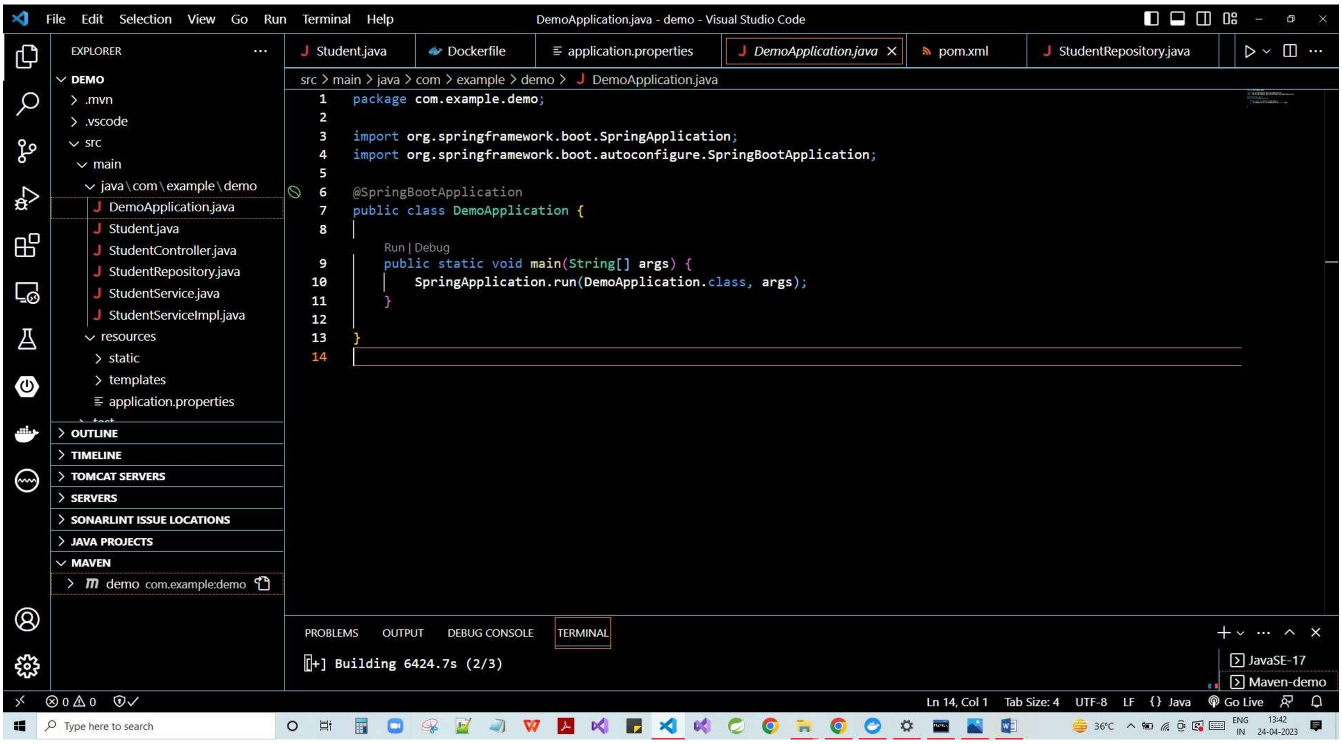
- DOCKER using VS CODE
    - Dockerized the Spring boot application with MYSQL database

// Created Maven project with DemoApplication.java

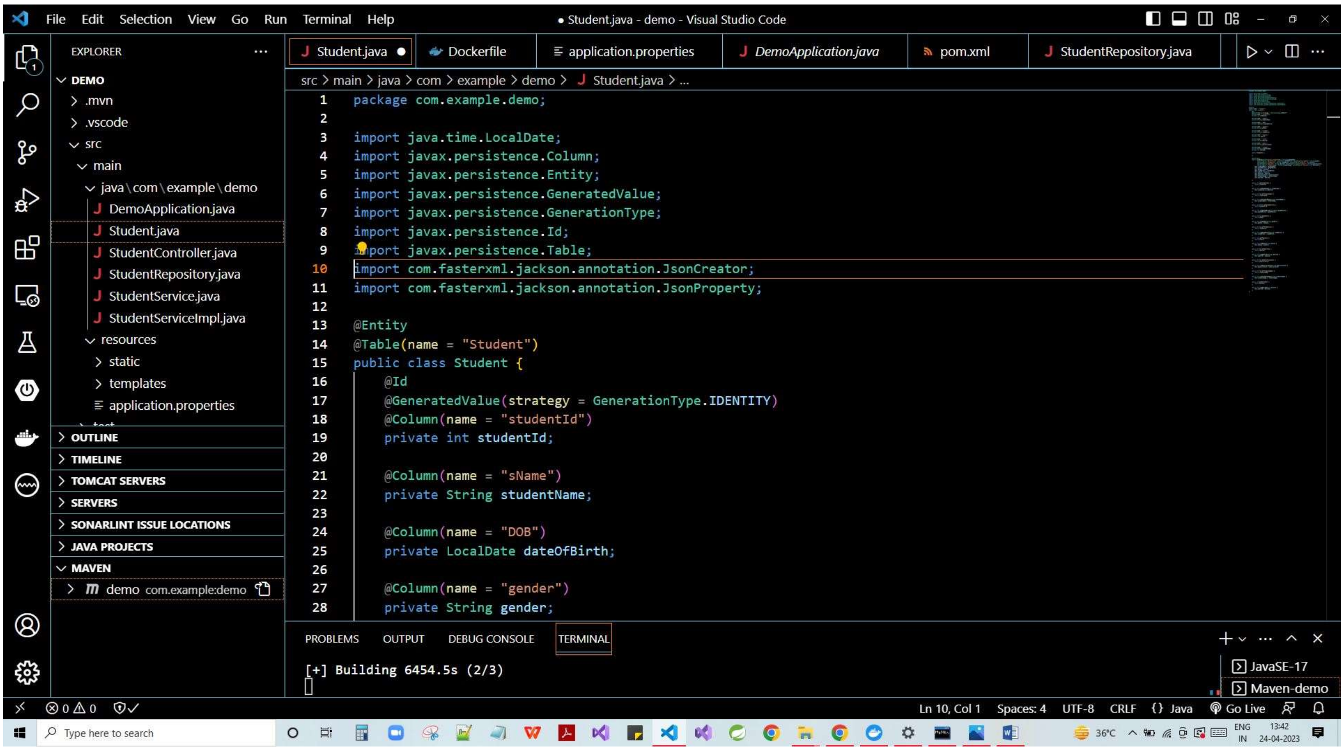


DemoApplication.java - demo - Visual Studio Code
Student.java
Dockerfile
application.properties
DemoApplication.java
pom.xml
StudentRepository.java
src > main > java > com > example > demo > DemoApplication.java

```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

```
[+] Building 6424.7s (2/3)
```

// Created Student Entity

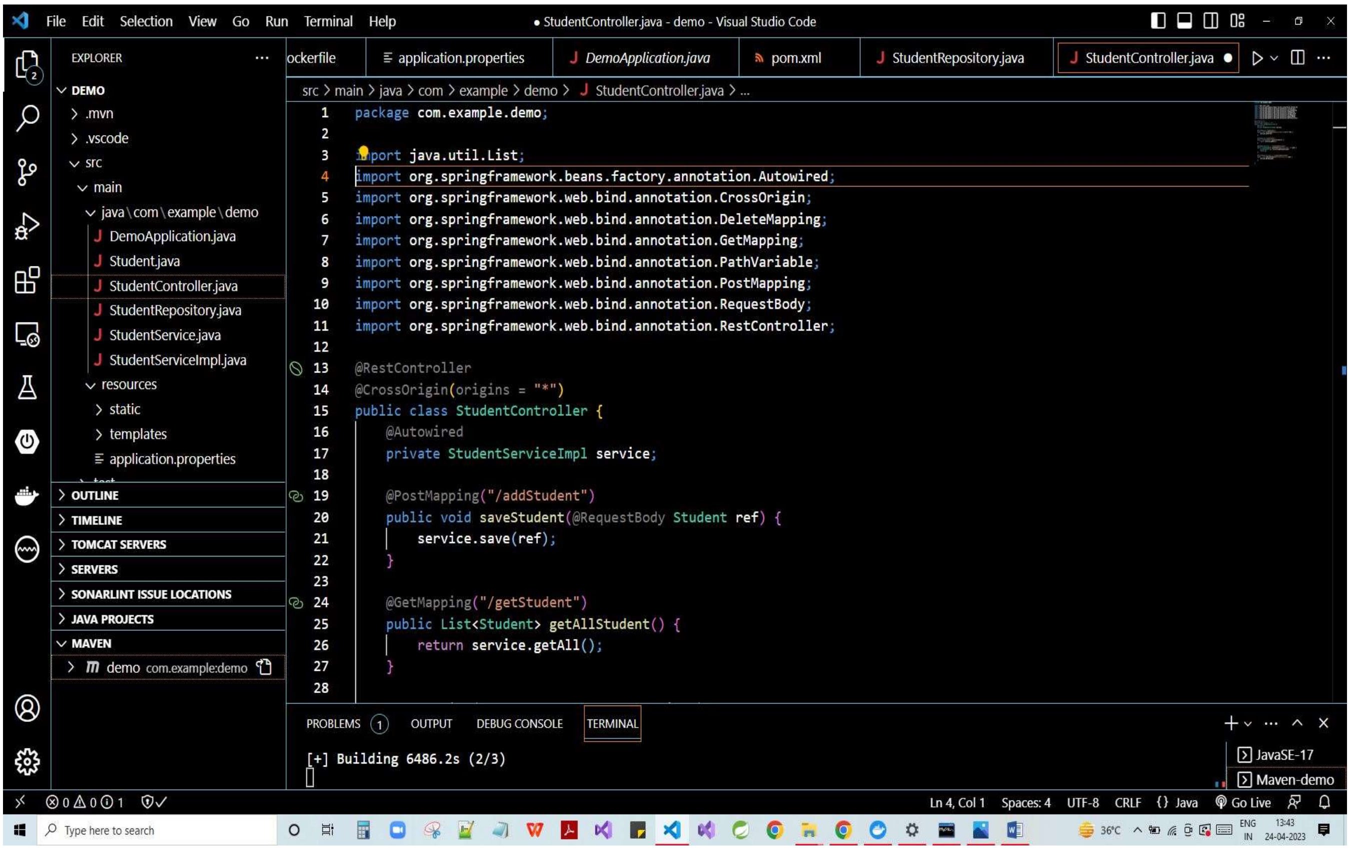


• Student.java - demo - Visual Studio Code
Student.java
Dockerfile
application.properties
DemoApplication.java
pom.xml
StudentRepository.java
src > main > java > com > example > demo > Student.java > ...

