```
#!/bin/sh
# ----------------------------------------------------------------------------
#
# under one
#
# NOTICE file
# distributed with this work for additional information
#
# this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#    https://www.apache.org/licenses/LICENSE-2.0
#
# writing,
# software distributed under the License is distributed on an
# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
# KIND, either express or implied.  See the License for the
#
# limitations
# under the License.
# ----------------------------------------------------------------------------
#
# ----------------------------------------------------------------------------
# Maven Start Up Batch script
#
# Required ENV vars:
# ------------------
#   JAVA_HOME - location of a JDK home dir
#
```

```
28 # Optional ENV vars
29 # -----------------
30 #   M2_HOME - location of maven2's installed home dir
31 #   MAVEN_OPTS - parameters passed to the Java VM
   when running Maven
32 #     e.g. to debug Maven itself, use
33 #       set MAVEN_OPTS=-Xdebug -Xrunjdwp:transport=
   dt_socket,server=y,suspend=y,address=8000
34 #   MAVEN_SKIP_RC - flag to disable loading of
   mavenrc files
35 # ----------------------------------------------------
   ------------------------
36
37 if [ -z "$MAVEN_SKIP_RC" ] ; then
38
39   if [ -f /usr/local/etc/mavenrc ] ; then
40     . /usr/local/etc/mavenrc
41   fi
42
43   if [ -f /etc/mavenrc ] ; then
44     . /etc/mavenrc
45   fi
46
47   if [ -f "$HOME/.mavenrc" ] ; then
48     . "$HOME/.mavenrc"
49   fi
50
51 fi
52
53
   true or false.
54 cygwin=false;
55 darwin=false;
56 mingw=false
57 case "`uname`" in
58   CYGWIN*) cygwin=true ;;
59   MINGW*) mingw=true;;
60   Darwin*) darwin=true
61     # Use /usr/libexec/java_home if available,
   otherwise fall back to /Library/Java/Home
62
```

```
qa1170/_index.html
    if [ -z "$JAVA_HOME" ]; then
      if [ -x "/usr/libexec/java_home" ]; then
        export JAVA_HOME="`/usr/libexec/java_home`"
      else
        export JAVA_HOME="/Library/Java/Home"
      fi
    fi
    ;;
esac

if [ -z "$JAVA_HOME" ] ; then
  if [ -r /etc/gentoo-release ] ; then
    JAVA_HOME=`java-config --jre-home`
  fi
fi

if [ -z "$M2_HOME" ] ; then
  ## resolve links - $0 may be a link to maven's
home
  PRG="$0"

  # need this for relative symlinks
  while [ -h "$PRG" ] ; do
    ls=`ls -ld "$PRG"`
    link=`expr "$ls" : '.*-> \(.*\)$'`
    if expr "$link" : '/.*' > /dev/null; then
      PRG="$link"
    else
      PRG="`dirname "$PRG"`/$link"
    fi
  done

  saveddir=`pwd`

  M2_HOME=`dirname "$PRG"`/..

  # make it fully qualified
  M2_HOME=`cd "$M2_HOME" && pwd`

  cd "$saveddir"
```

```
102   # echo Using m2 at $M2_HOME
103 fi
104
105
     anything is touched
106 if $cygwin ; then
107   [ -n "$M2_HOME" ] &&
108     M2_HOME=`cygpath --unix "$M2_HOME"`
109   [ -n "$JAVA_HOME" ] &&
110     JAVA_HOME=`cygpath --unix "$JAVA_HOME"`
111   [ -n "$CLASSPATH" ] &&
112     CLASSPATH=`cygpath --path --unix "$CLASSPATH"`
113 fi
114
115 # For Mingw, ensure paths are in UNIX format before
    anything is touched
116 if $mingw ; then
117   [ -n "$M2_HOME" ] &&
118     M2_HOME="`(cd "$M2_HOME"; pwd)`"
119   [ -n "$JAVA_HOME" ] &&
120     JAVA_HOME="`(cd "$JAVA_HOME"; pwd)`"
121 fi
122
123 if [ -z "$JAVA_HOME" ]; then
124   javaExecutable="`which javac`"
125   if [ -n "$javaExecutable" ] && ! [ "`expr \"
    $javaExecutable\" : '\([^ ]*\)'`" = "no" ]; then
126     # readlink(1) is not available as standard on
    Solaris 10.
127     readLink=`which readlink`
128     if [ ! `expr "$readLink" : '\([^ ]*\)'` = "no"
     ]; then
129       if $darwin ; then
130         javaHome="`dirname \"$javaExecutable\"`"
131         javaExecutable="`cd \"$javaHome\" && pwd -P`
    /javac"
132       else
133         javaExecutable="`readlink -f \"
    $javaExecutable\"`"
134       fi
135       javaHome="`dirname \"$javaExecutable\"`"
```

```
136       javaHome=`expr "$javaHome" : '\(.*\)/bin'`
137       JAVA_HOME="$javaHome"
138       export JAVA_HOME
139     fi
140   fi
141 fi
142
143 if [ -z "$JAVACMD" ] ; then
144   if [ -n "$JAVA_HOME"  ] ; then
145     if [ -x "$JAVA_HOME/jre/sh/java" ] ; then
146       # IBM's JDK on AIX uses strange locations for
    the executables
147       JAVACMD="$JAVA_HOME/jre/sh/java"
148     else
149       JAVACMD="$JAVA_HOME/bin/java"
150     fi
151   else
152     JAVACMD="`\\unset -f command; \\command -v java`
    "
153   fi
154 fi
155
156 if [ ! -x "$JAVACMD" ] ; then
157   echo "Error: JAVA_HOME is not defined correctly."
     >&2
158   echo "  We cannot execute $JAVACMD" >&2
159   exit 1
160 fi
161
162 if [ -z "$JAVA_HOME" ] ; then
163   echo "Warning: JAVA_HOME environment variable is
    not set."
164 fi
165
166
    .launcher.Launcher
167
168
    directory to filesystem root
169 # first directory with .mvn subdirectory is
    considered project base directory
```

```
170 find_maven_basedir() {
171
172   if [ -z "$1" ]
173   then
174     echo "Path not specified to find_maven_basedir"
175     return 1
176   fi
177
178   basedir="$1"
179   wdir="$1"
180   while [ "$wdir" != '/' ] ; do
181     if [ -d "$wdir"/.mvn ] ; then
182       basedir=$wdir
183       break
184     fi
185     # workaround for JBEAP-8937 (on Solaris 10/Sparc
    )
186     if [ -d "${wdir}" ]; then
187       wdir=`cd "$wdir/.."; pwd`
188     fi
189     # end of workaround
190   done
191   echo "${basedir}"
192 }
193
194 # concatenates all lines of a file
195 concat_lines() {
196   if [ -f "$1" ]; then
197     echo "$(tr -s '\n' ' ' < "$1")"
198   fi
199 }
200
201 BASE_DIR=`find_maven_basedir "$(pwd)"`
202 if [ -z "$BASE_DIR" ]; then
203   exit 1;
204 fi
205
206

207
    maven-wrapper.jar from Maven-central
```

```
208 # This allows using the maven wrapper in projects
    that prohibit checking in binary data.
209
210 if [ -r "$BASE_DIR/.mvn/wrapper/maven-wrapper.jar"
     ]; then
211     if [ "$MVNW_VERBOSE" = true ]; then
212       echo "Found .mvn/wrapper/maven-wrapper.jar"
213     fi
214 else
215     if [ "$MVNW_VERBOSE" = true ]; then
216       echo "Couldn't find .mvn/wrapper/maven-wrapper
    .jar, downloading it ..."
217     fi
218     if [ -n "$MVNW_REPOURL" ]; then
219       jarUrl="$MVNW_REPOURL/org/apache/maven/wrapper
    /maven-wrapper/3.1.0/maven-wrapper-3.1.0.jar"
220     else
221       jarUrl="https://repo.maven.apache.org/maven2/

    wrapper-3.1.0.jar"
222     fi
223     while IFS="=" read key value; do
224       case "$key" in (wrapperUrl) jarUrl="$value";
    break ;;
225       esac
226     done < "$BASE_DIR/.mvn/wrapper/maven-wrapper.
    properties"
227     if [ "$MVNW_VERBOSE" = true ]; then
228       echo "Downloading from: $jarUrl"
229     fi
230     wrapperJarPath="$BASE_DIR/.mvn/wrapper/maven-
    wrapper.jar"
231     if $cygwin; then
232       wrapperJarPath=`cygpath --path --windows "
    $wrapperJarPath"`
233     fi
234
235     if command -v wget > /dev/null; then
236         if [ "$MVNW_VERBOSE" = true ]; then
237           echo "Found wget ... using wget"
```

```
238         fi
239         if [ -z "$MVNW_USERNAME" ] || [ -z "
    $MVNW_PASSWORD" ]; then
240             wget "$jarUrl" -O "$wrapperJarPath" ||
    rm -f "$wrapperJarPath"
241         else
242             wget --http-user=$MVNW_USERNAME --http-
    password=$MVNW_PASSWORD "$jarUrl" -O "
    $wrapperJarPath" || rm -f "$wrapperJarPath"
243         fi
244     elif command -v curl > /dev/null; then
245         if [ "$MVNW_VERBOSE" = true ]; then
246           echo "Found curl ... using curl"
247         fi
248         if [ -z "$MVNW_USERNAME" ] || [ -z "
    $MVNW_PASSWORD" ]; then
249             curl -o "$wrapperJarPath" "$jarUrl" -f
250         else
251             curl --user $MVNW_USERNAME:
    $MVNW_PASSWORD -o "$wrapperJarPath" "$jarUrl" -f
252         fi
253
254     else
255         if [ "$MVNW_VERBOSE" = true ]; then
256           echo "Falling back to using Java to
    download"
257         fi
258         javaClass="$BASE_DIR/.mvn/wrapper/
    MavenWrapperDownloader.java"
259         # For Cygwin, switch paths to Windows format
     before running javac
260         if $cygwin; then
261           javaClass=`cygpath --path --windows "
    $javaClass"`
262         fi
263         if [ -e "$javaClass" ]; then
264             if [ ! -e "$BASE_DIR/.mvn/wrapper/
    MavenWrapperDownloader.class" ]; then
265                 if [ "$MVNW_VERBOSE" = true ]; then
266                   echo " - Compiling
    MavenWrapperDownloader.java ..."
```

```
                fi
                # Compiling the Java class
                ("$JAVA_HOME/bin/javac" "$javaClass")
            fi
            if [ -e "$BASE_DIR/.mvn/wrapper/MavenWrapperDownloader.class" ]; then
                # Running the downloader
                if [ "$MVNW_VERBOSE" = true ]; then
                  echo " - Running MavenWrapperDownloader.java ..."
                fi
                ("$JAVA_HOME/bin/java" -cp .mvn/wrapper MavenWrapperDownloader "$MAVEN_PROJECTBASEDIR")
            fi
        fi
    fi
fi

# End of extension


export MAVEN_PROJECTBASEDIR=${MAVEN_BASEDIR:-"$BASE_DIR"}
if [ "$MVNW_VERBOSE" = true ]; then
  echo $MAVEN_PROJECTBASEDIR
fi
MAVEN_OPTS="$(concat_lines "$MAVEN_PROJECTBASEDIR/.mvn/jvm.config") $MAVEN_OPTS"


# running java
if $cygwin; then
  [ -n "$M2_HOME" ] &&
    M2_HOME=`cygpath --path --windows "$M2_HOME"`
  [ -n "$JAVA_HOME" ] &&
    JAVA_HOME=`cygpath --path --windows "$JAVA_HOME"`
```

```
297   [ -n "$CLASSPATH" ] &&
298     CLASSPATH=`cygpath --path --windows "$CLASSPATH"
    `
299   [ -n "$MAVEN_PROJECTBASEDIR" ] &&
300     MAVEN_PROJECTBASEDIR=`cygpath --path --windows "
    $MAVEN_PROJECTBASEDIR"`
301 fi
302
303 # Provide a "standardized" way to retrieve the CLI
    args that will
304
305 MAVEN_CMD_LINE_ARGS="$MAVEN_CONFIG $@"
306 export MAVEN_CMD_LINE_ARGS
307
308 WRAPPER_LAUNCHER=org.apache.maven.wrapper.
    MavenWrapperMain
309
310 exec "$JAVACMD" \
311   $MAVEN_OPTS \
312   $MAVEN_DEBUG_OPTS \
313   -classpath "$MAVEN_PROJECTBASEDIR/.mvn/wrapper/
    maven-wrapper.jar" \
314   "-Dmaven.home=${M2_HOME}" \
315   "-Dmaven.multiModuleProjectDirectory=${
    MAVEN_PROJECTBASEDIR}" \
316   ${WRAPPER_LAUNCHER} $MAVEN_CONFIG "$@"
317
```

```
# Getting Started

### Reference Documentation

sections:

* [Official Apache Maven documentation](https://maven.apache.org/guides/index.html)
* [Spring Boot Maven Plugin Reference Guide](https://reference/html/)
* [Create an OCI image](https://docs.spring.io/spring-image)
* [Spring Boot DevTools](https://docs.spring.io/devtools)
* [Spring Web](https://docs.spring.io/spring-boot/docs/2.7.8/reference/htmlsingle/#web)
* [Spring Data JPA](https://docs.spring.io/spring-and-spring-data)

### Guides
The following guides illustrate how to use some features concretely:

* [Building a RESTful Web Service](https://spring.io/guides/gs/rest-service/)
* [Serving Web Content with Spring MVC](https://spring.io/guides/gs/serving-web-content/)
* [Building REST services with Spring](https://spring.io/guides/tutorials/rest/)
* [Accessing Data with JPA](https://spring.io/guides/gs/accessing-data-jpa/)
* [Accessing data with MySQL](https://spring.io/guides/gs/accessing-data-mysql/)


