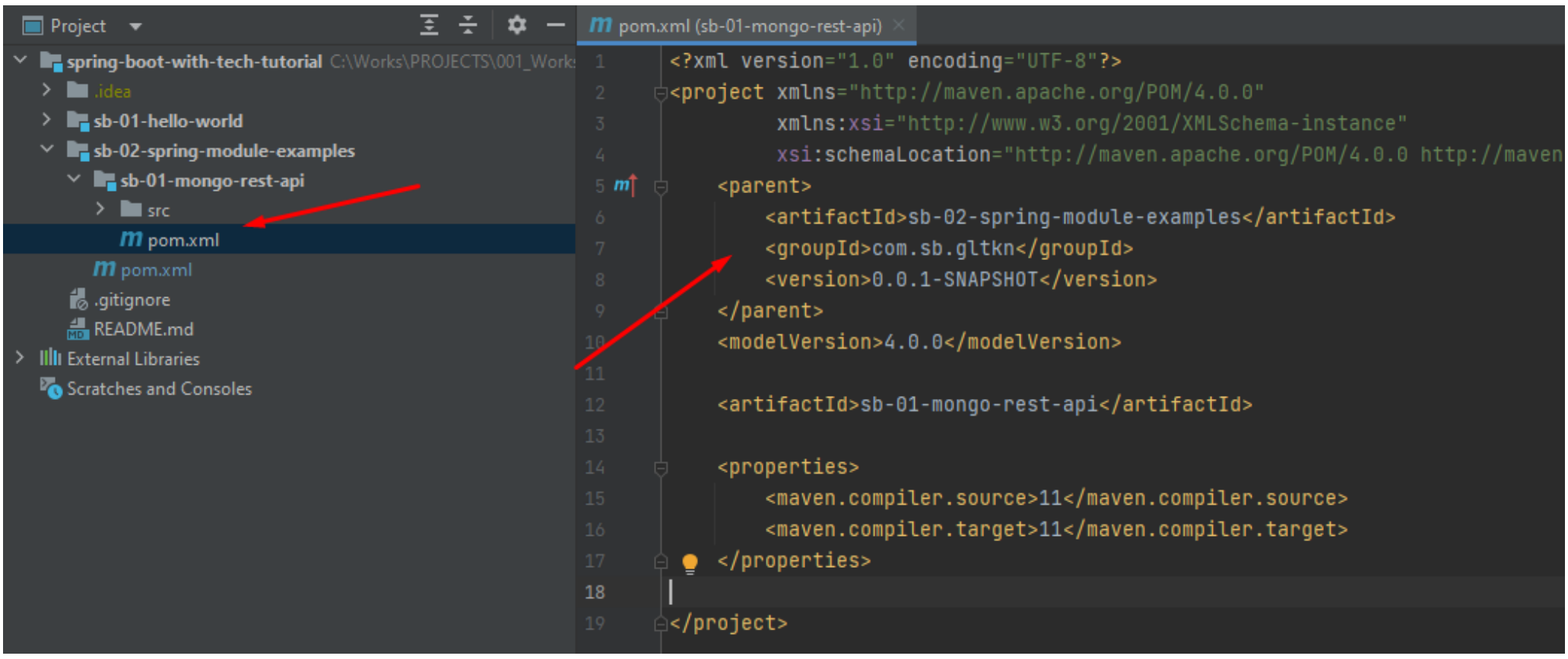


Parent projenin src dosyasını ve mvn dosyaları sildik ve pom.xml dosyasını da aşağıdaki gibi yaptık.

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                 Unknown   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\choco
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)"