```java
package com.example.demo;

import java.time.LocalDate;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import com.fasterxml.jackson.annotation.JsonCreator;
import com.fasterxml.jackson.annotation.JsonProperty;

@Entity
@Table(name = "Student")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "studentId")
    private int studentId;

    @Column(name = "sName")
    private String studentName;

    @Column(name = "DOB")
    private LocalDate dateOfBirth;

    @Column(name = "gender")
    private String gender;
```

```
[+] Building 6454.5s (2/3)
```

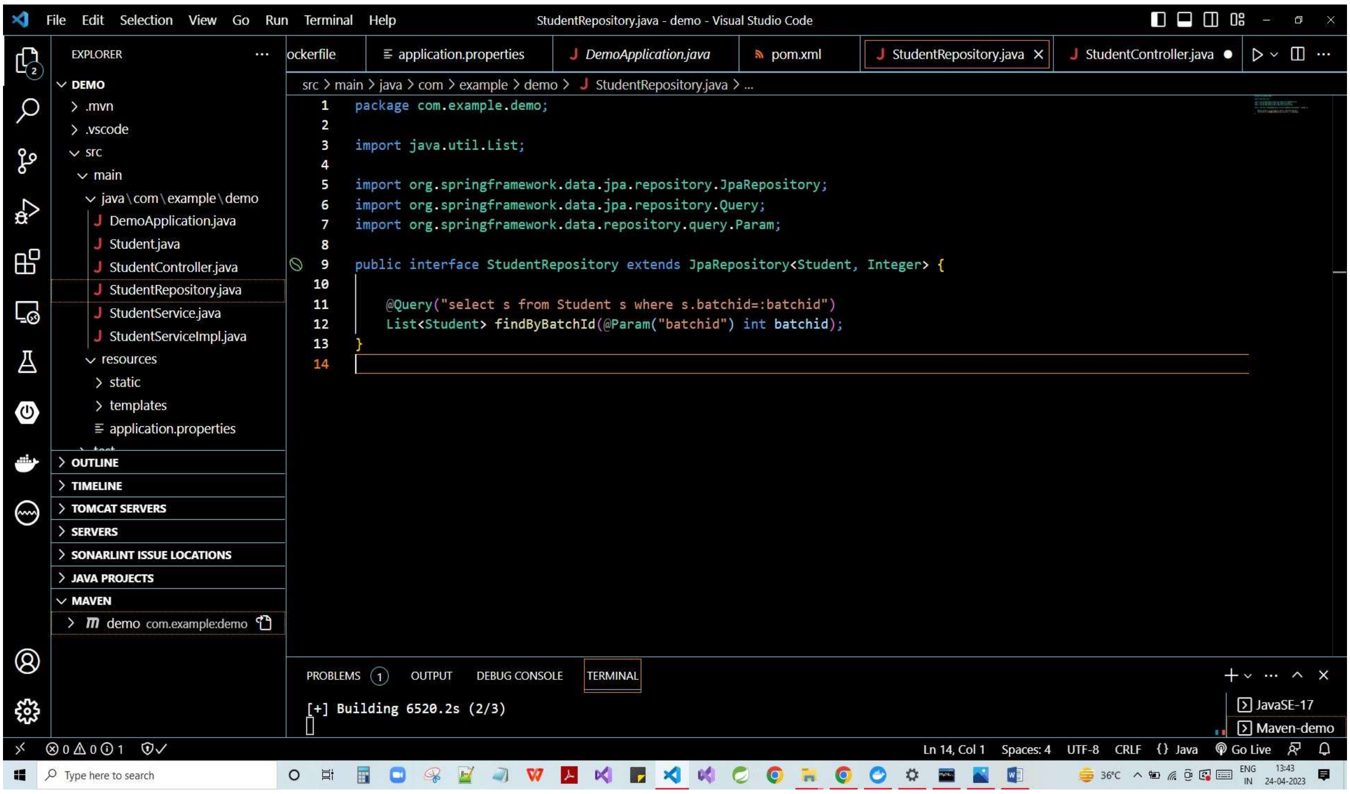
// Created Student Controller class with mapping methods



```java
package com.example.demo;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

@RestController
@CrossOrigin(origins = "*")
public class StudentController {
    @Autowired
    private StudentServiceImpl service;

    @PostMapping("/addStudent")
    public void saveStudent(@RequestBody Student ref) {
        service.save(ref);
    }

    @GetMapping("/getStudent")
    public List<Student> getAllStudent() {
        return service.getAll();
    }

```

```
[+] Building 6486.2s (2/3)
```

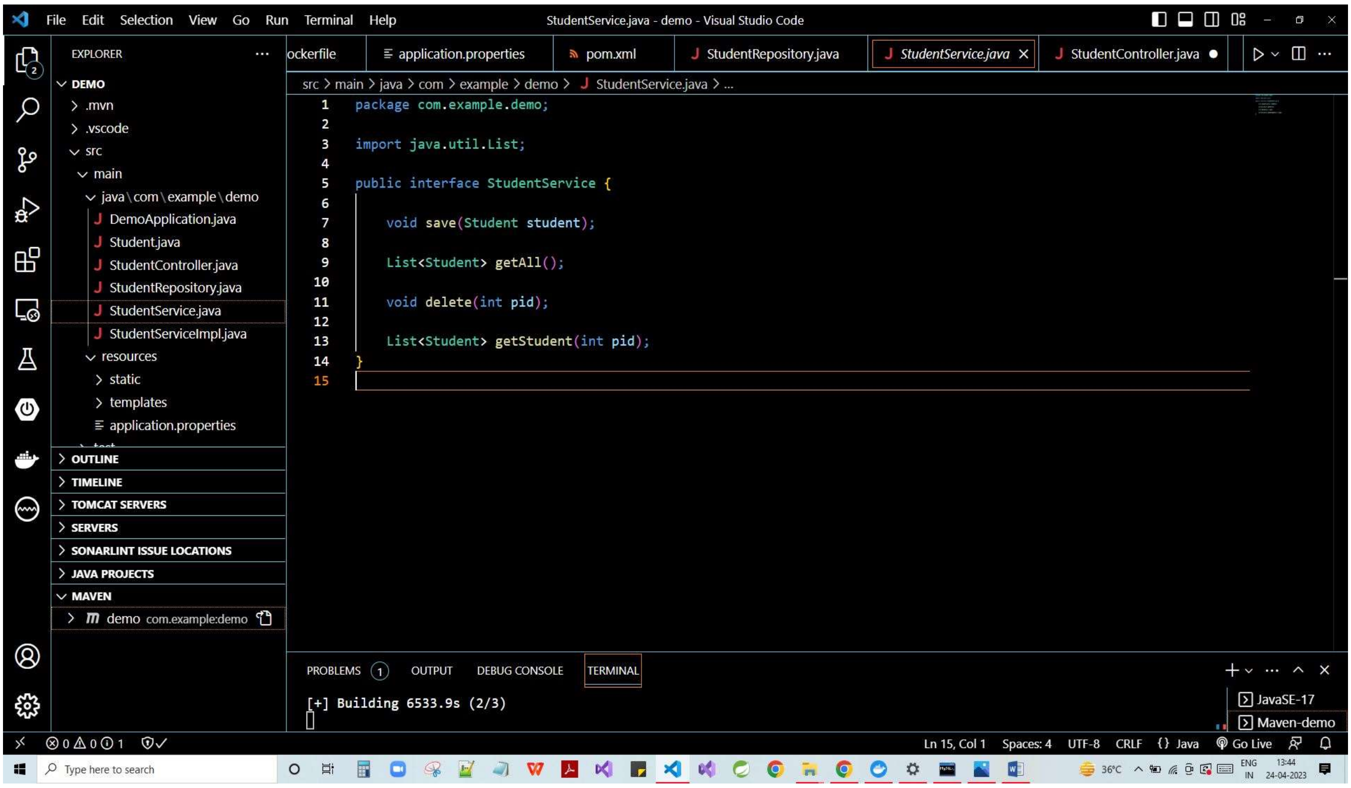
// Created Student Repository class using JPA Repository


StudentRepository.java - demo - Visual Studio Code
EXPLORER
DEMO
src > main > java > com > example > demo > StudentRepository.java > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JavaSE-17
Maven-demo
Ln 14, Col 1

```java
package com.example.demo;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

public interface StudentRepository extends JpaRepository<Student, Integer> {

    @Query("select s from Student s where s.batchid=:batchid")
    List<Student> findByBatchId(@Param("batchid") int batchid);
}

```

```
[+] Building 6520.2s (2/3)
```

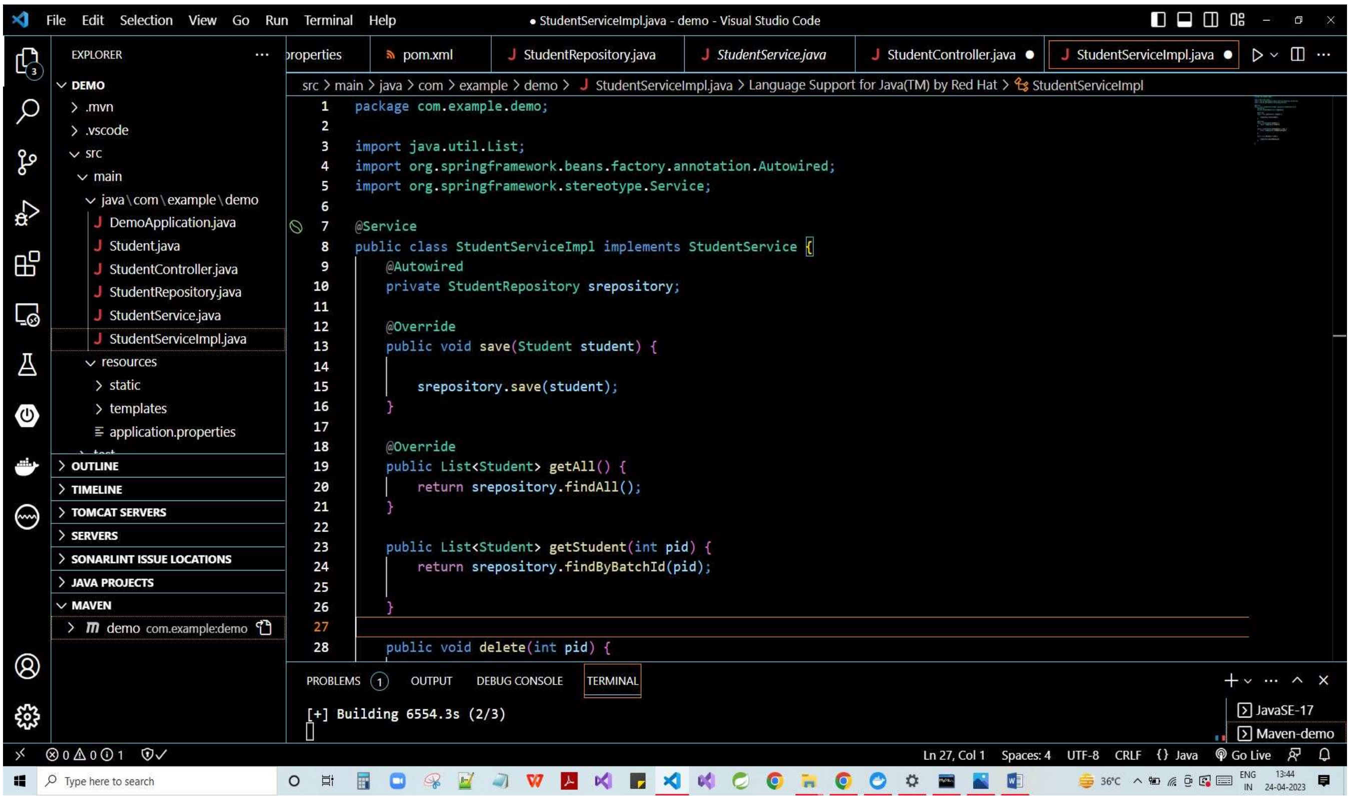
// Created Student Service interface