```

```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
                                                    >
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</
artifactId>
        <version>2.7.8</version>
        <relativePath/> <!-- lookup parent from
repository -->
    </parent>
    <groupId>com</groupId>
    <artifactId>demo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo</name>
    <description>Project03</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</
groupId>
            <artifactId>spring-boot-starter-data-jpa
</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</
groupId>
            <artifactId>spring-boot-starter-web</
artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</
groupId>
            <artifactId>spring-boot-devtools</
artifactId>
```

```
32              <scope>runtime</scope>
33              <optional>true</optional>
34          </dependency>
35          <dependency>
36              <groupId>com.mysql</groupId>
37              <artifactId>mysql-connector-j</artifactId
    >
38              <scope>runtime</scope>
39          </dependency>
40          <dependency>
41              <groupId>org.springframework.boot</
    groupId>
42              <artifactId>spring-boot-starter-test</
    artifactId>
43              <scope>test</scope>
44          </dependency>
45          <dependency>
46              <groupId>io.springfox</groupId>
47              <artifactId>springfox-boot-starter</
    artifactId>
48              <version>3.0.0</version>
49          </dependency>
50          <dependency>
51              <groupId>io.springfox</groupId>
52              <artifactId>springfox-swagger-ui</
    artifactId>
53              <version>3.0.0</version>
54          </dependency>
55      </dependencies>
56
57      <build>
58          <plugins>
59              <plugin>
60                  <groupId>org.springframework.boot</
    groupId>
61                  <artifactId>spring-boot-maven-plugin
    </artifactId>
62              </plugin>
63          </plugins>
64      </build>
65
```

```
@REM ----------------------------------------------------------------------------
@REM Licensed to the Apache Software Foundation (ASF) under one
@REM or more contributor license agreements.  See the NOTICE file
@REM distributed with this work for additional information
@REM this file
@REM to you under the Apache License, Version 2.0 (the
@REM "License"); you may not use this file except in compliance
@REM with the License.  You may obtain a copy of the License at
@REM
@REM    https://www.apache.org/licenses/LICENSE-2.0
@REM
@REM Unless required by applicable law or agreed to in writing,
@REM software distributed under the License is distributed on an
@REM "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
@REM KIND, either express or implied.  See the License for the
@REM limitations
@REM under the License.
@REM ----------------------------------------------------------------------------

@REM ----------------------------------------------------------------------------
@REM Maven Start Up Batch script
@REM
@REM Required ENV vars:
```

```
24 @REM JAVA_HOME - location of a JDK home dir
25 @REM
26 @REM Optional ENV vars
27 @REM M2_HOME - location of maven2's installed home
   dir
28 @REM MAVEN_BATCH_ECHO - set to 'on' to enable the
   echoing of the batch commands
29 @REM MAVEN_BATCH_PAUSE - set to 'on' to wait for a
   keystroke before ending
30 @REM MAVEN_OPTS - parameters passed to the Java VM
   when running Maven
31 @REM     e.g. to debug Maven itself, use
32 @REM set MAVEN_OPTS=-Xdebug -Xrunjdwp:transport=
   dt_socket,server=y,suspend=y,address=8000
33 @REM MAVEN_SKIP_RC - flag to disable loading of
   mavenrc files
34 @REM
    ----------------------------------------------------
   ----------------------
35
36 @REM Begin all REM lines with '@' in case
   MAVEN_BATCH_ECHO is 'on'
37 @echo off
38 @REM set title of command window
39 title %0
40 @REM enable echoing by setting MAVEN_BATCH_ECHO to '
   on'
41 @if "%MAVEN_BATCH_ECHO%" == "on"  echo %
   MAVEN_BATCH_ECHO%
42
43 @REM set %HOME% to equivalent of $HOME
44 if "%HOME%" == "" (set "HOME=%HOMEDRIVE%%HOMEPATH%")
45
46 @REM Execute a user defined script before this one
47 if not "%MAVEN_SKIP_RC%" == "" goto skipRcPre
48 @REM check for pre script, once with legacy .bat
   ending and once with .cmd ending
49 if exist "%USERPROFILE%\mavenrc_pre.bat" call "%
   USERPROFILE%\mavenrc_pre.bat" %*
50 if exist "%USERPROFILE%\mavenrc_pre.cmd" call "%
   USERPROFILE%\mavenrc_pre.cmd" %*
```

```
51 :skipRcPre
52
53 @setlocal
54
55 set ERROR_CODE=0
56
57 @REM
    scripts, we use another setlocal
58 @setlocal
59
60 @REM ==== START VALIDATION ====
61 if not "%JAVA_HOME%" == "" goto OkJHome
62
63 echo.
64 echo Error: JAVA_HOME not found in your environment.
   >&2
65 echo Please set the JAVA_HOME variable in your
   environment to match the >&2
66 echo location of your Java installation. >&2
67 echo.
68 goto error
69
70 :OkJHome
71 if exist "%JAVA_HOME%\bin\java.exe" goto init
72
73 echo.
74 echo Error: JAVA_HOME is set to an invalid directory
   . >&2
75 echo JAVA_HOME = "%JAVA_HOME%" >&2
76 echo Please set the JAVA_HOME variable in your
   environment to match the >&2
77 echo location of your Java installation. >&2
78 echo.
79 goto error
80
81 @REM ==== END VALIDATION ====
82
83 :init
84
85 @REM Find the project base dir, i.e. the directory
   that contains the folder ".mvn".
```

```
 86 @REM Fallback to current working directory if not
    found.
 87
 88 set MAVEN_PROJECTBASEDIR=%MAVEN_BASEDIR%
 89 IF NOT "%MAVEN_PROJECTBASEDIR%"=="" goto
    endDetectBaseDir
 90
 91 set EXEC_DIR=%CD%
 92 set WDIR=%EXEC_DIR%
 93 :findBaseDir
 94 IF EXIST "%WDIR%"\.mvn goto baseDirFound
 95 cd ..
 96 IF "%WDIR%"=="%CD%" goto baseDirNotFound
 97 set WDIR=%CD%
 98 goto findBaseDir
 99
100 :baseDirFound
101 set MAVEN_PROJECTBASEDIR=%WDIR%
102 cd "%EXEC_DIR%"
103 goto endDetectBaseDir
104
105 :baseDirNotFound
106 set MAVEN_PROJECTBASEDIR=%EXEC_DIR%
107 cd "%EXEC_DIR%"
108
109 :endDetectBaseDir
110
111 IF NOT EXIST "%MAVEN_PROJECTBASEDIR%\.mvn\jvm.config
    " goto endReadAdditionalConfig
112
113 @
114 for /F "usebackq delims=" %%a in ("%
    MAVEN_PROJECTBASEDIR%\.mvn\jvm.config") do set
    JVM_CONFIG_MAVEN_PROPS=!JVM_CONFIG_MAVEN_PROPS! %%a
115 @endlocal & set JVM_CONFIG_MAVEN_PROPS=%
    JVM_CONFIG_MAVEN_PROPS%
116
117 :endReadAdditionalConfig
118
119 SET MAVEN_JAVA_EXE="%JAVA_HOME%\bin\java.exe"
120 set WRAPPER_JAR="%MAVEN_PROJECTBASEDIR%\.mvn\wrapper
```

```
120 \maven-wrapper.jar"
121 set WRAPPER_LAUNCHER=org.apache.maven.wrapper.
    MavenWrapperMain
122
123 set DOWNLOAD_URL="https://repo.maven.apache.org/

    maven-wrapper-3.1.0.jar"
124
125 FOR /F "usebackq tokens=1,2 delims==" %%A IN ("%
    MAVEN_PROJECTBASEDIR%\.mvn\wrapper\maven-wrapper.
    properties") DO (
126     IF "%%A"=="wrapperUrl" SET DOWNLOAD_URL=%%B
127 )
128
129 @REM Extension to allow automatically downloading
    the maven-wrapper.jar from Maven-central
130 @REM This allows using the maven wrapper in projects
     that prohibit checking in binary data.
131 if exist %WRAPPER_JAR% (
132     if "%MVNW_VERBOSE%" == "true" (
133         echo Found %WRAPPER_JAR%
134     )
135 ) else (
136     if not "%MVNW_REPOURL%" == "" (
137         SET DOWNLOAD_URL="%MVNW_REPOURL%/org/apache/

    0.jar"
138     )
139     if "%MVNW_VERBOSE%" == "true" (
140         echo Couldn't find %WRAPPER_JAR%,
    downloading it ...
141         echo Downloading from: %DOWNLOAD_URL%
142     )
143
144     powershell -Command "&{"^
145         "$webclient = new-object System.Net.
    WebClient;"^
146         "if (-not ([string]::IsNullOrEmpty('%
    MVNW_USERNAME%') -and [string]::IsNullOrEmpty('%
    MVNW_PASSWORD%'))) {"^
147         "$webclient.Credentials = new-object System.
```

```
147 Net.NetworkCredential('%MVNW_USERNAME%', '%
    MVNW_PASSWORD%');"^
148         "}"^
149         "[Net.ServicePointManager]::SecurityProtocol
     = [Net.SecurityProtocolType]::Tls12; $webclient.
    DownloadFile('%DOWNLOAD_URL%', '%WRAPPER_JAR%')"^
150         "}"
151     if "%MVNW_VERBOSE%" == "true" (
152         echo Finished downloading %WRAPPER_JAR%
153     )
154 )
155 @REM End of extension
156
157 @REM Provide a "standardized" way to retrieve the
    CLI args that will
158 @REM work with both Windows and non-Windows
    executions.
159 set MAVEN_CMD_LINE_ARGS=%*
160
161 %MAVEN_JAVA_EXE% ^
162   %JVM_CONFIG_MAVEN_PROPS% ^
163   %MAVEN_OPTS% ^
164   %MAVEN_DEBUG_OPTS% ^
165   -classpath %WRAPPER_JAR% ^
166   "-Dmaven.multiModuleProjectDirectory=%
    MAVEN_PROJECTBASEDIR%" ^
167   %WRAPPER_LAUNCHER% %MAVEN_CONFIG% %*
168 if ERRORLEVEL 1 goto error
169 goto end
170
171 :error
172 set ERROR_CODE=1
173
174 :end
175 @endlocal & set ERROR_CODE=%ERROR_CODE%
176
177 if not "%MAVEN_SKIP_RC%"=="" goto skipRcPost
178 @REM check for post script, once with legacy .bat
    ending and once with .cmd ending
179 if exist "%USERPROFILE%\mavenrc_post.bat" call "%
    USERPROFILE%\mavenrc_post.bat"
```

```
HELP.md
target/
!.mvn/wrapper/maven-wrapper.jar
!**/src/main/**/target/
!**/src/test/**/target/

### STS ###
.apt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache

### IntelliJ IDEA ###
.idea
*.iws
*.iml
*.ipr

### NetBeans ###
/nbproject/private/
/nbbuild/
/dist/
/nbdist/
/.nb-gradle/
build/
!**/src/main/**/build/
!**/src/test/**/build/

### VS Code ###
.vscode/