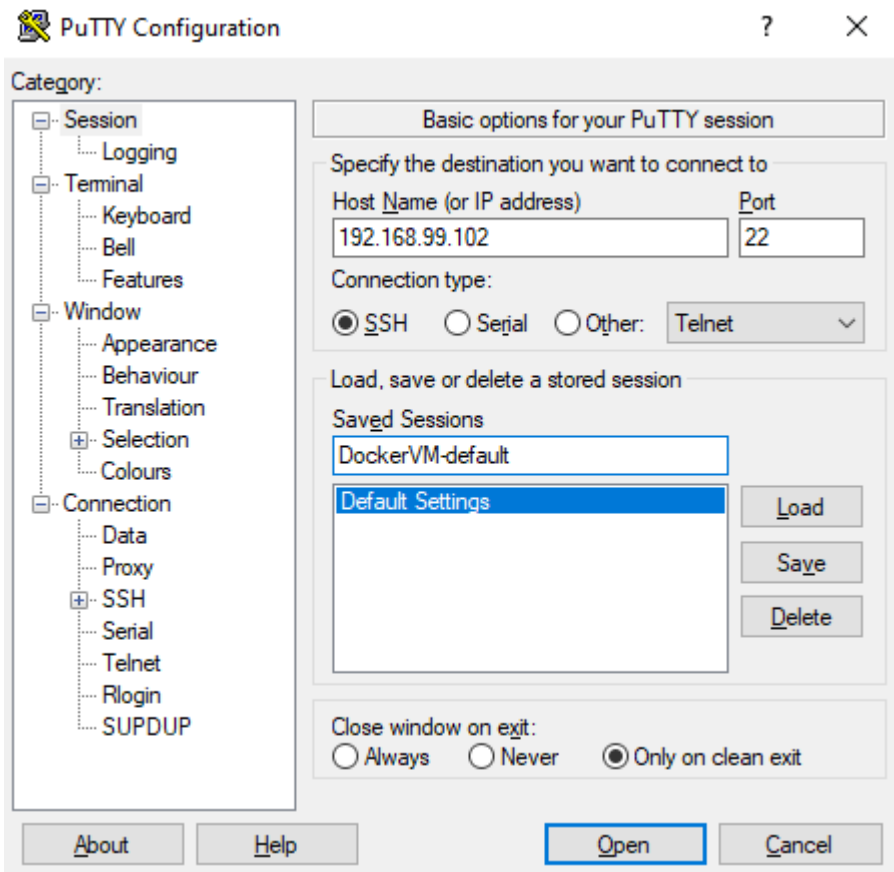
docker-machine start default

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
```

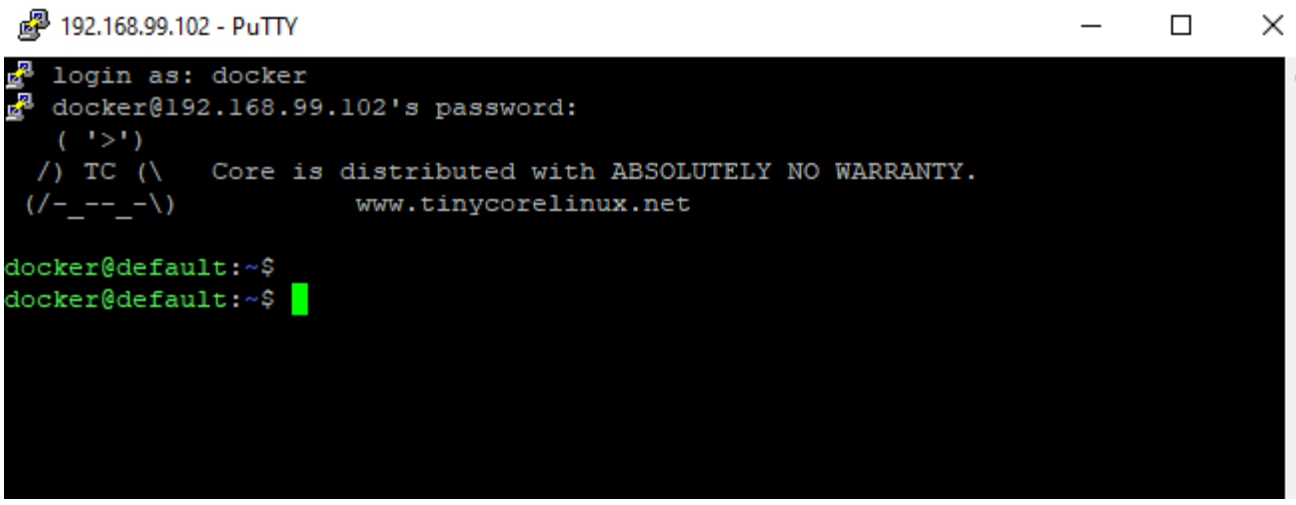
Docker Machine IP: 192.168.99.102

Şimdi docker makinemize putty üzerinden bağlanalım.



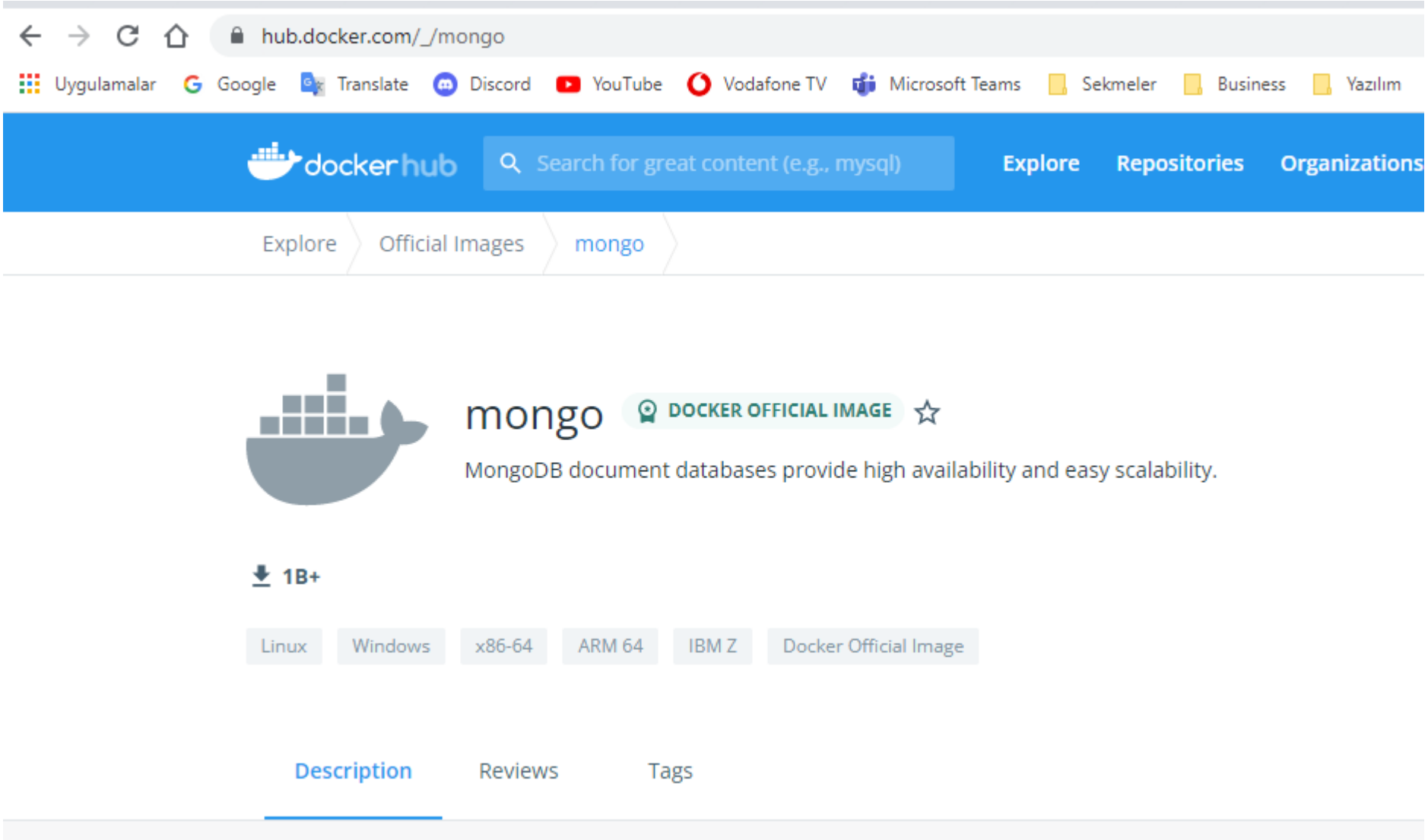
Default values;

Username:	docker
Password:	tcuser



Mongo db kurulumunu gerçekleştirelim.

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



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

tag yerine versiyon bilgisi yazılır. Son versiyonu almak için aşağıdaki gibi latest yazılarak çalıştırılır.

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
$
```