```java
package com.example.demo;

import java.util.List;

public interface StudentService {

    void save(Student student);

    List<Student> getAll();

    void delete(int pid);

    List<Student> getStudent(int pid);
}
```

```
[+] Building 6533.9s (2/3)
```

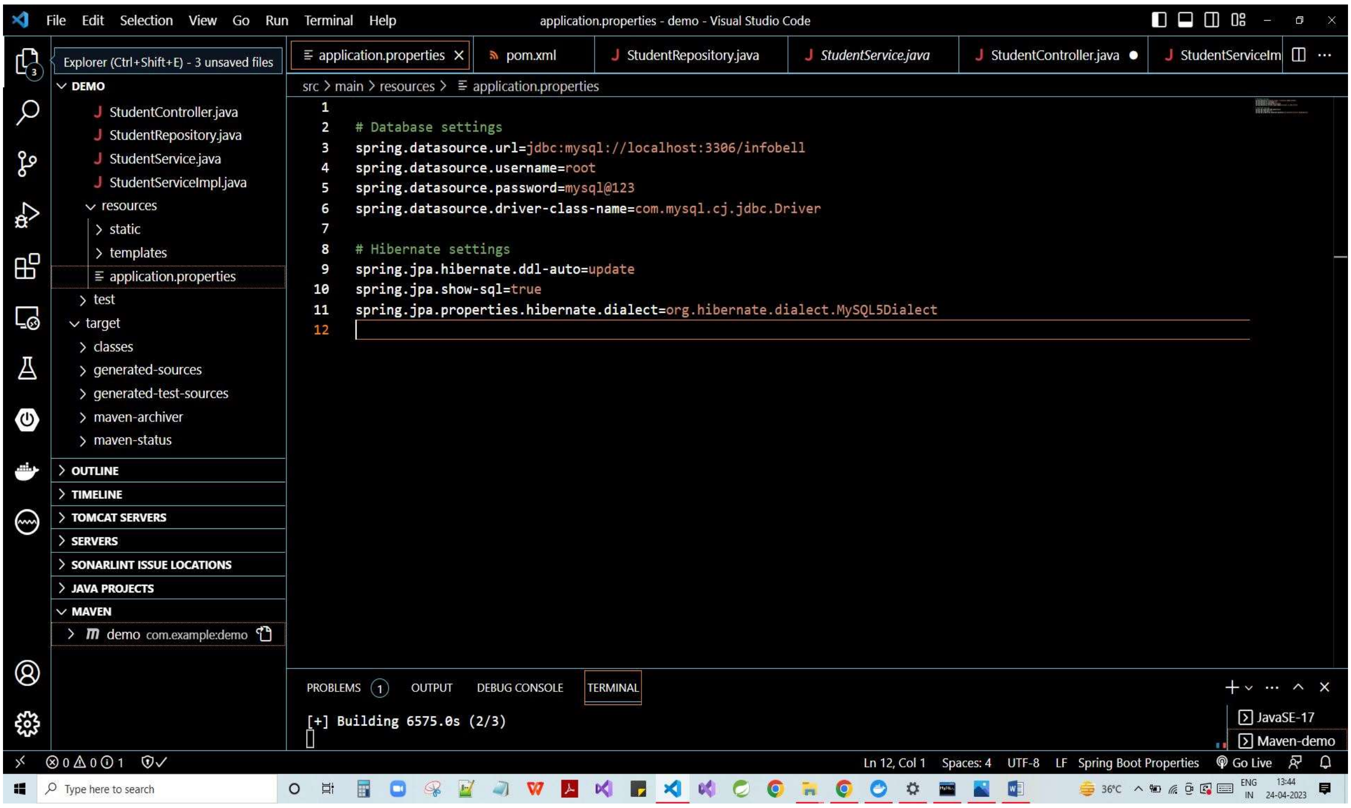
// Created Implementation class for Student Service interface


● StudentServiceImpl.java - demo - Visual Studio Code
EXPLORER
DEMO
StudentServiceImpl.java
src > main > java > com > example > demo > StudentServiceImpl.java > Language Support for Java(TM) by Red Hat > StudentServiceImpl
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Ln 27, Col 1
Spaces: 4
UTF-8
CRLF
Java
Go Live

```java
package com.example.demo;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class StudentServiceImpl implements StudentService {
    @Autowired
    private StudentRepository srepository;

    @Override
    public void save(Student student) {

        srepository.save(student);
    }

    @Override
    public List<Student> getAll() {
        return srepository.findAll();
    }

    public List<Student> getStudent(int pid) {
        return srepository.findByBatchId(pid);

    }

    public void delete(int pid) {
```

```
[+] Building 6554.3s (2/3)
```

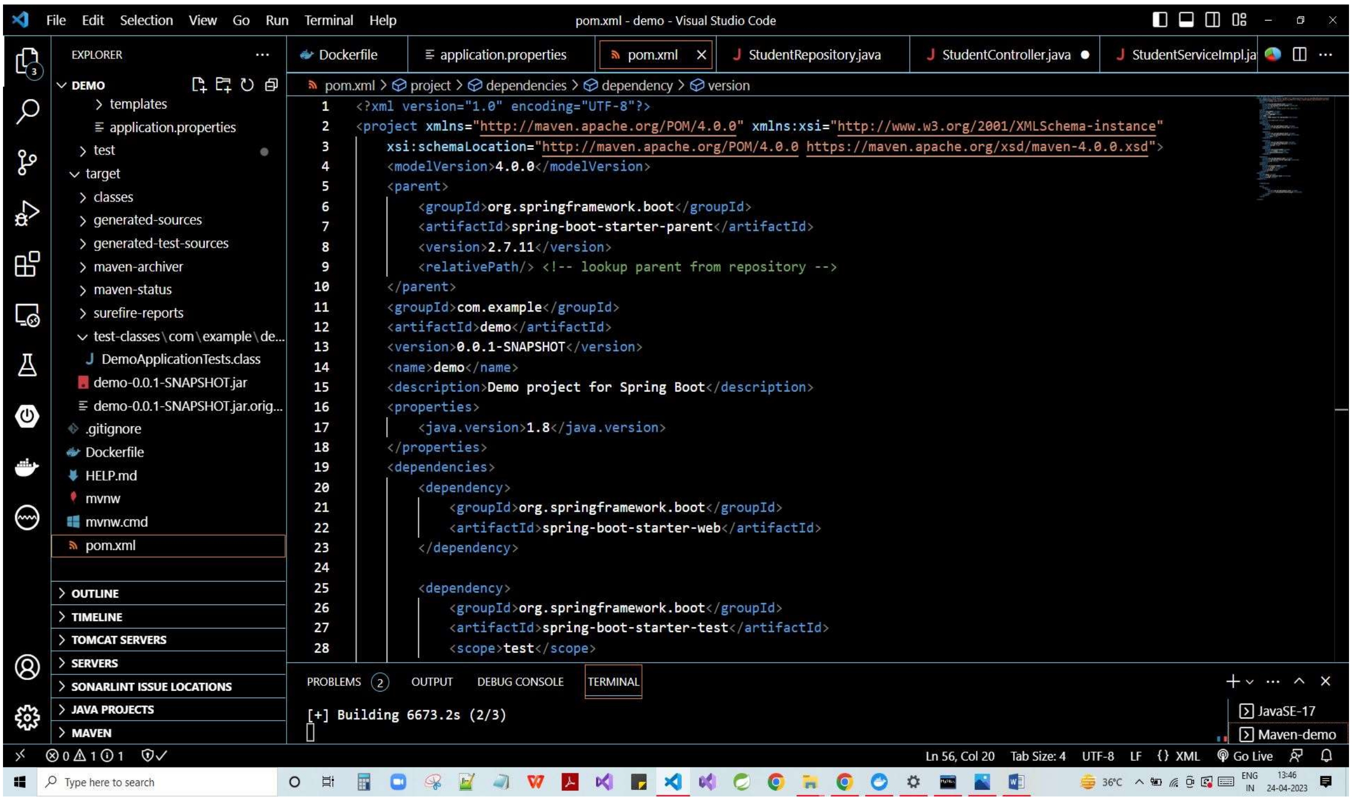
// Created application.properties file with database credentials



```
# Database settings
spring.datasource.url=jdbc:mysql://localhost:3306/infobell
spring.datasource.username=root
spring.datasource.password=mysql@123
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
```