```

```java
package com.demo.config;

import
import org.springframework.context.annotation.
Configuration;

import springfox.documentation.builders.
RequestHandlerSelectors;
import
import
Docket;

@Configuration
public class SwaggerConfig {

    @Bean
    public Docket api() {
        return new Docket(DocumentationType.SWAGGER_2

();
    }

}

```

```java
package com.demo.models;

import javax.persistence.*;

@Entity
@Table(name = "user")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "user_id")
    private Long userId;
    @Column(name = "user_name")
    private String userName;
    @Column(name = "user_email")
    private String userEmail;
    @Column(name = "user_pwd")
    private String userPwd;
    @Column(name = "user_contact")
    private Long userContact;

    public User() {
    }

    public User(String userName, String userEmail, String userPwd, Long userContact) {
        this.userName = userName;
        this.userEmail = userEmail;
        this.userPwd = userPwd;
        this.userContact = userContact;
    }

    public Long getUserId() {
        return userId;
    }

    public void setUserId(Long userId) {
        this.userId = userId;
    }

```

```
40      public String getUserName() {
41          return userName;
42      }
43
44      public void setUserName(String userName) {
45          this.userName = userName;
46      }
47
48      public String getUserEmail() {
49          return userEmail;
50      }
51
52      public void setUserEmail(String userEmail) {
53          this.userEmail = userEmail;
54      }
55
56      public String getUserPwd() {
57          return userPwd;
58      }
59
60      public void setUserPwd(String userPwd) {
61          this.userPwd = userPwd;
62      }
63
64      public Long getUserContact() {
65          return userContact;
66      }
67
68      public void setUserContact(Long userContact) {
69          this.userContact = userContact;
70      }
71
72  }
73
```

```java
package com.demo.models;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "admin")
public class Admin {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "admin_id")
    private Long adminId;

    @Column(name = "admin_name")
    private String name;
    @Column(name = "admin_email")
    private String email;
    @Column(name = "admin_password")
    private String pwd;
    @Column(name = "admin_contact")
    private String contact;

    public Admin() {
    }
    public Admin(String name, String email, String pwd, String contact) {
        this.name = name;
        this.email = email;
        this.pwd = pwd;
        this.contact = contact;
    }

    public Long getAdminId() {
        return adminId;
    }
```

```
40
41      public void setAdminId(Long adminId) {
42          this.adminId = adminId;
43      }
44
45      public String getName() {
46          return name;
47      }
48
49      public void setName(String name) {
50          this.name = name;
51      }
52
53      public String getEmail() {
54          return email;
55      }
56
57      public void setEmail(String email) {
58          this.email = email;
59      }
60
61      public String getPwd() {
62          return pwd;
63      }
64
65      public void setPwd(String pwd) {
66          this.pwd = pwd;
67      }
68
69      public String getContact() {
70          return contact;
71      }
72
73      public void setContact(String contact) {
74          this.contact = contact;
75      }
76
77  }
78
```

```java
package com.demo.models;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "product")
public class Product {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    @Column(name = "product_id")
    private Long productId;
    @Column(name = "product_category")
    private String category;
    @Column(name = "product_company")
    private String company;
    @Column(name = "product_size")
    private int size;
    @Column(name = "product_origin")
    private String origin;
    @Column(name = "product_price")
    private Float price;
    @Column(name = "product_tag")
    private int tag;

    public Product()    {
    }
    public Product(String category, String company,
int size, String origin, Float price, int tag) {
        this.category = category;
        this.company = company;
        this.size = size;
        this.origin = origin;
        this.price = price;
        this.tag = tag;
```

```
40      }
41
42      public Long getProductId() {
43          return productId;
44      }
45      public void setProductId(Long productId) {
46          this.productId = productId;
47      }
48
49      public String getCategory() {
50          return category;
51      }
52      public void setCategory(String category) {
53          this.category = category;
54      }
55
56      public String getCompany() {
57          return company;
58      }
59      public void setCompany(String company) {
60          this.company = company;
61      }
62
63
64      public int getSize() {
65          return size;
66      }
67      public void setSize(int size) {
68          this.size = size;
69      }
70
71      public String getOrigin() {
72          return origin;
73      }
74      public void setOrigin(String origin) {
75          this.origin = origin;
76      }
77
78      public Float getPrice() {
79          return price;
80      }
```

```
81      public void setPrice(Float price) {
82          this.price = price;
83      }
84
85      public int getTag() {
86          return tag;
87      }
88      public void setTag(int tag) {
89          this.tag = tag;
90      }
91  }
92
```

```java
package com.demo.models;

import javax.persistence.*;

import java.util.HashSet;
import java.util.Set;

@Entity
@Table(name = "customer")
public class Customer {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "customer_id")
    private Long customerId;
    @Column(name = "customer_name")
    private String customerName;
    @Column(name = "customer_email")
    private String email;
    @Column(name = "customer_contact")
    private Long contact;
    @Column(name = "purchased_date")
    private String date;
    @Column(name = "total_price")
    private Float totalPrice;

    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name = "customer_id")
    private Set<PurchasedProduct> list = new HashSet<>();

    public Customer() {
        totalPrice = (float) 0;
    }
    public Customer(String customerName, String email, Long contact, String date, Float totalPrice) {
        this.customerName = customerName;
        this.email = email;
        this.contact = contact;
        this.date = date;
```

```
39          this.totalPrice = totalPrice;
40      }
41
42      public Long getCustomerId() {
43          return customerId;
44      }
45
46      public void setCustomerId(Long customerId) {
47          this.customerId = customerId;
48      }
49
50      public String getCustomerName() {
51          return customerName;
52      }
53
54      public void setCustomerName(String customerName
    ) {
55          this.customerName = customerName;
56      }
57
58      public String getEmail() {
59          return email;
60      }
61
62      public void setEmail(String email) {
63          this.email = email;
64      }
65
66      public Long getContact() {
67          return contact;
68      }
69
70      public void setContact(Long contact) {
71          this.contact = contact;
72      }
73
74
75      public String getDate() {
76          return date;
77      }
78
```

```
79      public void setDate(String date) {
80          this.date = date;
81      }
82
83      public Float getTotalPrice() {
84          return totalPrice;
85      }
86
87      public void setTotalPrice(Float totalPrice) {
88          this.totalPrice = totalPrice;
89      }
90
91      public Set<PurchasedProduct> getList() {
92          return list;
93      }
94
95      public void setList(Set<PurchasedProduct> list
    ) {
96          this.list = list;
97      }
98
99  }
100
```

```java
package com.demo.models;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "purchased_product")
public class PurchasedProduct {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    @Column(name = "product_id")
    private Long productId;
    @Column(name = "product_category")
    private String category;
    @Column(name = "product_company")
    private String company;
    @Column(name = "product_size")
    private int size;
    @Column(name = "product_origin")
    private String origin;
    @Column(name = "product_price")
    private Float price;
    @Column(name = "product_tag")
    private int tag;

    public PurchasedProduct()    {
    }
    public PurchasedProduct(String category, String
company, int size, String origin, Float price, int
tag) {
        this.category = category;
        this.company = company;
        this.size = size;
        this.origin = origin;
        this.price = price;
```

```
39          this.tag = tag;
40      }
41
42      public Long getProductId() {
43          return productId;
44      }
45      public void setProductId(Long productId) {
46          this.productId = productId;
47      }
48
49      public String getCategory() {
50          return category;
51      }
52      public void setCategory(String category) {
53          this.category = category;
54      }
55
56      public String getCompany() {
57          return company;
58      }
59      public void setCompany(String company) {
60          this.company = company;
61      }
62
63      public int getSize() {
64          return size;
65      }
66      public void setSize(int size) {
67          this.size = size;
68      }
69
70      public String getOrigin() {
71          return origin;
72      }
73      public void setOrigin(String origin) {
74          this.origin = origin;
75      }
76
77      public Float getPrice() {
78          return price;
79      }
```

```
80      public void setPrice(Float price) {
81          this.price = price;
82      }
83
84      public int getTag() {
85          return tag;
86      }
87      public void setTag(int tag) {
88          this.tag = tag;
89      }
90
91  }
92
```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;
import com.demo.models.User;

import java.util.List;

public interface IUserService {

    public Customer insertProductinExistingUserInDB(Customer customer, Long userid, Long productid) throws ResourceNotFound;
    public User insertUserInDB(User user);
    public void updateUserInDB(User user,Long userId) throws ResourceNotFound;
    public List<User> getAllUsers();

}

```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Admin;

public interface IAdminService {

    public Admin insertAdminInDB(Admin admin);
    public void updateAdminInDB(Admin admin,Long adminId) throws ResourceNotFound;

}

```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Product;

import java.util.List;

public interface IProductService {

    // CRUD Operations
    public Product insertProductInDB(Product product
);
    public List<Product> getAllProducts();
    public void updateProductInDB(Product product,
Long productId) throws ResourceNotFound;
    public void deleteProductInDB(Long productId);
    public Product getProductInDB(Long productId)
throws ResourceNotFound;