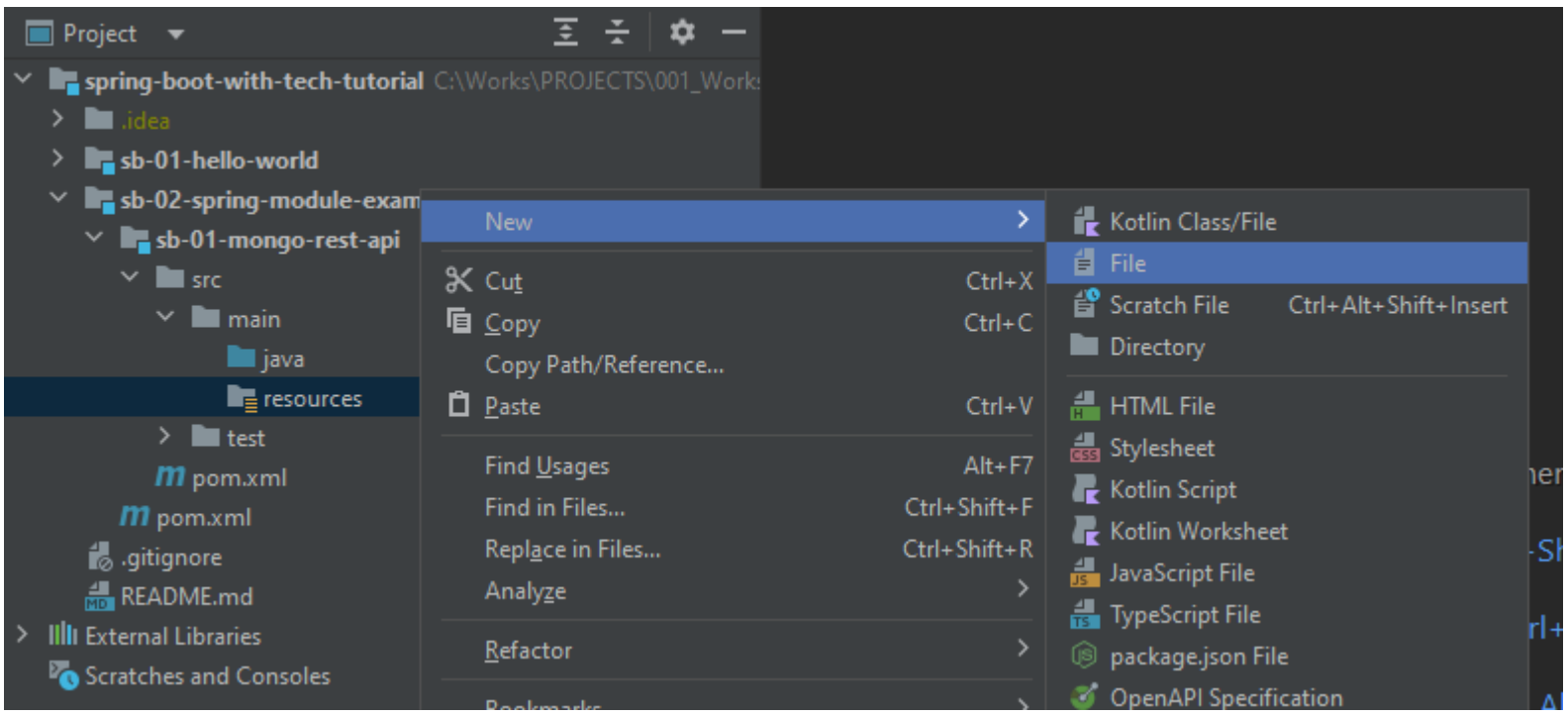
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 /c/Program Files/Docker Toolbox
$

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

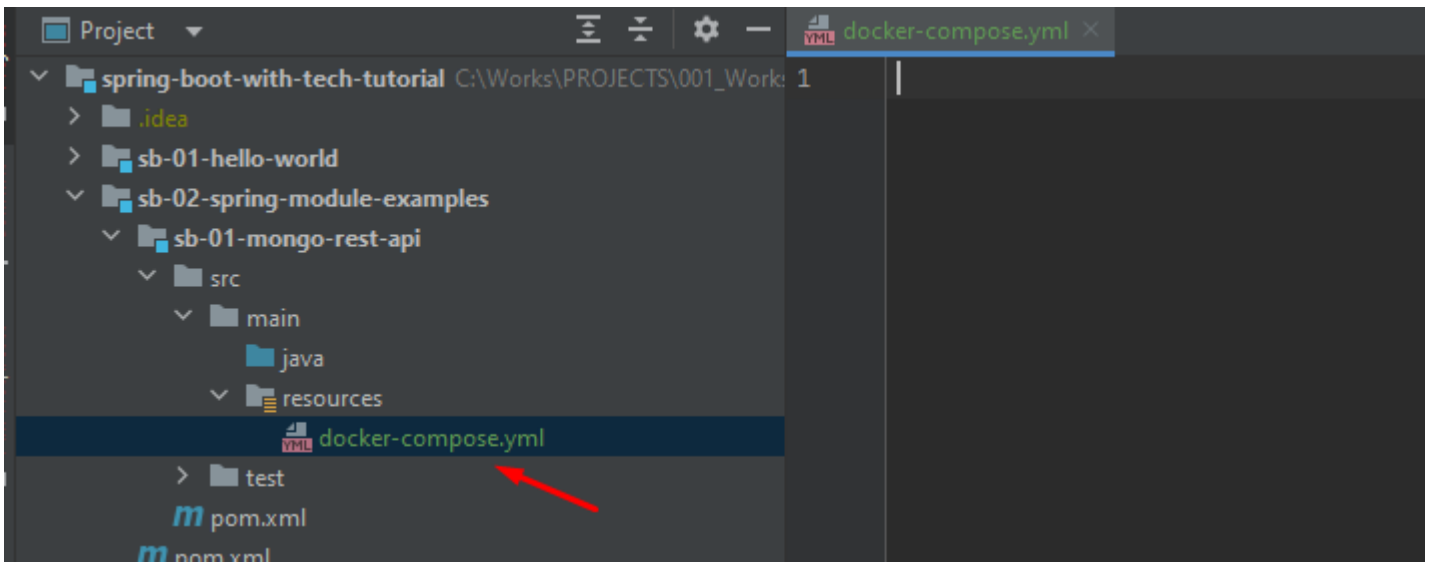
Şimdi git bash ile docker image'larına bakalım.