```
[+] Building 6575.0s (2/3)
```

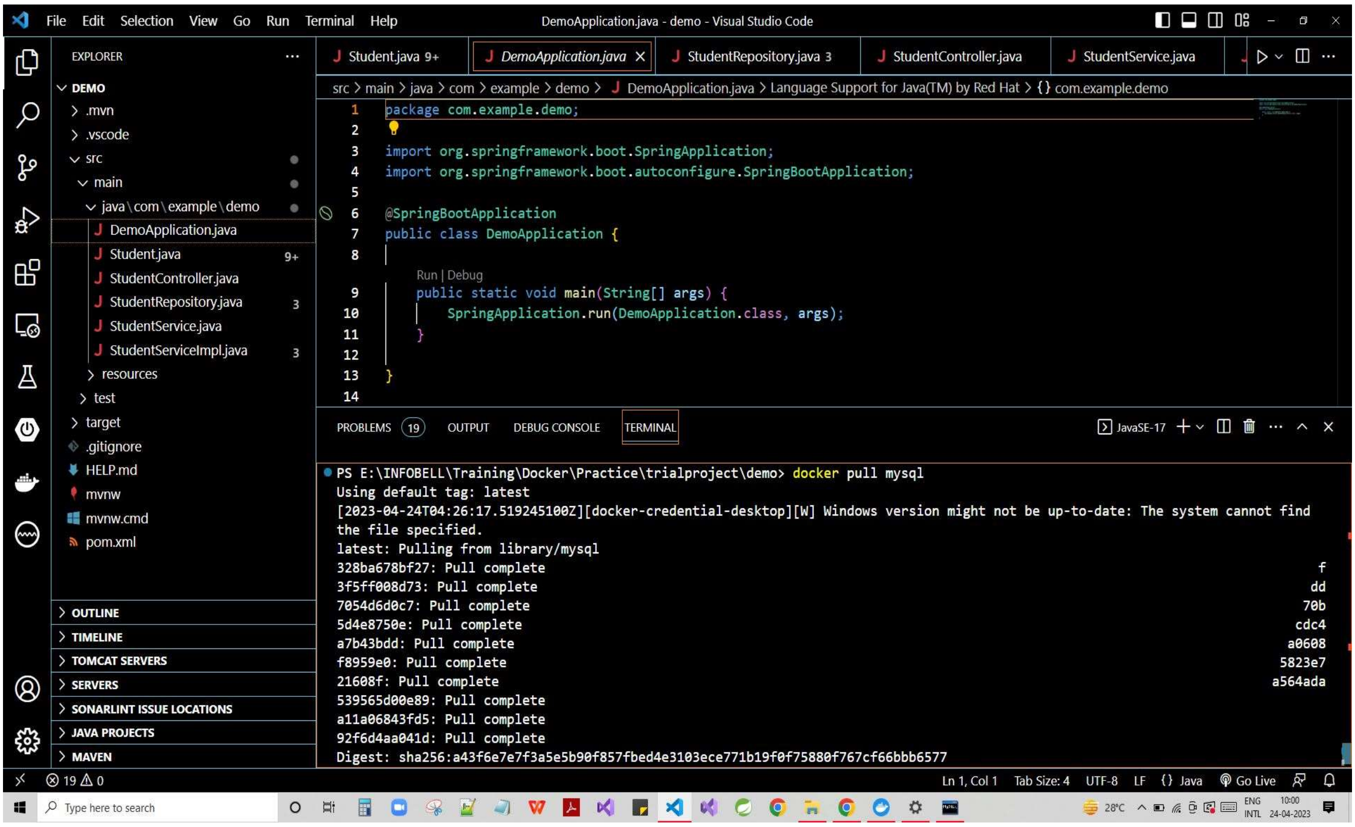
// Added required dependencies to Pom.xml file



```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 https://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>2.7.11</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>demo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
```

```
[+] Building 6673.2s (2/3)
```

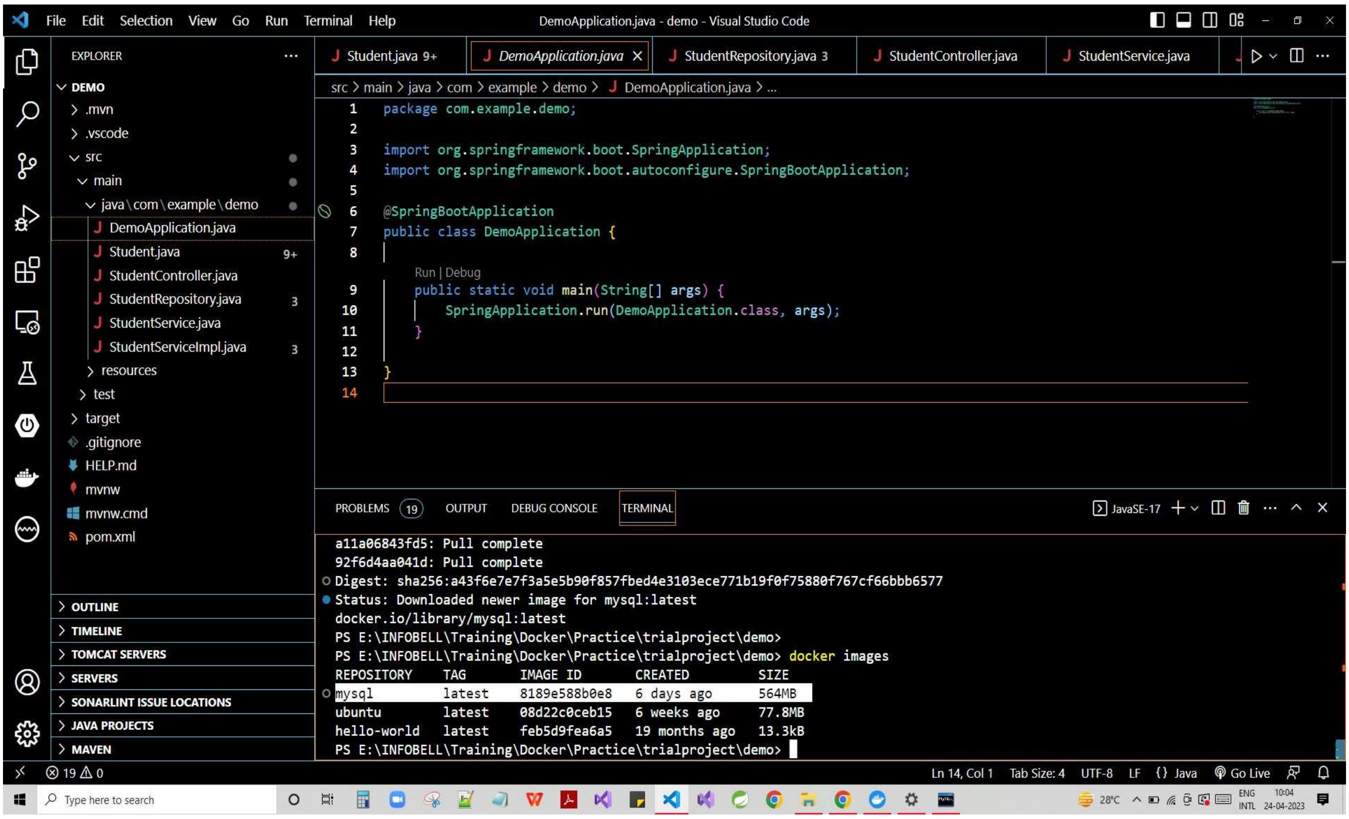
// Pulled mysql image from docker hub



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

```
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker pull mysql
Using default tag: latest
[2023-04-24T04:26:17.519245100Z][docker-credential-desktop][W] Windows version might not be up-to-date: The system cannot find
the file specified.
latest: Pulling from library/mysql
328ba678bf27: Pull complete                                                                                                  f
3f5ff008d73: Pull complete                                                                                                  dd
7054d6d0c7: Pull complete                                                                                                  70b
5d4e8750e: Pull complete                                                                                                  cdc4
a7b43bdd: Pull complete                                                                                                  a0608
f8959e0: Pull complete                                                                                                  5823e7
21608f: Pull complete                                                                                                   a564ada
539565d00e89: Pull complete
a11a06843fd5: Pull complete
92f6d4aa041d: Pull complete
Digest: sha256:a43f6e7e7f3a5e5b90f857fbed4e3103ece771b19f0f75880f767cf66bbb6577
```

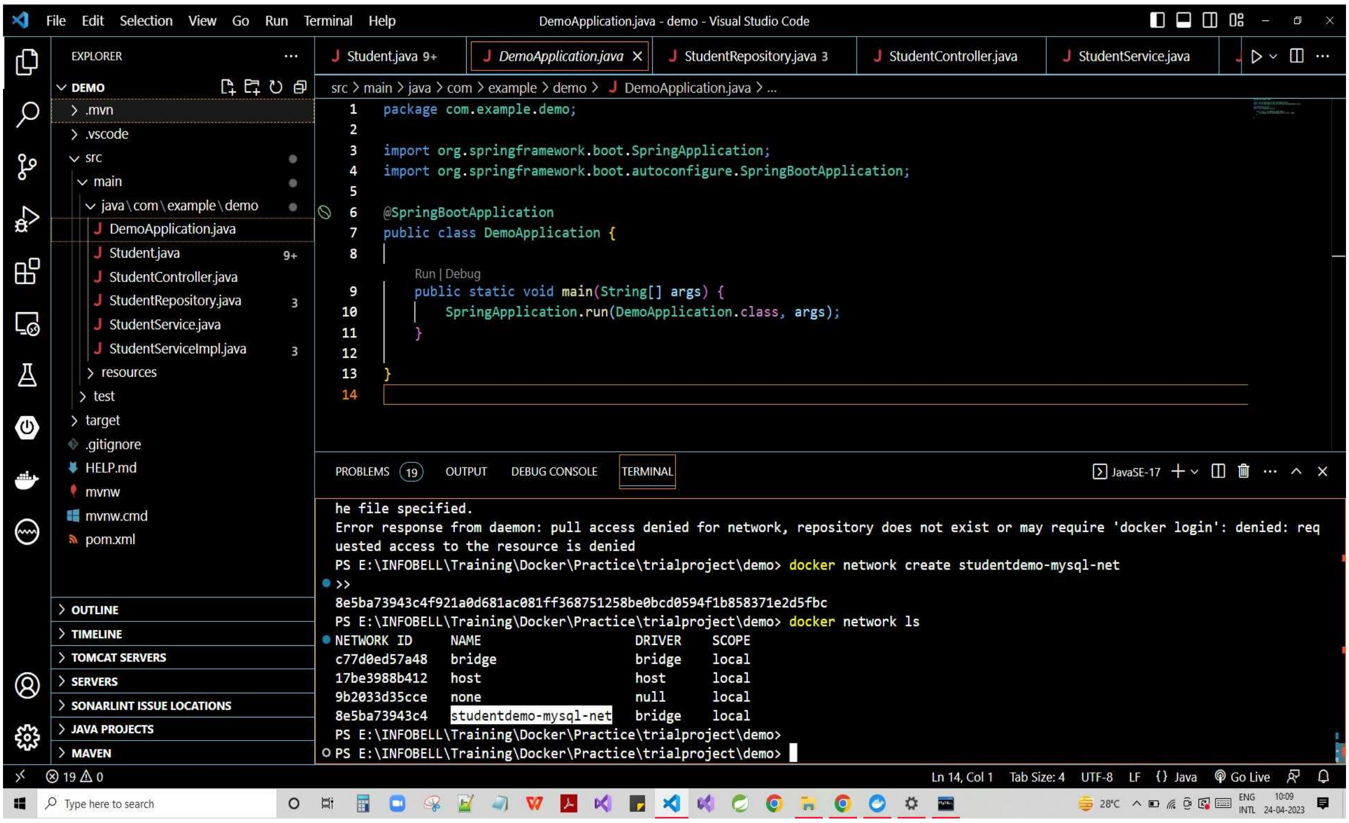
// to check the pulled image from docker hub (docker images)



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