}

```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;
import com.demo.models.Product;
import com.demo.models.PurchasedProduct;
import com.demo.models.User;
import com.demo.repository.CustomerRepository;
import com.demo.repository.ProductRepository;
import com.demo.repository.UserRepository;

import
Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Set;

@Service
public class UserServiceImpl implements  IUserService
 {

    @Autowired
    UserRepository userRepo;
    @Autowired
    ProductRepository productRepo;
    @Autowired
    CustomerRepository customerRepo;
    Float total = (float) 0;

    @Override
    public Customer insertProductinExistingUserInDB(
Customer customer, Long userid, Long product_id)
throws ResourceNotFound {
        Set<PurchasedProduct> list;
        User newUser = userRepo.findById(userid).get
();
        if((newUser != null) && productRepo.
existsById(product_id)) {
            total = (float) 0;
            Product prod = productRepo.findById(
```

```
35 product_id).get();
36             PurchasedProduct n1 = new
   PurchasedProduct();
37             Customer new1 = new Customer();
38             // Setting up Product Details
39             n1.setPrice(prod.getPrice());
40             n1.setOrigin(prod.getOrigin());
41             n1.setCategory(prod.getCategory());
42             n1.setCompany(prod.getCompany());
43             n1.setSize(prod.getSize());
44             n1.setTag(prod.getTag());
45             list = new1.getList();
46             list.add(n1);
47             // Setting up Customer Details
48             new1.setCustomerName(newUser.getUserName
   ());
49             new1.setEmail(newUser.getUserEmail());
50             new1.setContact(newUser.getUserContact
   ());
51             total += n1.getPrice();
52             new1.setDate(customer.getDate());
53             new1.setTotalPrice(total);
54             new1.setList(list);
55             return customerRepo.save(new1);
56         } else throw new ResourceNotFound("Not found"
   );
57     }
58
59     @Override
60     public User insertUserInDB(User user) {
61         return userRepo.save(user);
62     }
63
64     @Override
65     public void updateUserInDB(User user, Long userId
   ) throws ResourceNotFound {
66         User existingUser = userRepo.findById(userId
   ).get();
67         if(user != null) {
68             // Update existing Singed up User Details
    with new Details.
```

```
69              existingUser.setUserName(user.
    getUserName());
70              existingUser.setUserContact(user.
    getUserContact());
71              existingUser.setUserEmail(user.
    getUserEmail());
72              existingUser.setUserPwd(user.getUserPwd
    ());
73              userRepo.save(existingUser);
74          } else {
75              throw new ResourceNotFound("Not able to
    do Update");
76          }
77      }
78
79      @Override
80      public List<User> getAllUsers() {
81          return userRepo.findAll();
82      }
83 }
84
```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Admin;
import com.demo.repository.AdminRepository;
import com.demo.repository.CustomerRepository;

import
Autowired;
import org.springframework.stereotype.Service;

@Service
public class AdminServiceImpl implements
IAdminService {

    @Autowired
    private AdminRepository adminRepo;
    @Autowired
    private CustomerRepository custRepo;

    @Override
    public Admin insertAdminInDB(Admin admin) {
        return adminRepo.save(admin);
    }

    @Override
    public void updateAdminInDB(Admin admin, Long
adminId) throws ResourceNotFound {
        Admin existingAdmin = adminRepo.findById(
adminId).get();
        if(admin != null) {
            // Update existing Admin with Password.
            existingAdmin.setPwd(admin.getPwd());
            adminRepo.save(existingAdmin);
        } else {
            throw new ResourceNotFound("Customer not
found");
        }
    }


```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;

import java.util.List;

public interface ICustomerService {

    public Customer insertCustomerInDB(Customer customer, Long product_id) throws ResourceNotFound;
    public List<Customer> getAllCustomers();
    public void updateCustomerInDB(Customer customer, Long customerId) throws ResourceNotFound;
    public void deleteCustomerInDB(Long customerId);
    public Customer getCustomerInDB(Long customerId) throws ResourceNotFound;
    public Customer

    , Long customer_id, Long product_id) throws ResourceNotFound;

    // Custom
    public List<Customer> findByDate(String date);
    public List<Customer> findByCustomerName(String customerName);
}

```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Product;
import com.demo.repository.ProductRepository;

import
Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class ProductServiceImpl implements
IProductService  {

    @Autowired
    private ProductRepository productRepo;

    @Override
    public Product insertProductInDB(Product product
) {
        return productRepo.save(product);
    }

    @Override
    public List<Product> getAllProducts() {
        return productRepo.findAll();
    }

    @Override
    public void updateProductInDB(Product product,
Long productId) throws ResourceNotFound {
        Product existingProduct = productRepo.
findById(productId).get();
        if(product != null) {
            // Update existing Product Details with
new Details.
            existingProduct.setCategory(product.
getCategory());
            existingProduct.setCompany(product.
```

```
34 getCompany());
35                 existingProduct.setOrigin(product.
   getOrigin());
36                 existingProduct.setPrice(product.getPrice
   ());
37                 existingProduct.setPrice(product.getPrice
   ());
38                 existingProduct.setTag(product.getTag());
39                 productRepo.save(existingProduct);
40             }else {
41                 throw new ResourceNotFound("Product not
   found");
42             }
43         }
44
45         @Override
46         public void deleteProductInDB(Long productId) {
47             productRepo.deleteById(productId);
48         }
49
50         @Override
51         public Product getProductInDB(Long productId)
   throws ResourceNotFound {
52             return productRepo.findById(productId).get();
53         }
54 }
55
```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;
import com.demo.models.Product;
import com.demo.models.PurchasedProduct;
import com.demo.repository.CustomerRepository;
import com.demo.repository.ProductRepository;

import
Autowired;
import org.springframework.stereotype.Service;

import java.util.HashSet;
import java.util.List;
import java.util.Set;

@Service
public class CustomerServiceImpl implements
ICustomerService {

    @Autowired
    private CustomerRepository customerRepo;
    @Autowired
    private ProductRepository productRepo;
    float total = 0;

    @Override
    public Customer insertCustomerInDB(Customer
customer,Long product_id) throws ResourceNotFound {
        total = customer.getTotalPrice();
        Set<PurchasedProduct> list = new HashSet<>();
        if(productRepo.existsById(product_id)) {
            Product prod = productRepo.findById(
product_id).get();
            PurchasedProduct n1 = new
PurchasedProduct();
            n1.setPrice(prod.getPrice());
            n1.setOrigin(prod.getOrigin());
            n1.setCategory(prod.getCategory());
            n1.setCompany(prod.getCompany());
```

```
70          customerRepo.deleteById(customerId);
71      }
72
73      @Override
74      public Customer getCustomerInDB(Long customerId
    ) throws ResourceNotFound {
75          return customerRepo.findById(customerId).get
    ();
76      }
77
78      @Override
79      public Customer

    customer, Long customer_id, Long product_id) throws
    ResourceNotFound {
80
81          Set<PurchasedProduct> list;
82          Customer new1 = customerRepo.findById(
    customer_id).get();
83          if((new1 != null) && productRepo.existsById(
    product_id)) {
84              total = new1.getTotalPrice();
85              Product prod = productRepo.findById(
    product_id).get();
86              PurchasedProduct n1 = new
    PurchasedProduct();
87              customerRepo.deleteById(customer_id);
88              n1.setPrice(prod.getPrice());
89              n1.setOrigin(prod.getOrigin());
90              n1.setCategory(prod.getCategory());
91              n1.setCompany(prod.getCompany());
92              n1.setSize(prod.getSize());
93              n1.setTag(prod.getTag());
94              list = new1.getList();
95              list.add(n1);
96              total += prod.getPrice();
97              new1.setDate(customer.getDate());
98              new1.setTotalPrice(total);
99              new1.setList(list);
100             return customerRepo.save(new1);
101         } else throw new ResourceNotFound("Not found
```

```
101 ");
102     }
103 
104     //Custom
105     @Override
106     public List<Customer> findByDate(String date) {
107         return customerRepo.findByDate(date);
108     }
109 
110     @Override
111     public List<Customer> findByCustomerName(String
    customerName) {
112         return customerRepo.findByCustomerName(
    customerName);
113     }
114 }
115 
```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.PurchasedProduct;
import
;

import
Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class PurchasedProductImpl implements
IPurchasedProductService {

    @Autowired
    private PurchasedProductRepository
purchasedproductRepo;

    @Override
    public PurchasedProduct

) {
        return purchasedproductRepo.save(product);
    }

    @Override
    public List<PurchasedProduct>
getAllPurchasedProducts() {
        return purchasedproductRepo.findAll();
    }

    @Override
    public void updatePurchasedProductInDB(
PurchasedProduct product, Long productId) throws
ResourceNotFound {
        PurchasedProduct existingProduct =

        if(product != null) {
```

```
32              // Update existing Product Details with
    new Details.
33              existingProduct.setCategory(product.
    getCategory());
34              existingProduct.setCompany(product.
    getCompany());
35              existingProduct.setOrigin(product.
    getOrigin());
36              existingProduct.setPrice(product.getPrice
    ());
37              existingProduct.setPrice(product.getPrice
    ());
38              existingProduct.setTag(product.getTag());
39
    );
40          }else {
41              throw new ResourceNotFound("Product not
    found");
42          }
43      }
44
45      @Override
46      public void deletePurchasedProductInDB(Long
    productId) {
47
48      }
49
50      @Override
51      public PurchasedProduct getPurchasedProductInDB(
    Long productId) throws ResourceNotFound {
52          return purchasedproductRepo.findById(
    productId).get();
53      }
54
55      @Override
56      public List<PurchasedProduct> findByCategory(
    String category) {
57          return purchasedproductRepo.findByCategory(
    category);
58      }
59
```