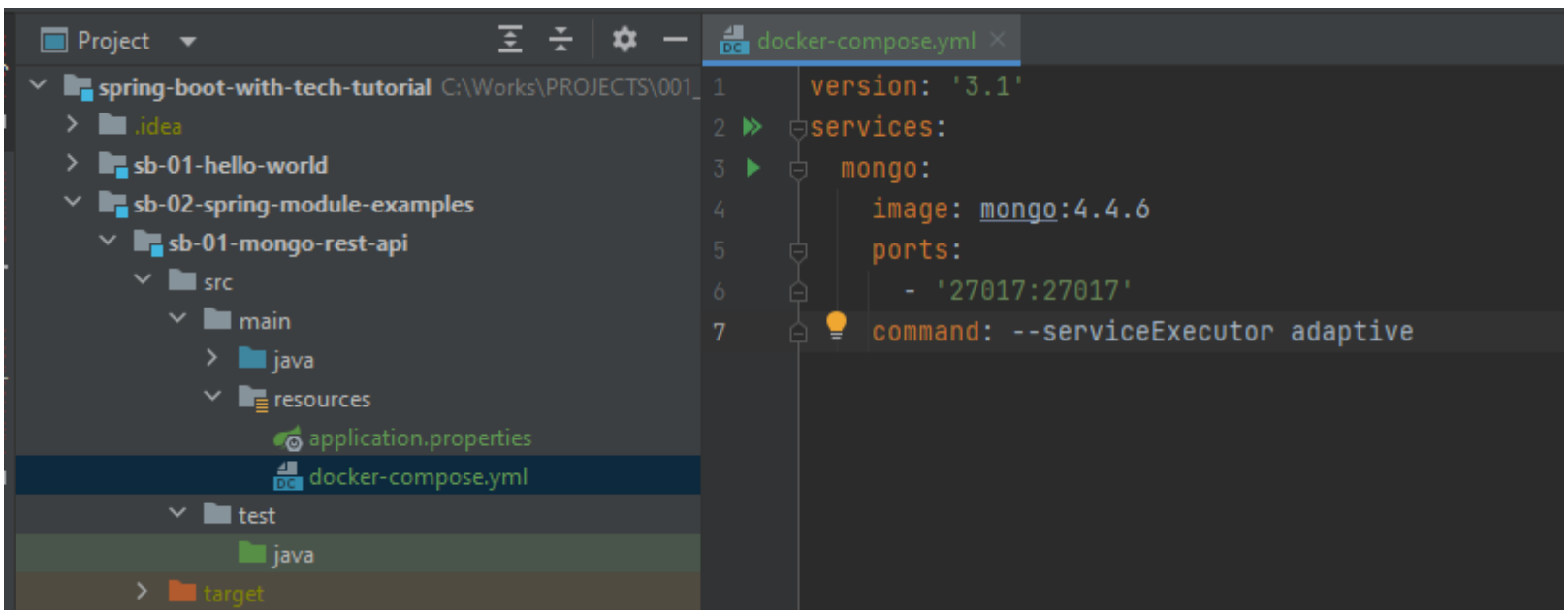
docker image ls

```
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