```
a11a06843fd5: Pull complete
92f6d4aa041d: Pull complete
Digest: sha256:a43f6e7e7f3a5e5b90f857fbed4e3103ece771b19f0f75880f767cf66bbb6577
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
mysql         latest    8189e588b0e8   6 days ago      564MB
ubuntu        latest    08d22c0ceb15   6 weeks ago     77.8MB
hello-world   latest    feb5d9fea6a5   19 months ago   13.3kB
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
```

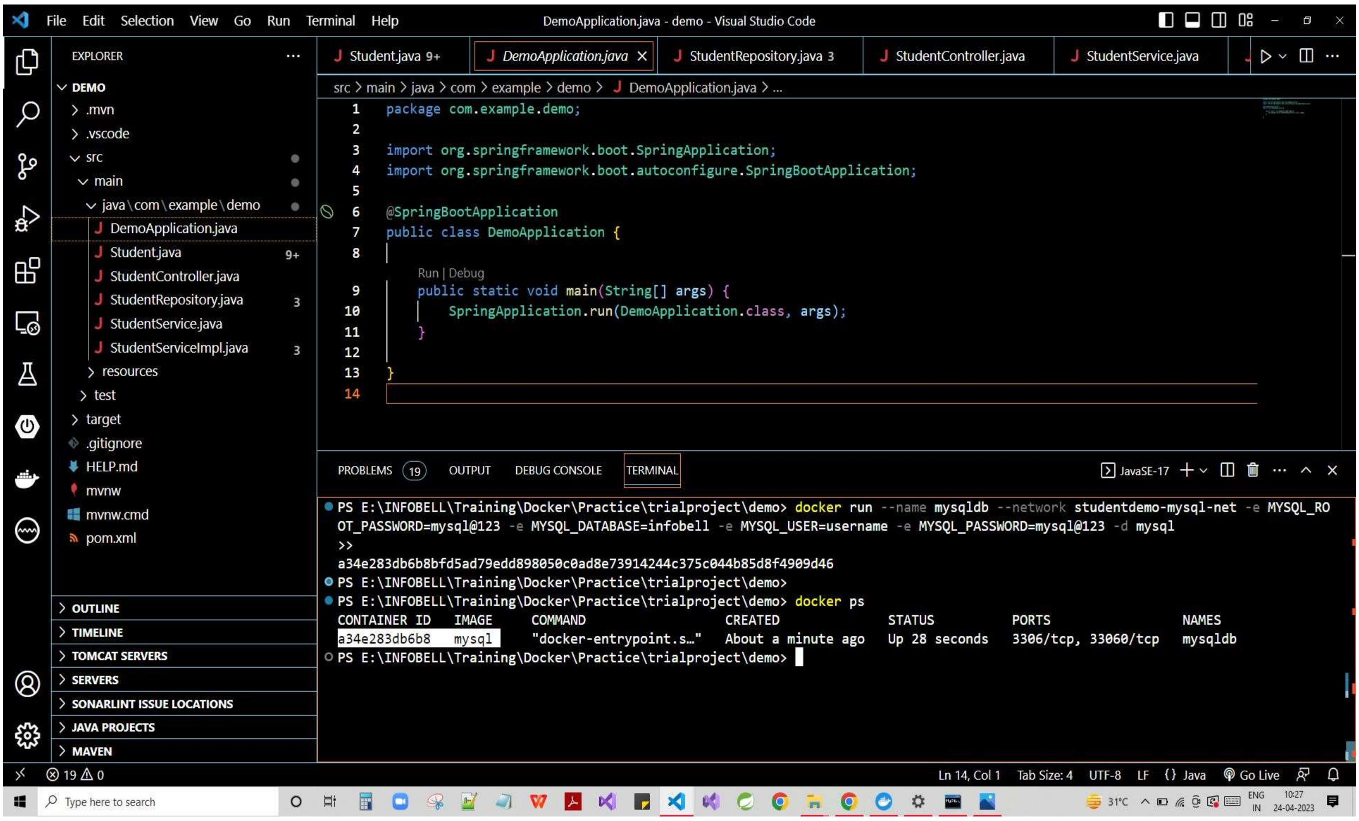
// Created a network to communicate between springboot application & MYSQL database & checked



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

```
he file specified.
Error response from daemon: pull access denied for network, repository does not exist or may require 'docker login': denied: req
uested access to the resource is denied
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker network create studentdemo-mysql-net
>>
8e5ba73943c4f921a0d681ac081ff368751258be0bcd0594f1b858371e2d5fbc
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker network ls
NETWORK ID     NAME                   DRIVER    SCOPE
c77d0ed57a48   bridge                 bridge    local
17be3988b412   host                   host      local
9b2033d35cce   none                   null      local
8e5ba73943c4   studentdemo-mysql-net  bridge    local
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
```

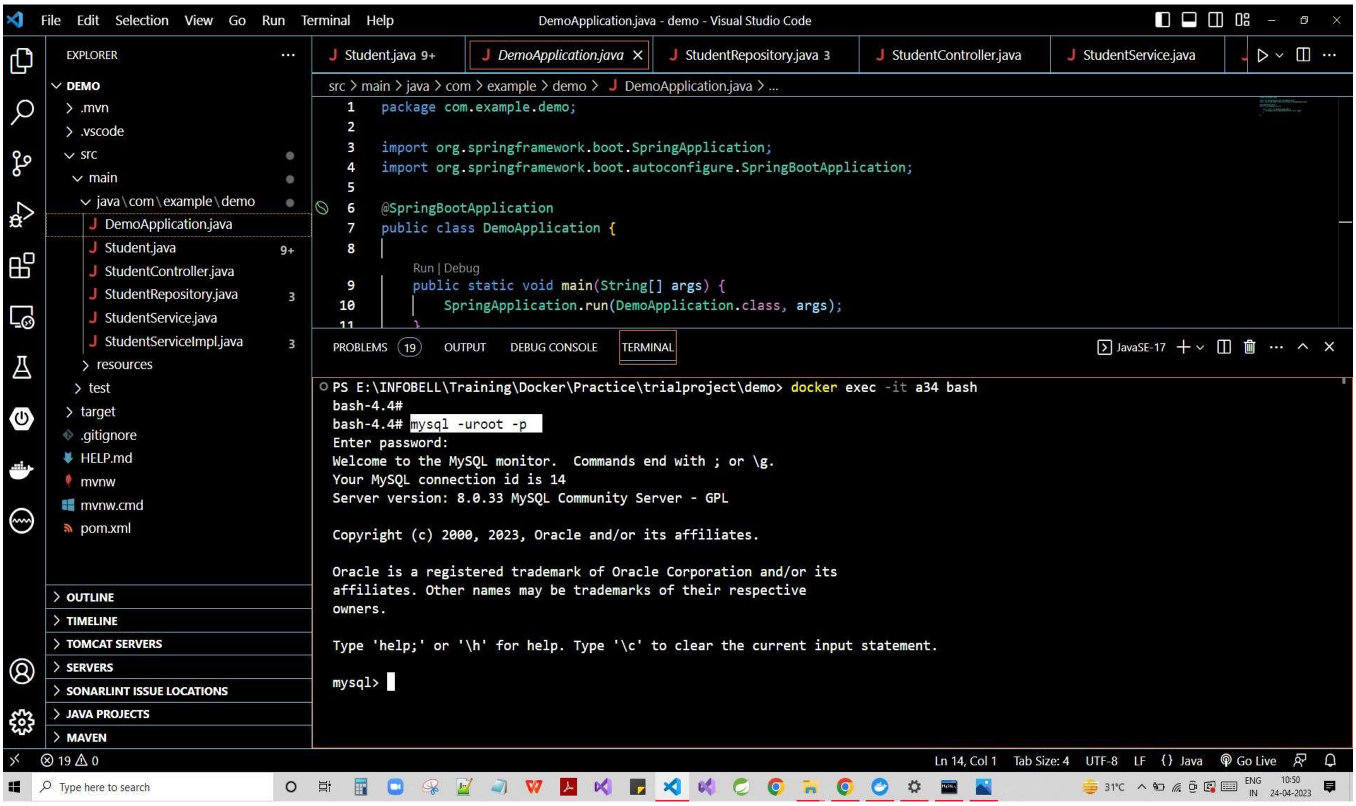
// to Run the mysql container in the network (docker run …..environment variables)



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

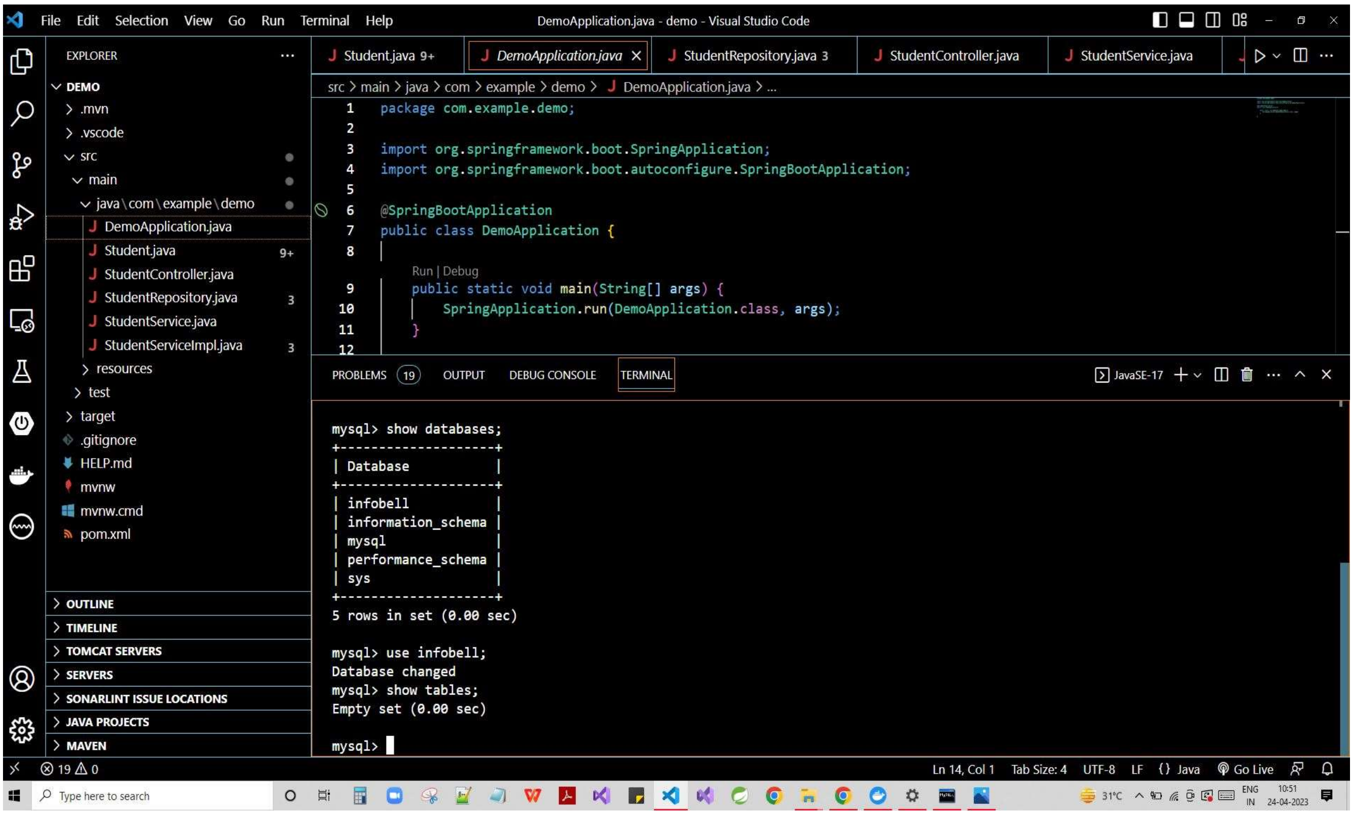
```
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker run --name mysqldb --network studentdemo-mysql-net -e MYSQL_RO
OT_PASSWORD=mysql@123 -e MYSQL_DATABASE=infobell -e MYSQL_USER=username -e MYSQL_PASSWORD=mysql@123 -d mysql
>>
a34e283db6b8bfd5ad79edd898050c0ad8e73914244c375c044b85d8f4909d46
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS          PORTS                 NAMES
a34e283db6b8   mysql     "docker-entrypoint.s…"   About a minute ago   Up 28 seconds   3306/tcp, 33060/tcp   mysqldb
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
```

// to run in the interactive mode (docker exec –it) & to check the created database



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
```