```java
package com.demo.services;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.PurchasedProduct;
import org.springframework.data.repository.query.
Param;

import java.util.List;

public interface IPurchasedProductService {

    // CRUD Operations
    public PurchasedProduct

    public List<PurchasedProduct>
getAllPurchasedProducts();
    public void updatePurchasedProductInDB(
PurchasedProduct product,Long productId) throws
ResourceNotFound;
    public void deletePurchasedProductInDB(Long
productId);
    public PurchasedProduct getPurchasedProductInDB(
Long productId) throws ResourceNotFound;
    public List<PurchasedProduct> findByCategory(
String category);
    public List<PurchasedProduct> findByCustomerId(@
Param("id") Long id);

}

```

```java
package com.demo.repository;

import com.demo.models.Admin;

import org.springframework.data.jpa.repository.
JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface AdminRepository extends
JpaRepository<Admin, Long> {
}

```

```java
package com.demo.repository;

import com.demo.models.PurchasedProduct;

import org.springframework.data.jpa.repository.JpaRepository;
import
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface PurchasedProductRepository extends JpaRepository<PurchasedProduct, Long> {

    public List<PurchasedProduct> findByCategory(String category);

    // Complex Queries.
    @Query("from PurchasedProduct where customer_id=:id")
    public List<PurchasedProduct> findByCustomerId(@Param("id") Long id);

}

```

```java
package com.demo.controllers;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;
import com.demo.models.PurchasedProduct;
import com.demo.models.User;
import com.demo.services.ICustomerService;
import
import com.demo.services.IUserService;

import
Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

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

    @Autowired
    private IUserService userService;
    @Autowired
    private IPurchasedProductService prodService;
    @Autowired
    private ICustomerService customerService;

    @PostMapping("/signup")
    public User insertUser(@RequestBody User newUser
) {
        return userService.insertUserInDB(newUser);
    }

    @PostMapping("/insertproductinuser/{userid}/{
prodid}")
    public Customer insertProductUserInDB(@
                                          "userid"
) Long userid, @PathVariable("prodid") Long prodid)
throws ResourceNotFound {
        return userService.
```

```
34 prodid);
35     }
36
37     @PutMapping("/updateuserbyid/{userid}")
38     public void updateUser(@PathVariable("userid")
   Long userId, @RequestBody User user) throws
   ResourceNotFound {
39         userService.updateUserInDB(user, userId);
40     }
41
42     @PutMapping("/addmoreuserproducts/{custid}/{
   prodid}")
43     public void updateProductInCustomer(@PathVariable
   ("custid") Long userId, @PathVariable("prodid") Long
   prodid, @RequestBody Customer user) throws
   ResourceNotFound {
44         customerService.

   prodid);
45     }
46
47     @GetMapping("/getproducts/{userid}")
48     public List<PurchasedProduct> getproducts(@
   PathVariable("userid") Long id)    {
49         return prodService.findByCustomerId(id);
50     }
51
52 }
53
```

```java
package com.demo.controllers;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Admin;
import com.demo.models.Customer;
import com.demo.models.PurchasedProduct;
import com.demo.models.User;
import com.demo.services.*;

import
Autowired;
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.
PutMapping;
import org.springframework.web.bind.annotation.
RequestBody;
import org.springframework.web.bind.annotation.
RestController;
import org.springframework.web.bind.annotation.
RequestMapping;

import java.util.List;

@RestController
@RequestMapping("/admin")
public class AdminController {

    @Autowired
    private IAdminService adminService;
    @Autowired
    private ICustomerService custService;
    @Autowired
    private IPurchasedProductService prodService;
    @Autowired
    private IUserService userService;

```

```
34      @PostMapping("/insertadmin")
35      public Admin insertAdmin(@RequestBody Admin admin
    )   {
36          return adminService.insertAdminInDB(admin);
37      }
38
39      @PutMapping("/updateadminbyid/{adminid}")
40      public void updateAdmin(@PathVariable("adminid")
    Long adminId, @RequestBody Admin admin) throws
    ResourceNotFound {
41
42      }
43
44      @GetMapping("/getallvalidcustomers")
45      public List<Customer> getAllUsers(){
46          return custService.getAllCustomers();
47      }
48      @GetMapping("/getallcustomerbyname/{custname}")
49      public List<Customer> findByCustomerName(@
    PathVariable("custname") String customerName){
50          return custService.findByCustomerName(
    customerName);
51      }
52      @GetMapping("/getallcustomerbydate/{date}")
53      public List<Customer>
                                                    "date"
    ) String date) {
54          return custService.findByDate(date);
55      }
56
57      @GetMapping("/getallproductsbycategory/{category
    }")
58      public List<PurchasedProduct>
    getAllProductssByCategory(@PathVariable("category")
    String category) {
59          return prodService.findByCategory(category);
60      }
61
62      @GetMapping("/getallusers")
63      public List<User> getAllUsers1()    { return
    userService.getAllUsers(); }
```

```java
package com.demo.controllers;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Product;
import com.demo.services.IProductService;

import
Autowired;
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.
PutMapping;
import org.springframework.web.bind.annotation.
RequestBody;
import org.springframework.web.bind.annotation.
RestController;
import org.springframework.web.bind.annotation.
RequestMapping;

import java.util.List;

@RestController
@RequestMapping("/product")
public class ProductController {

    @Autowired
    private IProductService productService;

    @PostMapping("/insertproduct")
    public Product insertProduct(@RequestBody Product
 newProduct)    {
        return productService.insertProductInDB(
newProduct);
    }

```

```
31      @GetMapping("/getallproducts")
32      public List<Product> getAllProducts(){
33          return productService.getAllProducts();
34      }
35
36      @GetMapping("/getproduct/{productid}")
37      public Product getProduct(@PathVariable("
    productid") Long productId) throws ResourceNotFound {
38          return productService.getProductInDB(
    productId);
39      }
40
41      @DeleteMapping("/deleteproduct/{productid}")
42      public void deleteProduct(@PathVariable("
    productid") Long productId) {
43          productService.deleteProductInDB(productId);
44      }
45
46      @PutMapping("/updateproductbyid/{productid}")
47      public void updateProduct(@PathVariable("
    productid") Long productId, @RequestBody Product
    product) throws ResourceNotFound {
48          productService.updateProductInDB(product,
    productId);
49      }
50
51  }
52
```

```java
package com.demo.controllers;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.Customer;
import com.demo.services.ICustomerService;

import
Autowired;
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.
PutMapping;
import org.springframework.web.bind.annotation.
RequestBody;
import org.springframework.web.bind.annotation.
RestController;
import org.springframework.web.bind.annotation.
RequestMapping;

import java.util.List;

@RestController
@RequestMapping("/customer")
public class CustomerController {

    @Autowired
    private ICustomerService customerService;

    @PostMapping("/insertcustomer/{id}")
    public Customer insertCustomer(@RequestBody
Customer newUser, @PathVariable("id") Long id) throws
 ResourceNotFound {
        return customerService.insertCustomerInDB(
newUser, id);
    }
```

```
30
31      @GetMapping("/getallcustomers")
32      public List<Customer> getAllCustomers(){
33          return customerService.getAllCustomers();
34      }
35
36      @GetMapping("/getcustomer/{userid}")
37      public Customer getCustomer(@PathVariable("userid
    ") Long customerId) throws ResourceNotFound {
38          return customerService.getCustomerInDB(
    customerId);
39      }
40
41      @DeleteMapping("/deletecustomer/{userid}")
42      public void deleteCustomer(@PathVariable("userid"
    ) Long customerId) {
43
    );
44      }
45
46      @PutMapping("/updatecustomerbyid/{userid}")
47      public void updateCustomer(@PathVariable("userid"
    ) Long userId, @RequestBody Customer user) throws
    ResourceNotFound {
48          customerService.updateCustomerInDB(user,
    userId);
49      }
50
51      @PutMapping("/addcustomerproducts/{custid}/{
    prodid}")
52      public void updateProductInCustomer(@PathVariable
    ("custid") Long userId, @PathVariable("prodid") Long
    prodid, @RequestBody Customer user) throws
    ResourceNotFound {
53          customerService.