```
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
```

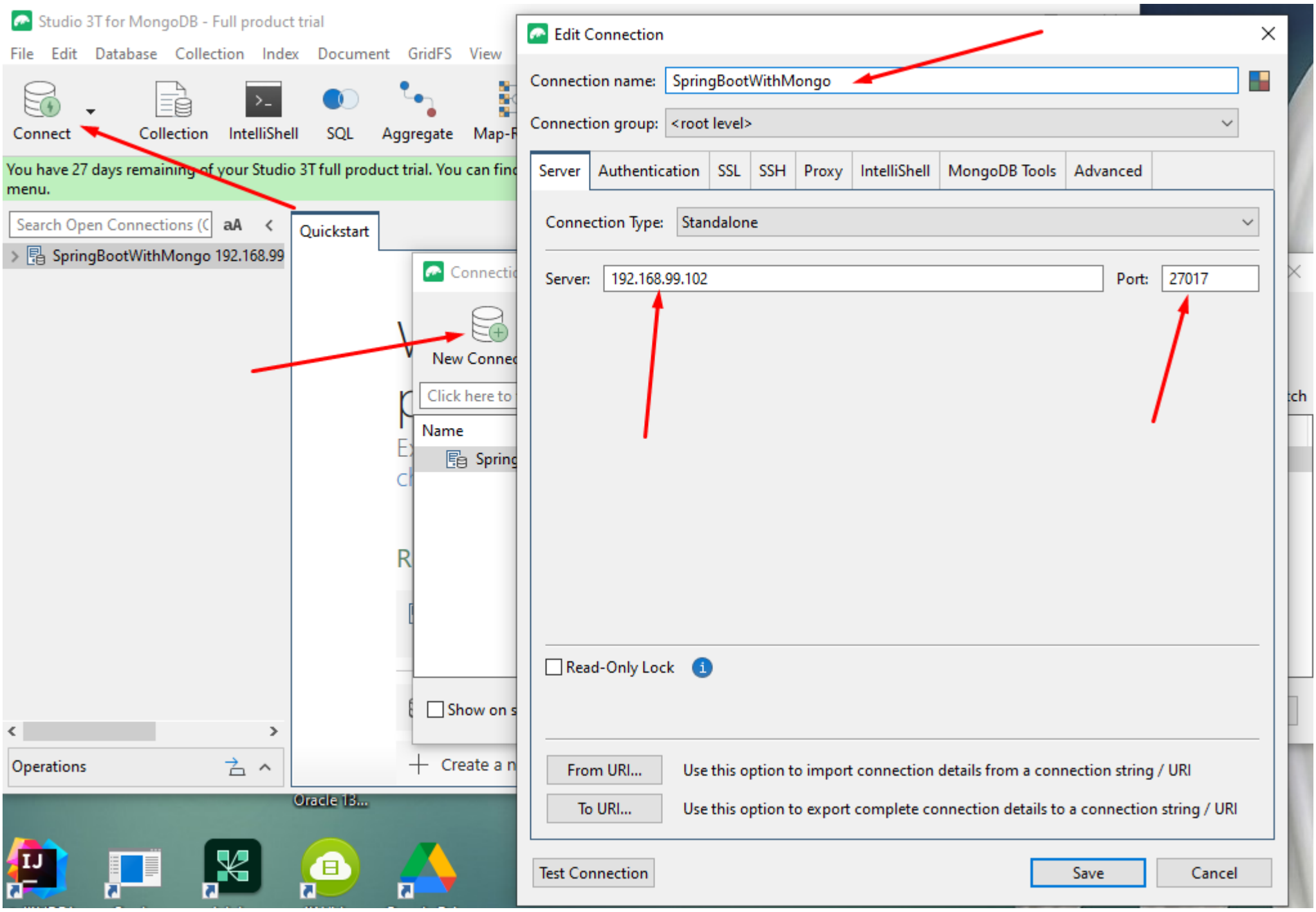
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS          PORTS                      NAMES
fc382afabeff   mongo:4.4.6   "docker-entrypoint.s…"   30 minutes ago   Up 30 minutes   0.0.0.0:27017->27017/tcp   resources_mongo_1
```

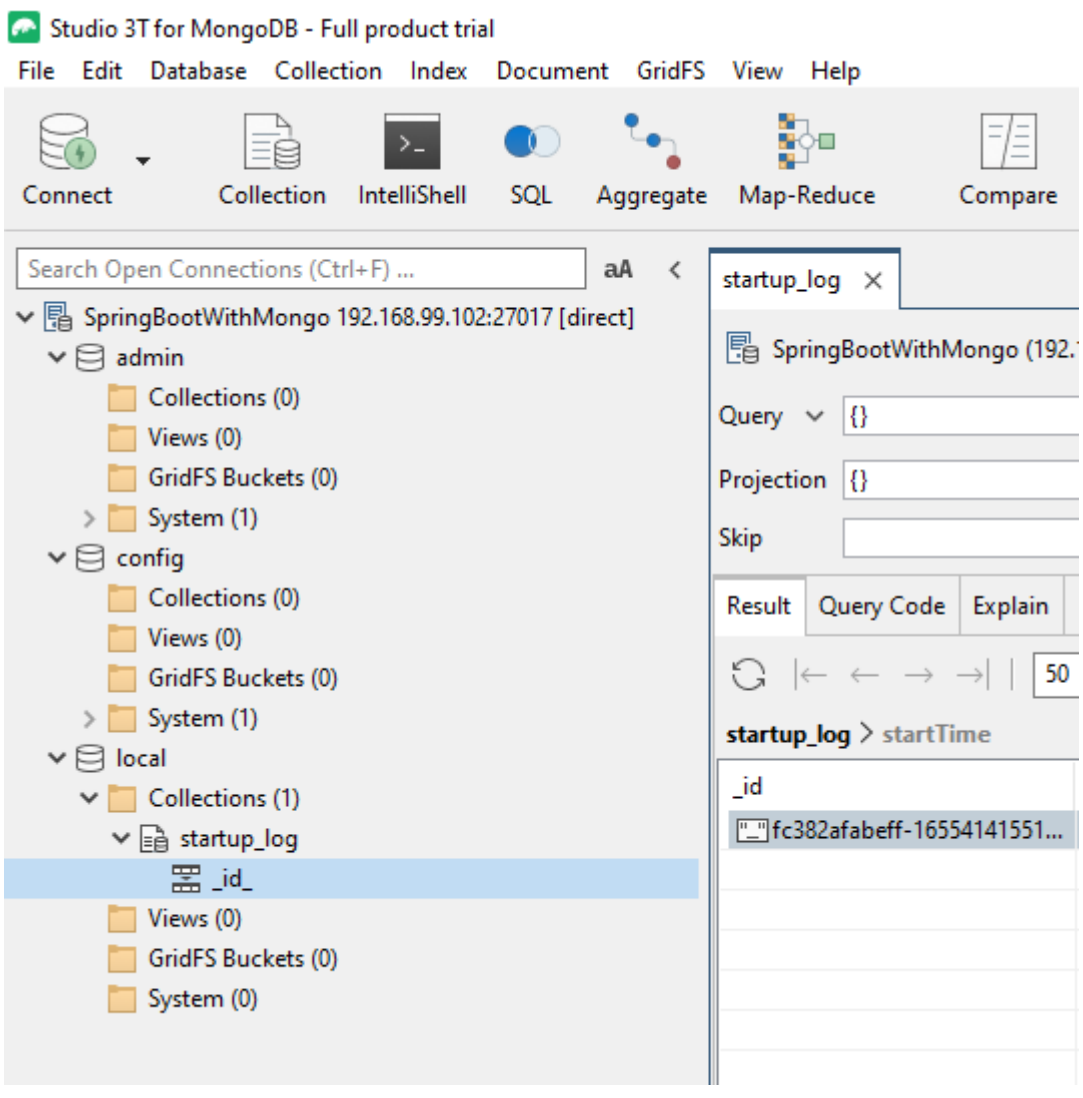
Şu komut ile istersek docker'da çalışan tüm image'ları durdurabilriz.