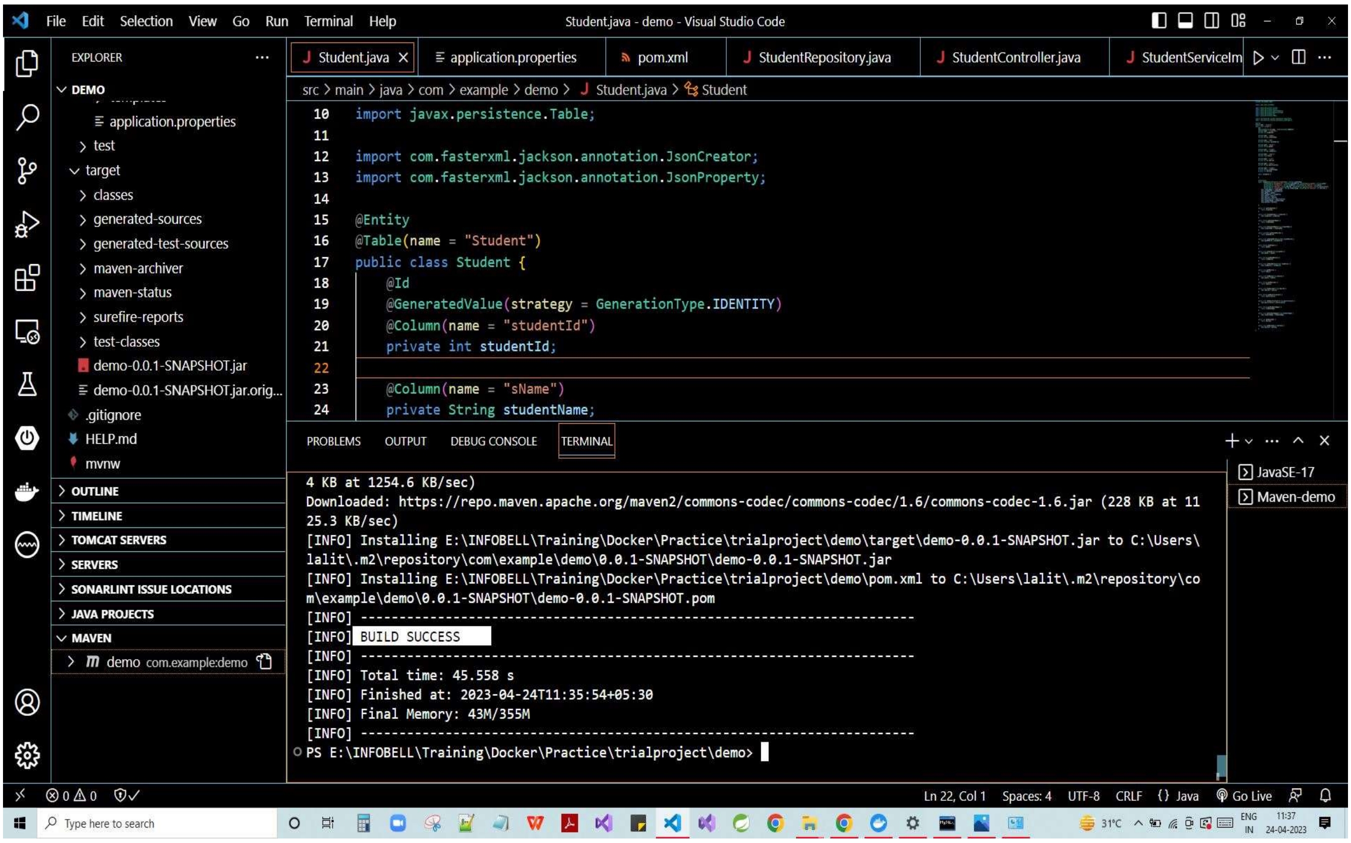
```
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker exec -it a34 bash
bash-4.4#
bash-4.4# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| infobell           |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use infobell;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> 
```

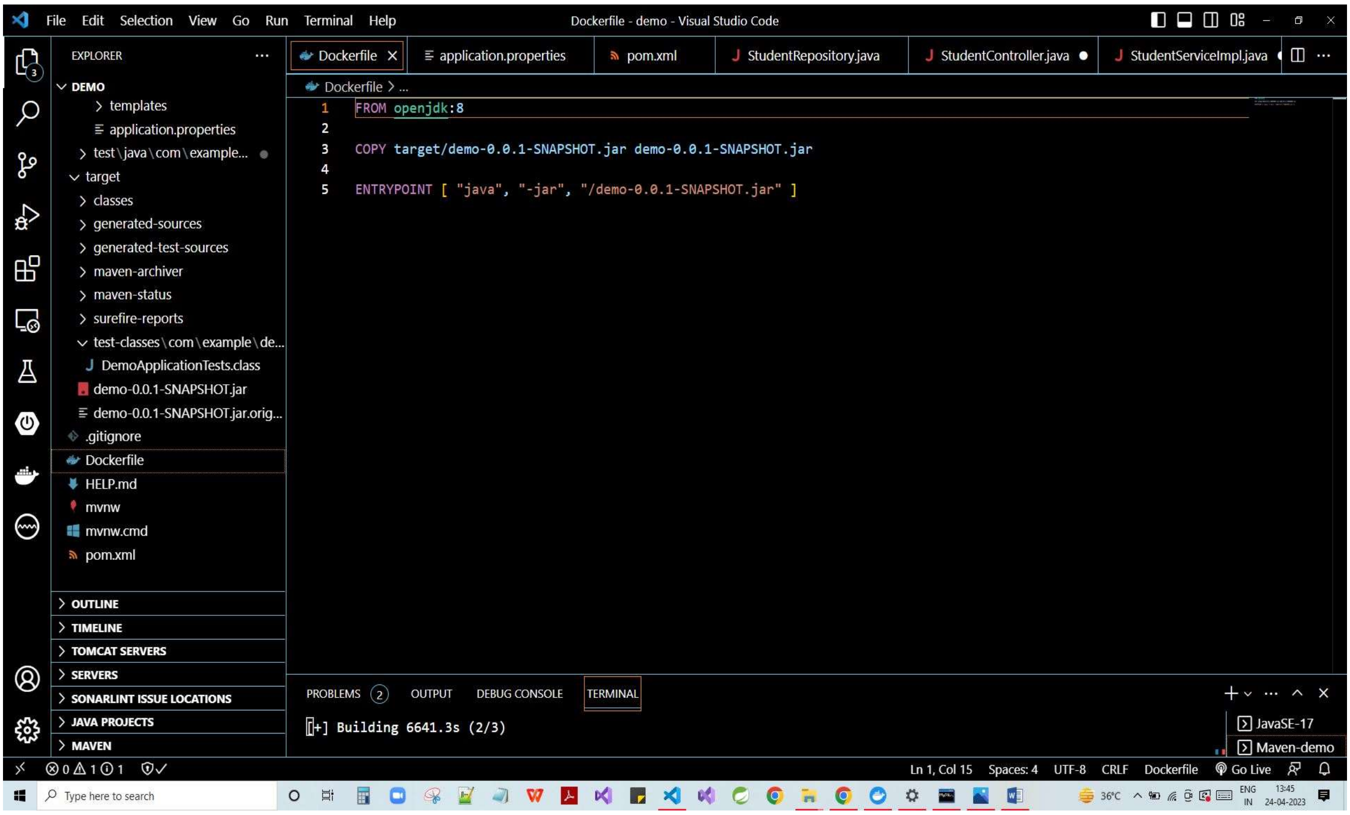
// Created a project .jar file in test-classes inside target folder



```java
import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonCreator;
import com.fasterxml.jackson.annotation.JsonProperty;

@Entity
@Table(name = "Student")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "studentId")
    private int studentId;

    @Column(name = "sName")
    private String studentName;
```

```
4 KB at 1254.6 KB/sec)
Downloaded: https://repo.maven.apache.org/maven2/commons-codec/commons-codec/1.6/commons-codec-1.6.jar (228 KB at 11
25.3 KB/sec)
[INFO] Installing E:\INFOBELL\Training\Docker\Practice\trialproject\demo\target\demo-0.0.1-SNAPSHOT.jar to C:\Users\
lalit\.m2\repository\com\example\demo\0.0.1-SNAPSHOT\demo-0.0.1-SNAPSHOT.jar
[INFO] Installing E:\INFOBELL\Training\Docker\Practice\trialproject\demo\pom.xml to C:\Users\lalit\.m2\repository\co
m\example\demo\0.0.1-SNAPSHOT\demo-0.0.1-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 45.558 s
[INFO] Finished at: 2023-04-24T11:35:54+05:30
[INFO] Final Memory: 43M/355M
[INFO] ------------------------------------------------------------------------
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
```