    prodid);
54      }
55
56 }
57
```

```java
package com.demo.controllers;

import com.demo.exceptions.ResourceNotFound;
import com.demo.models.PurchasedProduct;
import

import
Autowired;
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.
PutMapping;
import org.springframework.web.bind.annotation.
RequestBody;
import org.springframework.web.bind.annotation.
RestController;
import org.springframework.web.bind.annotation.
RequestMapping;

import java.util.List;

@RestController
@RequestMapping("/purchased")
public class PurchasedProductController {

    @Autowired
    private IPurchasedProductService
purchasedproductService;

    @PostMapping("/insertpurchasedproduct")
    public PurchasedProduct insertProduct(@
RequestBody PurchasedProduct newProduct){
        return purchasedproductService.
insertPurchasedProductInDB(newProduct);
    }
```

```
#connect to MySQL database
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url = jdbc:mysql://localhost:3306/

allowPublicKeyRetrieval=true&useSSL=false
spring.datasource.username = root
spring.datasource.password = ASD123qwe!@#

#Hibernate Properties

for the chosen database
spring.jpa.properties.hibernate.hibernate = org.hibernate.dialect.MySQL5InnoDBDialect


update)
#spring.jpa.hibernate.ddl-auto = update
spring.jpa.hibernate.ddl-auto = create
spring.jpa.properties.hibernate.show_sql=true
                                                =true
spring.jpa.properties.hibernate.format_sql=true

#Integrating Swagger - Documentation.
spring.mvc.pathmatch.matching-strategy = ANT_PATH_MATCHER


```

```
#connect to MySQL database
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url = jdbc:mysql://localhost:3306/

allowPublicKeyRetrieval=true&useSSL=false
spring.datasource.username = root
spring.datasource.password = ASD123qwe!@#

#Hibernate Properties

for the chosen database
spring.jpa.properties.hibernate.hibernate = org.hibernate.dialect.MySQL5InnoDBDialect


update)
#spring.jpa.hibernate.ddl-auto = update
spring.jpa.hibernate.ddl-auto = create
spring.jpa.properties.hibernate.show_sql=true
                                                =true
spring.jpa.properties.hibernate.format_sql=true