docker stop $(docker ps -qa)

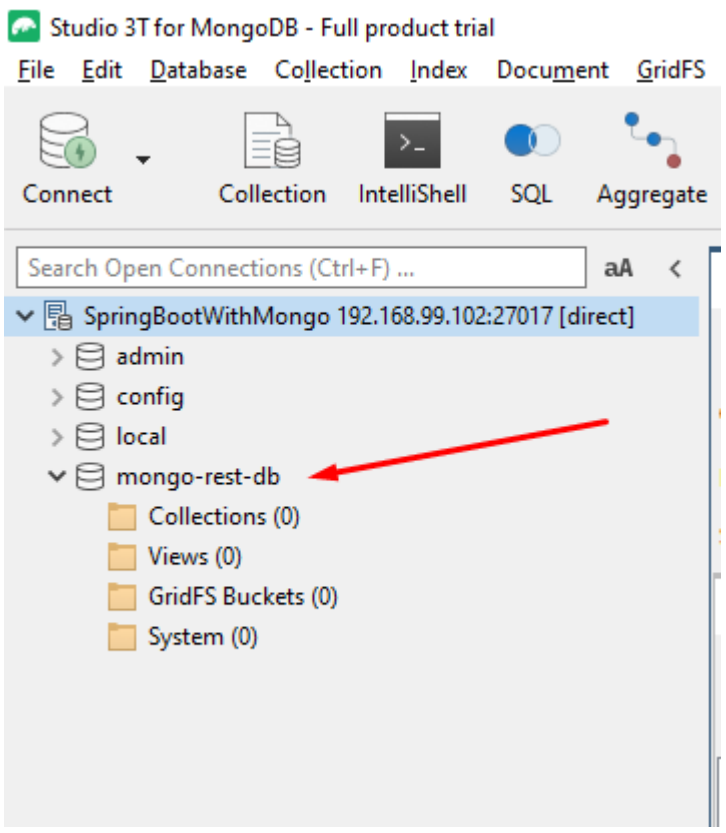
Mongo db arayüzüne bağlanmak için robomongo'yu kullanabiliriz.
Alttaki link üzerinden indirip kuruabilriz.

https://robomongo.org/

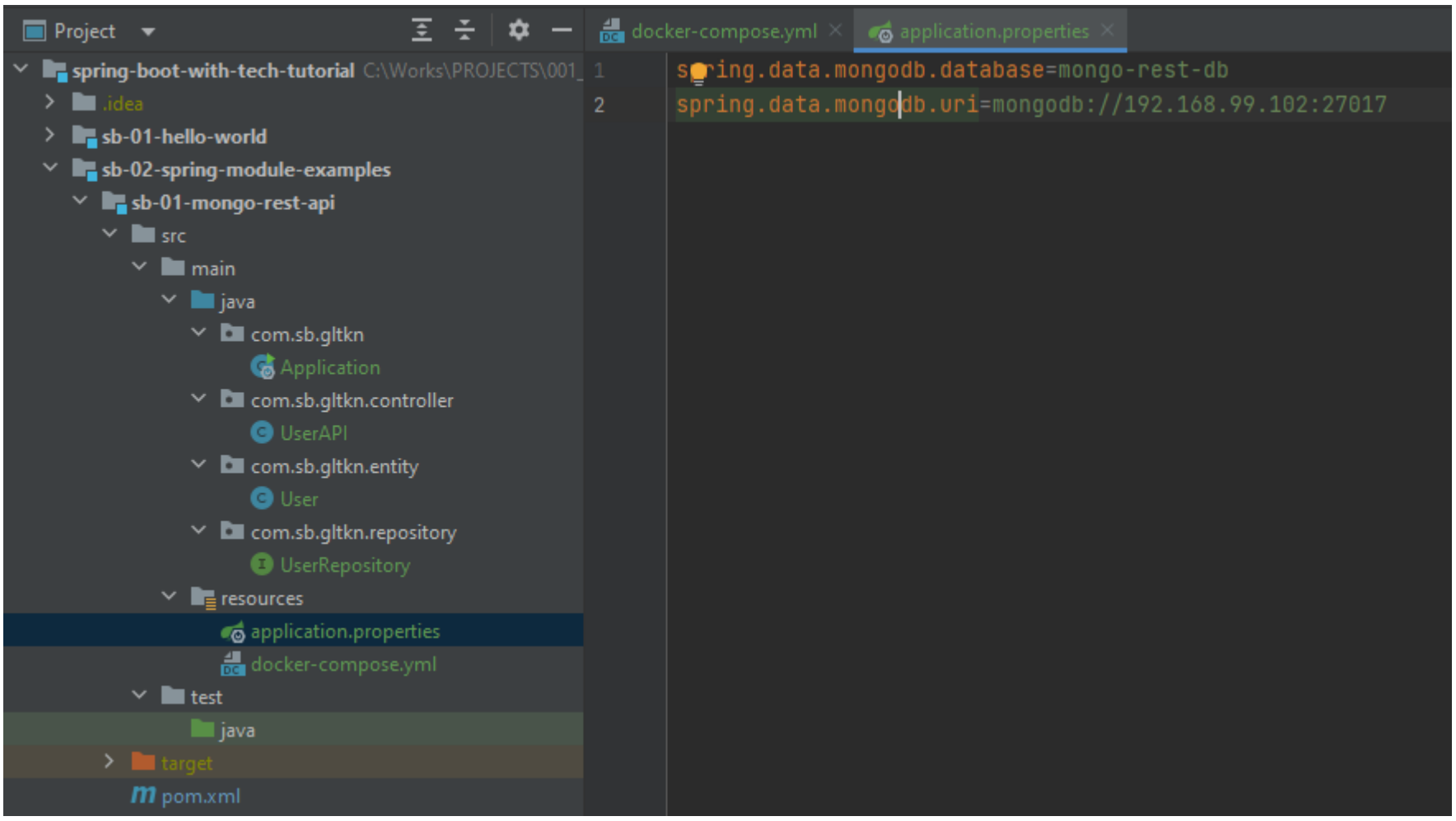




mongo-rest-db ismindeki database'imizi oluşturduk.