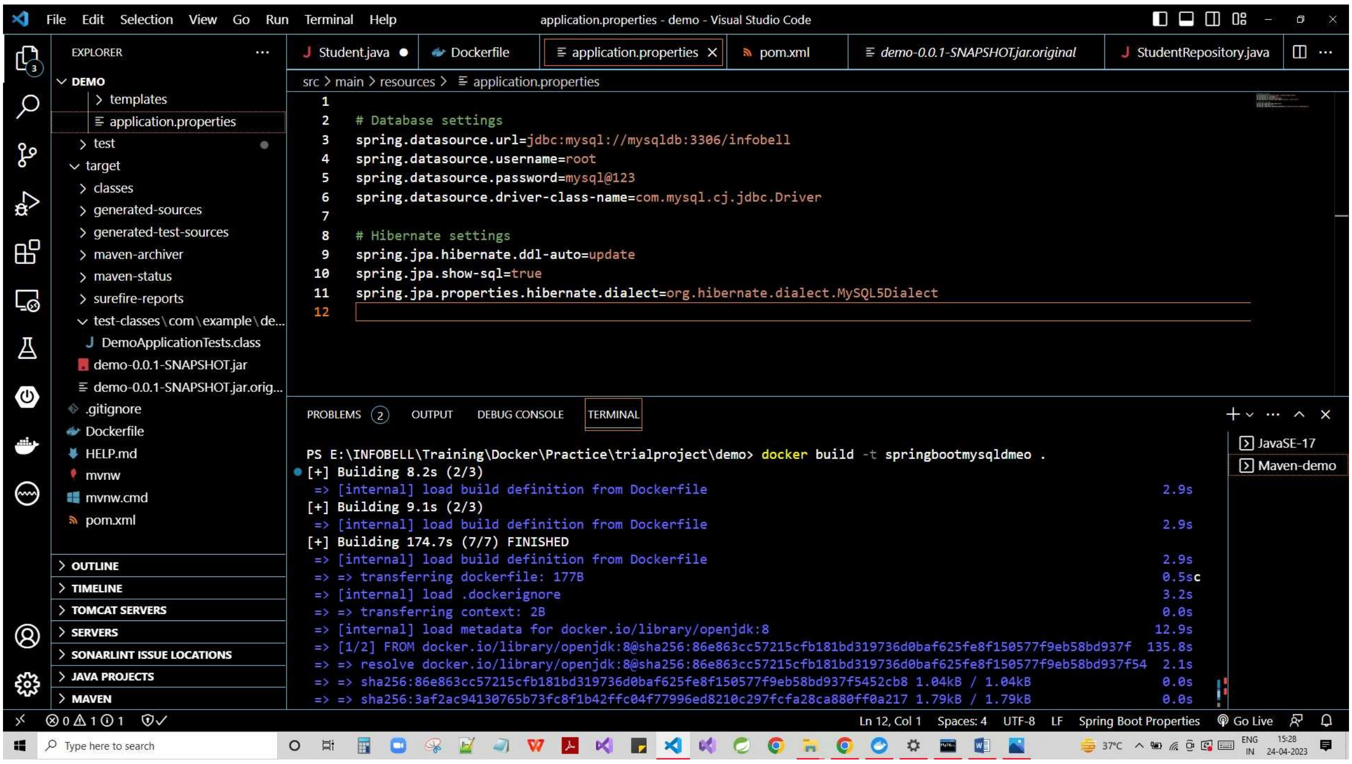
// Created a Docker file and copying project jar file with entrypoint



```dockerfile
FROM openjdk:8

COPY target/demo-0.0.1-SNAPSHOT.jar demo-0.0.1-SNAPSHOT.jar

ENTRYPOINT [ "java", "-jar", "/demo-0.0.1-SNAPSHOT.jar" ]
```

```
[+] Building 6641.3s (2/3)
```

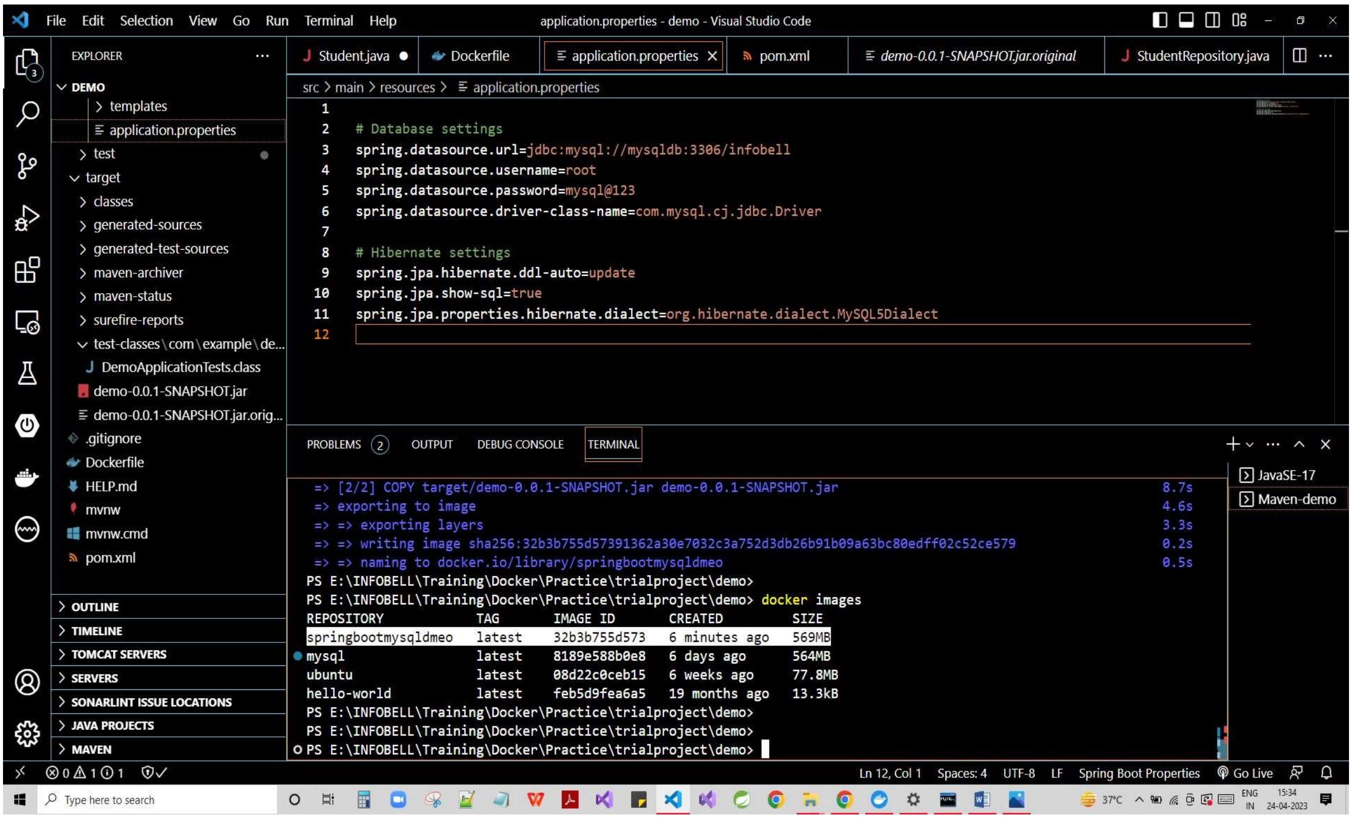
// to build the springboot docker image (docker build –t springbootmysqldemo)


application.properties - demo - Visual Studio Code
Student.java
Dockerfile
application.properties
pom.xml
demo-0.0.1-SNAPSHOT.jar.original
StudentRepository.java
src > main > resources > application.properties
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
JavaSE-17
Maven-demo

```
# Database settings
spring.datasource.url=jdbc:mysql://mysqldb:3306/infobell
spring.datasource.username=root
spring.datasource.password=mysql@123
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
```