#Integrating Swagger - Documentation.
spring.mvc.pathmatch.matching-strategy = ANT_PATH_MATCHER


```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo;
6
7 @org.springframework.boot.autoconfigure.
  SpringBootApplication
8 public class DemoApplication {
9     public DemoApplication() { /* compiled code */ }
10
11    public static void main(java.lang.String[] args
  ) { /* compiled code */ }
12 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.models;
6
7 @javax.persistence.Entity
8 @javax.persistence.Table(name = "user")
9 public class User {
10     @javax.persistence.Id
11
  javax.persistence.GenerationType.IDENTITY)
12     @javax.persistence.Column(name = "user_id")
13     private java.lang.Long userId;
14     @javax.persistence.Column(name = "user_name")
15     private java.lang.String userName;
16     @javax.persistence.Column(name = "user_email")
17     private java.lang.String userEmail;
18     @javax.persistence.Column(name = "user_pwd")
19     private java.lang.String userPwd;
20     @javax.persistence.Column(name = "user_contact")
21     private java.lang.Long userContact;
22
23     public User() { /* compiled code */ }
24
25     public User(java.lang.String userName, java.lang.
  .Long userContact) { /* compiled code */ }
26
27     public java.lang.Long getUserId() { /* compiled
  code */ }
28
29     public void setUserId(java.lang.Long userId) {
  /* compiled code */ }
30
31     public java.lang.String getUserName() { /*
  compiled code */ }
32
33     public void setUserName(java.lang.String userName
  ) { /* compiled code */ }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.models;
6
7 @javax.persistence.Entity
8 @javax.persistence.Table(name = "admin")
9 public class Admin {
10     @javax.persistence.Id
11
  javax.persistence.GenerationType.IDENTITY)
12     @javax.persistence.Column(name = "admin_id")
13     private java.lang.Long adminId;
14     @javax.persistence.Column(name = "admin_name")
15     private java.lang.String name;
16     @javax.persistence.Column(name = "admin_email")
17     private java.lang.String email;
18     @javax.persistence.Column(name = "admin_password"
  )
19     private java.lang.String pwd;
20     @javax.persistence.Column(name = "admin_contact")
21     private java.lang.String contact;
22
23     public Admin() { /* compiled code */ }
24
25     public Admin(java.lang.String name, java.lang.
  contact) { /* compiled code */ }
26
27     public java.lang.Long getAdminId() { /* compiled
  code */ }
28
29     public void setAdminId(java.lang.Long adminId) {
  /* compiled code */ }
30
31     public java.lang.String getName() { /* compiled
  code */ }
32
33     public void setName(java.lang.String name) { /*
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.models;
6
7 @javax.persistence.Entity
8 @javax.persistence.Table(name = "product")
9 public class Product {
10     @javax.persistence.Id
11
  javax.persistence.GenerationType.IDENTITY)
12     @javax.persistence.Column(name = "product_id")
13     private java.lang.Long productId;
14     @javax.persistence.Column(name = "
  product_category")
15     private java.lang.String category;
16     @javax.persistence.Column(name = "product_company
  ")
17     private java.lang.String company;
18     @javax.persistence.Column(name = "product_size")
19     private int size;
20     @javax.persistence.Column(name = "product_origin"
  )
21     private java.lang.String origin;
22     @javax.persistence.Column(name = "product_price")
23     private java.lang.Float price;
24     @javax.persistence.Column(name = "product_tag")
25     private int tag;
26
27     public Product() { /* compiled code */ }
28
29     public Product(java.lang.String category, java.
  lang.String company, int size, java.lang.String
  origin, java.lang.Float price, int tag) { /* compiled
   code */ }
30
31     public java.lang.Long getProductId() { /*
  compiled code */ }
32
```

```
33      public void setProductId(java.lang.Long productId
    ) { /* compiled code */ }
34
35      public java.lang.String getCategory() { /*
    compiled code */ }
36
37      public void setCategory(java.lang.String category
    ) { /* compiled code */ }
38
39      public java.lang.String getCompany() { /*
    compiled code */ }
40
41      public void setCompany(java.lang.String company
    ) { /* compiled code */ }
42
43      public int getSize() { /* compiled code */ }
44
45      public void setSize(int size) { /* compiled code
     */ }
46
47      public java.lang.String getOrigin() { /* compiled
     code */ }
48
49      public void setOrigin(java.lang.String origin) {
    /* compiled code */ }
50
51      public java.lang.Float getPrice() { /* compiled
    code */ }
52
53      public void setPrice(java.lang.Float price) { /*
    compiled code */ }
54
55      public int getTag() { /* compiled code */ }
56
57      public void setTag(int tag) { /* compiled code */
     }
58 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.models;
6
7 @javax.persistence.Entity
8 @javax.persistence.Table(name = "customer")
9 public class Customer {
10     @javax.persistence.Id
11
  javax.persistence.GenerationType.IDENTITY)
12     @javax.persistence.Column(name = "customer_id")
13     private java.lang.Long customerId;
14     @javax.persistence.Column(name = "customer_name")
15     private java.lang.String customerName;
16     @javax.persistence.Column(name = "customer_email"
  )
17     private java.lang.String email;
18     @javax.persistence.Column(name = "
  customer_contact")
19     private java.lang.Long contact;
20     @javax.persistence.Column(name = "purchased_date"
  )
21     private java.lang.String date;
22     @javax.persistence.Column(name = "total_price")
23     private java.lang.Float totalPrice;
24
  persistence.CascadeType.ALL})
25     @javax.persistence.JoinColumn(name = "customer_id
  ")
26     private java.util.Set<com.demo.models.
  PurchasedProduct> list;
27
28     public Customer() { /* compiled code */ }
29
30     public Customer(java.lang.String customerName,
  lang.String date, java.lang.Float totalPrice) { /*
  compiled code */ }
```

```
31
32      public java.lang.Long getCustomerId() { /*
    compiled code */ }
33
34      public void setCustomerId(java.lang.Long
    customerId) { /* compiled code */ }
35
36      public java.lang.String getCustomerName() { /*
    compiled code */ }
37
38      public void setCustomerName(java.lang.String
    customerName) { /* compiled code */ }
39
40      public java.lang.String getEmail() { /* compiled
    code */ }
41
42      public void setEmail(java.lang.String email) {
    /* compiled code */ }
43
44      public java.lang.Long getContact() { /* compiled
    code */ }
45
46      public void setContact(java.lang.Long contact) {
    /* compiled code */ }
47
48      public java.lang.String getDate() { /* compiled
    code */ }
49
50      public void setDate(java.lang.String date) { /*
    compiled code */ }
51
52      public java.lang.Float getTotalPrice() { /*
    compiled code */ }
53
54      public void setTotalPrice(java.lang.Float
    totalPrice) { /* compiled code */ }
55
56      public java.util.Set<com.demo.models.
    PurchasedProduct> getList() { /* compiled code */ }
57
58      public void setList(java.util.Set<com.demo.models
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.models;
6
7 @javax.persistence.Entity
8 @javax.persistence.Table(name = "purchased_product")
9 public class PurchasedProduct {
10     @javax.persistence.Id
11
  javax.persistence.GenerationType.IDENTITY)
12     @javax.persistence.Column(name = "product_id")
13     private java.lang.Long productId;
14     @javax.persistence.Column(name = "
  product_category")
15     private java.lang.String category;
16     @javax.persistence.Column(name = "product_company
  ")
17     private java.lang.String company;
18     @javax.persistence.Column(name = "product_size")
19     private int size;