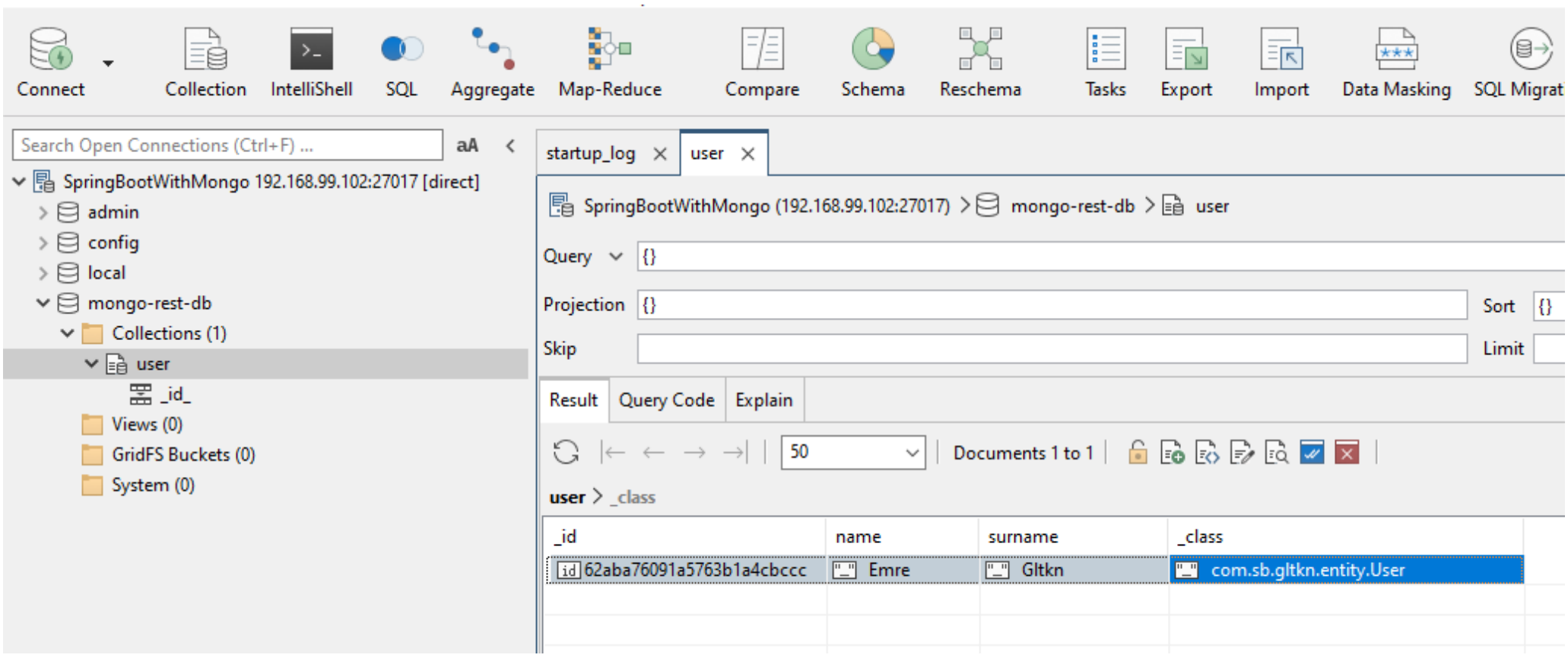


Spring boot ile **sb-01-mongo-rest-api** uygulamamızı da yazdık.



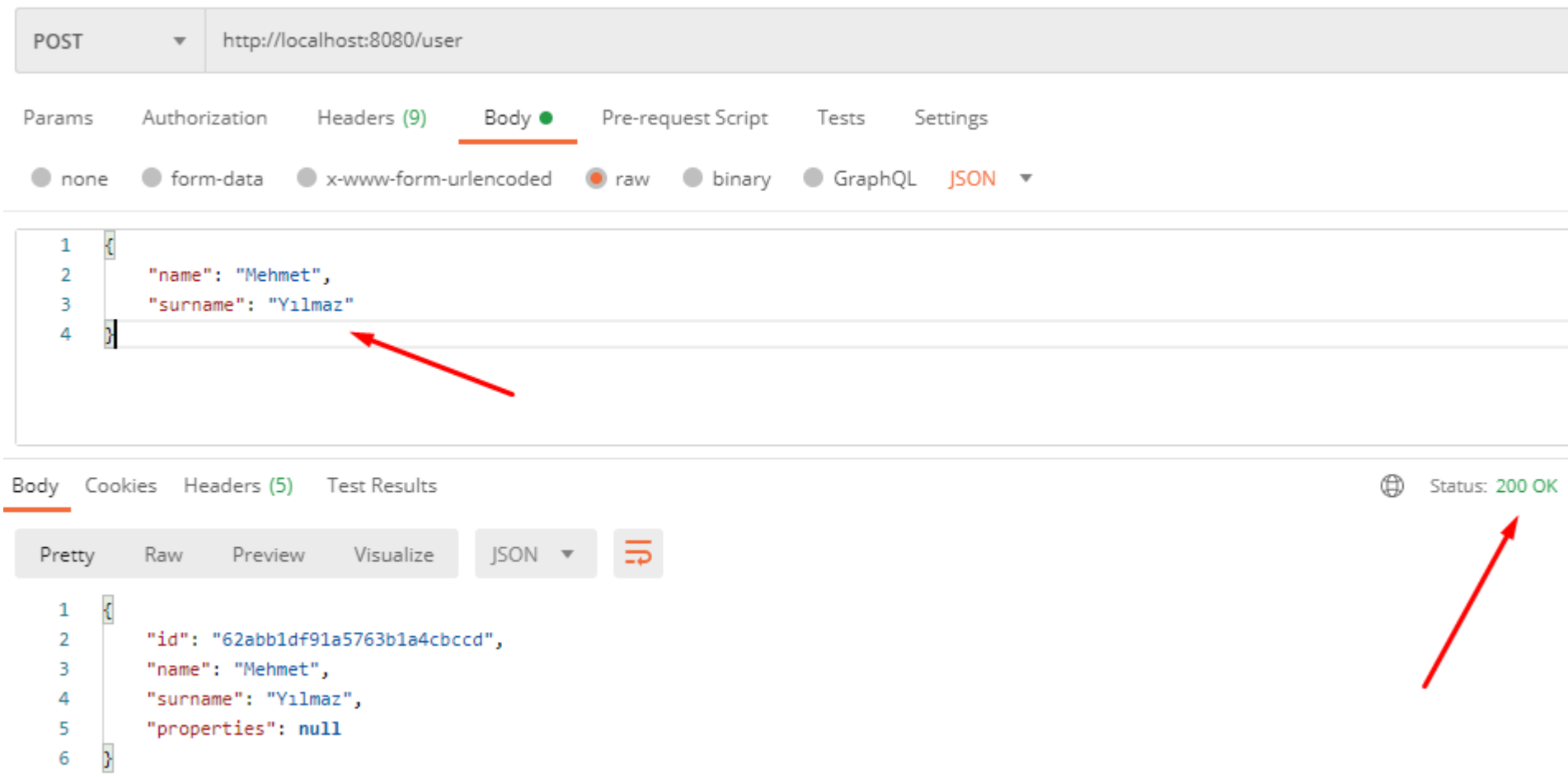
@PostConstruct ile ilk verinin eklenmesini sağladık.