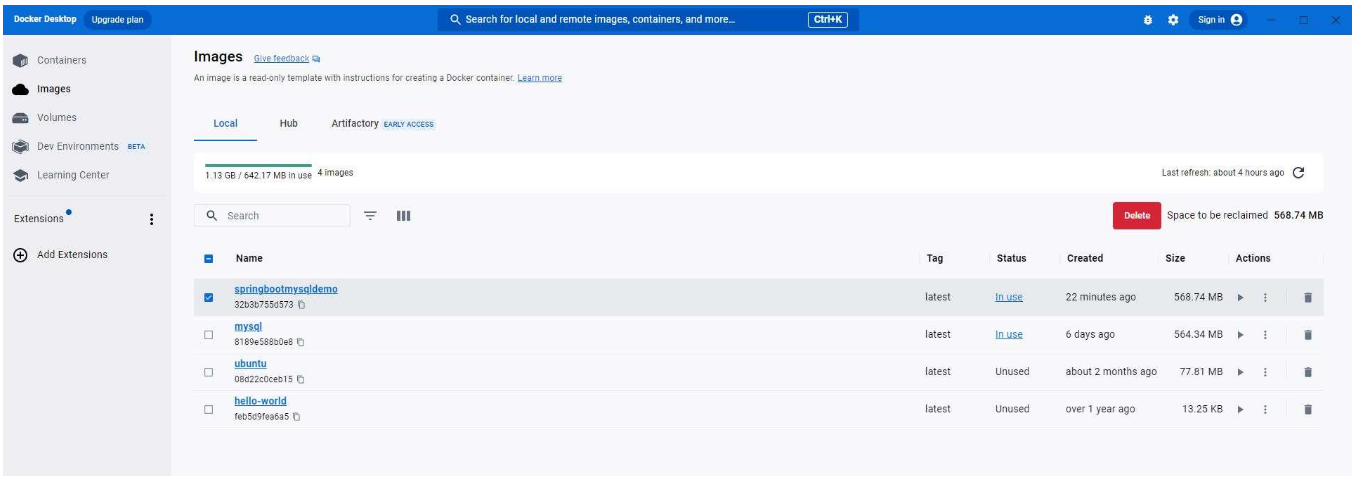
```
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker build -t springbootmysqldmeo .
[+] Building 8.2s (2/3)
 => [internal] load build definition from Dockerfile                                          2.9s
[+] Building 9.1s (2/3)
 => [internal] load build definition from Dockerfile                                          2.9s
[+] Building 174.7s (7/7) FINISHED
 => [internal] load build definition from Dockerfile                                          2.9s
 => => transferring dockerfile: 177B                                                          0.5sc
 => [internal] load .dockerignore                                                             3.2s
 => => transferring context: 2B                                                               0.0s
 => [internal] load metadata for docker.io/library/openjdk:8                                 12.9s
 => [1/2] FROM docker.io/library/openjdk:8@sha256:86e863cc57215cfb181bd319736d0baf625fe8f150577f9eb58bd937f  135.8s
 => => resolve docker.io/library/openjdk:8@sha256:86e863cc57215cfb181bd319736d0baf625fe8f150577f9eb58bd937f54  2.1s
 => => sha256:86e863cc57215cfb181bd319736d0baf625fe8f150577f9eb58bd937f5452cb8 1.04kB / 1.04kB  0.0s
 => => sha256:3af2ac94130765b73fc8f1b42ffc04f77996ed8210c297fcfa28ca880ff0a217 1.79kB / 1.79kB  0.0s
```

// Docker image created (springbootmysqldemo)


application.properties - demo - Visual Studio Code
Student.java
Dockerfile
application.properties
pom.xml
demo-0.0.1-SNAPSHOT.jar.original
StudentRepository.java
src > main > resources > application.properties
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
JavaSE-17
Maven-demo

```
# Database settings
spring.datasource.url=jdbc:mysql://mysqldb:3306/infobell
spring.datasource.username=root
spring.datasource.password=mysql@123
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
```

```
 => [2/2] COPY target/demo-0.0.1-SNAPSHOT.jar demo-0.0.1-SNAPSHOT.jar                         8.7s
 => exporting to image                                                                        4.6s
 => => exporting layers                                                                       3.3s
 => => writing image sha256:32b3b755d57391362a30e7032c3a752d3db26b91b09a63bc80edff02c52ce579  0.2s
 => => naming to docker.io/library/springbootmysqldmeo                                        0.5s
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker images
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
springbootmysqldmeo   latest    32b3b755d573   6 minutes ago    569MB
mysql                 latest    8189e588b0e8   6 days ago       564MB
ubuntu                latest    08d22c0ceb15   6 weeks ago      77.8MB
hello-world           latest    feb5d9fea6a5   19 months ago    13.3kB
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo>
```



Images Give feedback

An image is a read-only template with instructions for creating a Docker container. Learn more

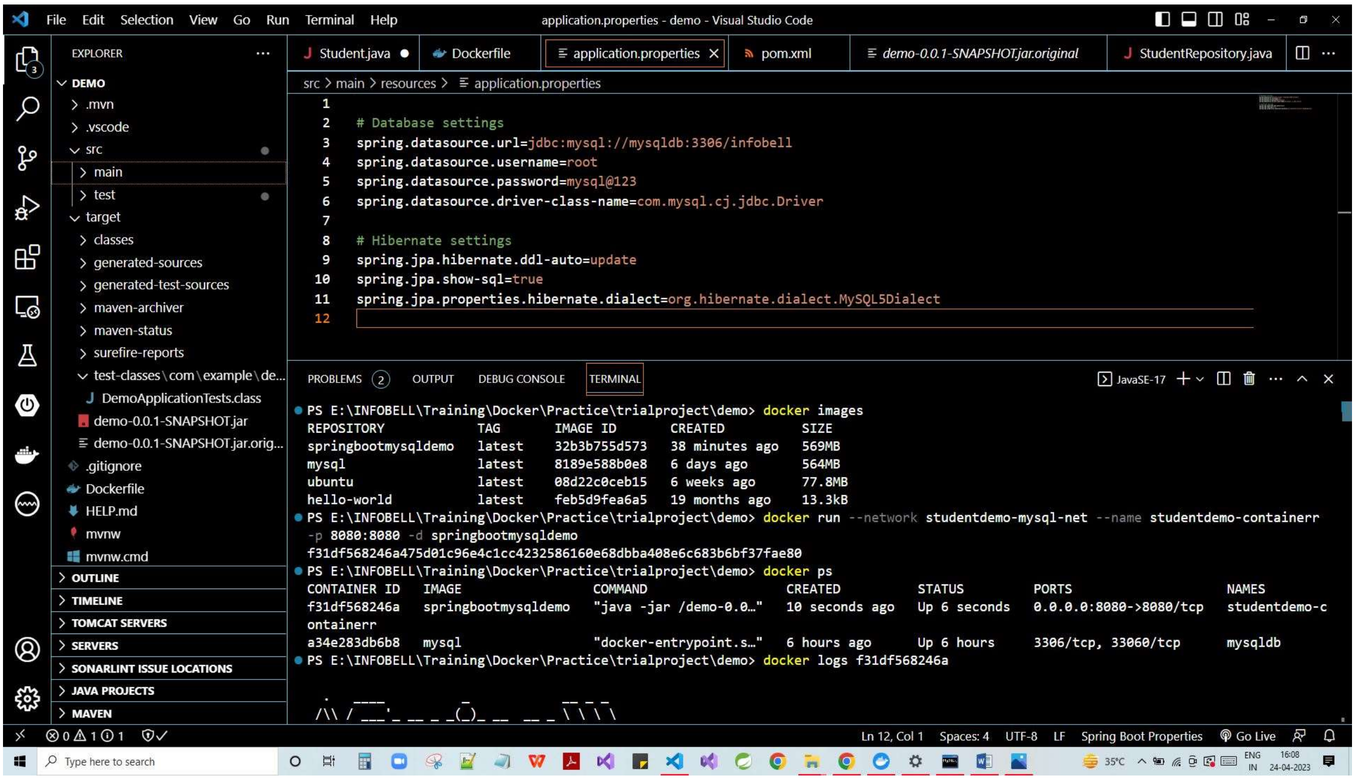
1.13 GB / 642.17 MB in use 4 images

Last refresh: about 4 hours ago

Space to be reclaimed 568.74 MB

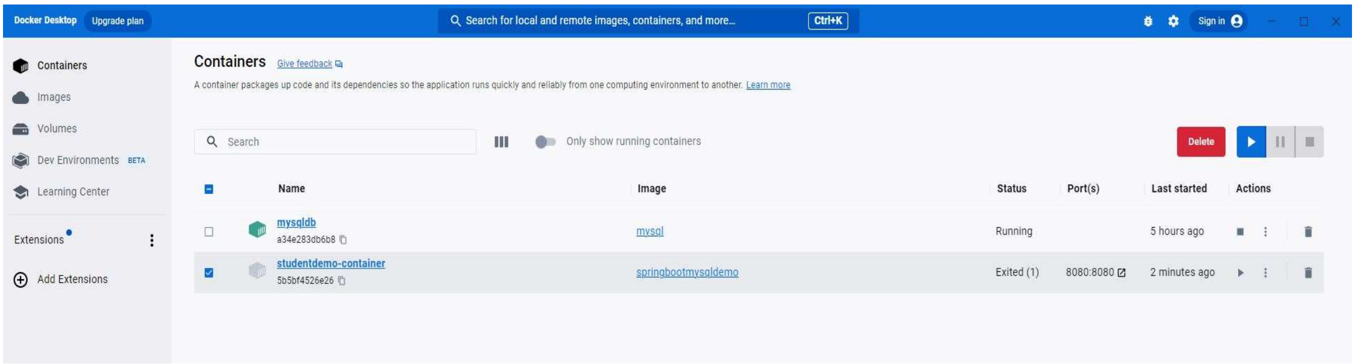
	Name	Tag	Status	Created	Size	Actions
☑	springbootmysqldemo 32b3b755d573	latest	In use	22 minutes ago	568.74 MB	
□	mysql 8189e588b0e8	latest	In use	6 days ago	564.34 MB	
□	ubuntu 08d22c0ceb15	latest	Unused	about 2 months ago	77.81 MB	
□	hello-world feb5d9fea6a5	latest	Unused	over 1 year ago	13.25 KB	

// Created a container as studentdemo-containerr using springbootmysqldemo image



```

# Database settings
spring.datasource.url=jdbc:mysql://mysqldb:3306/infobell
spring.datasource.username=root
spring.datasource.password=mysql@123
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

```

```
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker images
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
springbootmysqldemo   latest    32b3b755d573   38 minutes ago   569MB
mysql                 latest    8189e588b0e8   6 days ago       564MB
ubuntu                latest    08d22c0ceb15   6 weeks ago      77.8MB
hello-world           latest    feb5d9fea6a5   19 months ago    13.3kB
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker run --network studentdemo-mysql-net --name studentdemo-containerr
-p 8080:8080 -d springbootmysqldemo
f31df568246a475d01c96e4c1cc4232586160e68dbba408e6c683b6bf37fae80
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker ps
CONTAINER ID   IMAGE                 COMMAND                  CREATED          STATUS          PORTS                    NAMES
f31df568246a   springbootmysqldemo   "java -jar /demo-0.0…"   10 seconds ago   Up 6 seconds    0.0.0.0:8080->8080/tcp   studentdemo-c
ontainerr
a34e283db6b8   mysql                 "docker-entrypoint.s…"   6 hours ago      Up 6 hours      3306/tcp, 33060/tcp      mysqldb
PS E:\INFOBELL\Training\Docker\Practice\trialproject\demo> docker logs f31df568246a

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
```



Containers Give feedback

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

	Name	Image	Status	Port(s)	Last started	Actions
□	mysqldb a34e283db6b8	mysql	Running		5 hours ago	
☑	studentdemo-container 5b5bf4526e26	springbootmysqldemo	Exited (1)	8080:8080	2 minutes ago	