20     @javax.persistence.Column(name = "product_origin"
  )
21     private java.lang.String origin;
22     @javax.persistence.Column(name = "product_price")
23     private java.lang.Float price;
24     @javax.persistence.Column(name = "product_tag")
25     private int tag;
26
27     public PurchasedProduct() { /* compiled code */ }
28
29     public PurchasedProduct(java.lang.String category
  , java.lang.String company, int size, java.lang.
  String origin, java.lang.Float price, int tag) { /*
  compiled code */ }
30
31     public java.lang.Long getProductId() { /*
  compiled code */ }
32
```

```
33      public void setProductId(java.lang.Long productId
    ) { /* compiled code */ }
34
35      public java.lang.String getCategory() { /*
    compiled code */ }
36
37      public void setCategory(java.lang.String category
    ) { /* compiled code */ }
38
39      public java.lang.String getCompany() { /*
    compiled code */ }
40
41      public void setCompany(java.lang.String company
    ) { /* compiled code */ }
42
43      public int getSize() { /* compiled code */ }
44
45      public void setSize(int size) { /* compiled code
     */ }
46
47      public java.lang.String getOrigin() { /* compiled
     code */ }
48
49      public void setOrigin(java.lang.String origin) {
    /* compiled code */ }
50
51      public java.lang.Float getPrice() { /* compiled
    code */ }
52
53      public void setPrice(java.lang.Float price) { /*
    compiled code */ }
54
55      public int getTag() { /* compiled code */ }
56
57      public void setTag(int tag) { /* compiled code */
     }
58 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 @org.springframework.stereotype.Service
8 public class UserServiceImpl implements com.demo.
  services.IUserService {
9
  Autowired
10
11
  Autowired
12
  ;
13
  Autowired
14      com.demo.repository.CustomerRepository
  customerRepo;
15      java.lang.Float total;
16
17      public UserServiceImpl() { /* compiled code */ }
18
19      public com.demo.models.Customer


  Long product_id) throws com.demo.exceptions.
  ResourceNotFound { /* compiled code */ }
20
21      public com.demo.models.User insertUserInDB(com.
  demo.models.User user) { /* compiled code */ }
22
23      public void updateUserInDB(com.demo.models.User
  user, java.lang.Long userId) throws com.demo.
  exceptions.ResourceNotFound { /* compiled code */ }
24
25      public java.util.List<com.demo.models.User>
  getAllUsers() { /* compiled code */ }
26 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 public interface ICustomerService {
8

  product_id) throws com.demo.exceptions.
  ResourceNotFound;
9
10     java.util.List<com.demo.models.Customer>
  getAllCustomers();
11
12     void
  customer, java.lang.Long customerId) throws com.demo.
  exceptions.ResourceNotFound;
13
14     void
  );
15
16

  lang.Long customerId) throws com.demo.exceptions.
  ResourceNotFound;
17
18     com.demo.models.Customer


  lang.Long product_id) throws com.demo.exceptions.
  ResourceNotFound;
19
20     java.util.List<com.demo.models.Customer>
  findByDate(java.lang.String date);
21
22     java.util.List<com.demo.models.Customer>

23 }
```

```
1
2   // IntelliJ API Decompiler stub source generated from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 @org.springframework.stereotype.Service
8 public class ProductServiceImpl implements com.demo.services.IProductService {
9
  Autowired
10     private com.demo.repository.ProductRepository productRepo;
11
12     public ProductServiceImpl() { /* compiled code */ }
13
14     public com.demo.models.Product insertProductInDB(com.demo.models.Product product) { /* compiled code */ }
15
16     public java.util.List<com.demo.models.Product> getAllProducts() { /* compiled code */ }
17
18     public void updateProductInDB(com.demo.models.Product product, java.lang.Long productId) throws com.demo.exceptions.ResourceNotFound { /* compiled code */ }
19
20     public void deleteProductInDB(java.lang.Long productId) { /* compiled code */ }
21
22     public com.demo.models.Product getProductInDB(java.lang.Long productId) throws com.demo.exceptions.ResourceNotFound { /* compiled code */ }
23 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 @org.springframework.stereotype.Service
8 public class CustomerServiceImpl implements com.demo.
  services.ICustomerService {
9
  Autowired
10      private com.demo.repository.CustomerRepository
  customerRepo;
11
  Autowired
12      private com.demo.repository.ProductRepository
  productRepo;
13      float total;
14
15      public CustomerServiceImpl() { /* compiled code
     */ }
16
17      public com.demo.models.Customer

  , java.lang.Long product_id) throws com.demo.
  exceptions.ResourceNotFound { /* compiled code */ }
18
19      public java.util.List<com.demo.models.Customer>
  getAllCustomers() { /* compiled code */ }
20
21      public void updateCustomerInDB(com.demo.models.
  Customer customer, java.lang.Long customerId) throws
  com.demo.exceptions.ResourceNotFound { /* compiled
  code */ }
22
23      public void deleteCustomerInDB(java.lang.Long
  customerId) { /* compiled code */ }
24
25      public com.demo.models.Customer getCustomerInDB(
  java.lang.Long customerId) throws com.demo.exceptions
```

```
25 .ResourceNotFound { /* compiled code */ }
26
27     public com.demo.models.Customer


   lang.Long product_id) throws com.demo.exceptions.
   ResourceNotFound { /* compiled code */ }
28
29     public java.util.List<com.demo.models.Customer>
   findByDate(java.lang.String date) { /* compiled code
    */ }
30
31     public java.util.List<com.demo.models.Customer>

   /* compiled code */ }
32 }
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 @org.springframework.stereotype.Service
8 public class PurchasedProductImpl implements com.demo
  .services.IPurchasedProductService {
9
  Autowired
10      private com.demo.repository.

11
12      public PurchasedProductImpl() { /* compiled code
    */ }
13
14      public com.demo.models.PurchasedProduct

  PurchasedProduct product) { /* compiled code */ }
15
16      public java.util.List<com.demo.models.
  PurchasedProduct> getAllPurchasedProducts() { /*
  compiled code */ }
17
18      public void updatePurchasedProductInDB(com.demo.

  productId) throws com.demo.exceptions.
  ResourceNotFound { /* compiled code */ }
19
20      public void deletePurchasedProductInDB(java.lang.
  Long productId) { /* compiled code */ }
21
22      public com.demo.models.PurchasedProduct

  throws com.demo.exceptions.ResourceNotFound { /*
  compiled code */ }
23
24      public java.util.List<com.demo.models.
```

```
1
2   // IntelliJ API Decompiler stub source generated
  from a class file
3   // Implementation of methods is not available
4
5 package com.demo.services;
6
7 public interface IPurchasedProductService {
8     com.demo.models.PurchasedProduct

  PurchasedProduct product);
9
10

  getAllPurchasedProducts();
11
12     void

  throws com.demo.exceptions.ResourceNotFound;
13
14     void deletePurchasedProductInDB(java.lang.Long
  productId);
15
16     com.demo.models.PurchasedProduct

  throws com.demo.exceptions.ResourceNotFound;
17
18

  findByCategory(java.lang.String category);
19
20


  .query.Param("id") java.lang.Long id);
21 }
```

```
24      @org.springframework.web.bind.annotation.
    DeleteMapping({"/deletepurchasedproduct/{productid}"
    })
25      public void deletePurchasedProduct(@org.
                                                      "
    productid") java.lang.Long productId) { /* compiled
    code */ }
26
27      @org.springframework.web.bind.annotation.
    PutMapping({"/updatepurchasedproductbyid/{productid}"
    })
28      public void updatePurchasedProduct(@org.
                                                      "
    productid") java.lang.Long productId, @org.

    demo.models.PurchasedProduct product) throws com.demo
    .exceptions.ResourceNotFound { /* compiled code */ }
29  }
```