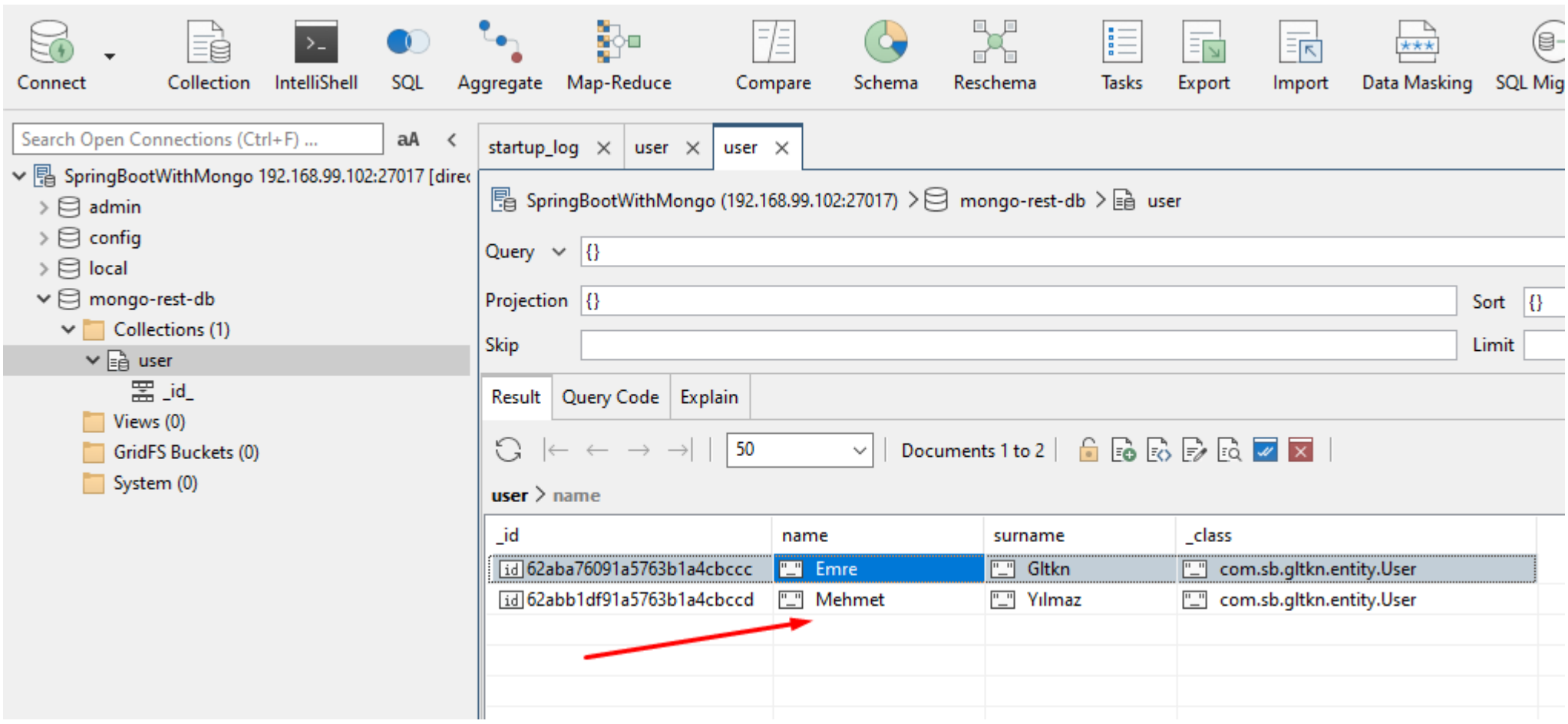
```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import javax.annotation.PostConstruct;
import java.util.List;

@RestController
@RequestMapping("/user")
public class UserAPI {

    3 usages
    @Autowired
    private UserRepository userRepository;

    @PostConstruct
    public void init(){
        User user = new User();
        user.setName("Emre");
        user.setSurname("Gltkn");
        userRepository.save(user);
    }

    @PostMapping
    public ResponseEntity<User> create(@RequestBody User user){
        return ResponseEntity.ok(userRepository.save(user));
    }

    @GetMapping
    public ResponseEntity<List<User>> listAll(){
        return ResponseEntity.ok(userRepository.findAll());
    }
}
```



http://localhost:8080/user

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
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
```

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
```