```
<dependencies>
    <!-- Hibernate core dependency -->
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>5.6.9.Final</version>
    </dependency>

    <!-- Hibernate Validator (optional) -->
    <dependency>
        <groupId>org.hibernate.validator</groupId>
        <artifactId>hibernate-validator</artifactId>
        <version>6.2.0.Final</version>
    </dependency>

    <!-- MySQL JDBC Driver -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.30</version>
    </dependency>

    <!-- Logging (Optional for Hibernate) -->
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-api</artifactId>
        <version>1.7.32</version>
    </dependency>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-simple</artifactId>
        <version>1.7.32</version>
    </dependency>

    <!-- Javax Persistence API -->
    <dependency>
        <groupId>javax.persistence</groupId>
        <artifactId>javax.persistence-api</artifactId>
        <version>2.2</version>
    </dependency>
</dependencies>
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD
3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <!-- JDBC Database connection settings -->
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/labexam</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">password</property>

        <!-- JDBC connection pool settings -->
        <property name="hibernate.c3p0.min_size">5</property>
        <property name="hibernate.c3p0.max_size">20</property>
```

```
        <!-- Echo all executed SQL to stdout -->
        <property name="hibernate.show_sql">true</property>

        <!-- Drop and re-create the database schema on startup -->
        <property name="hibernate.hbm2ddl.auto">update</property>

        <!-- Mention annotated class -->
        <mapping class="com.klef.jfsd.exam.Device"/>
        <mapping class="com.klef.jfsd.exam.Smartphone"/>
        <mapping class="com.klef.jfsd.exam.Tablet"/>
    </session-factory>
</hibernate-configuration>
package com.klef.jfsd.exam;

import javax.persistence.*;

@Entity
@Table(name = "device")
@Inheritance(strategy = InheritanceType.JOINED) // Table per Subclass strategy
public class Device {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "brand")
    private String brand;

    @Column(name = "model")
    private String model;

    @Column(name = "price")
    private double price;

    // Getters and Setters
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getBrand() {
        return brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }
```

```
    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }
}
package com.klef.jfsd.exam;

import javax.persistence.*;

@Entity
@Table(name = "smartphone")
public class Smartphone extends Device {

    @Column(name = "operating_system")
    private String operatingSystem;

    @Column(name = "camera_resolution")
    private String cameraResolution;

    // Getters and Setters
    public String getOperatingSystem() {
        return operatingSystem;
    }

    public void setOperatingSystem(String operatingSystem) {
        this.operatingSystem = operatingSystem;
    }

    public String getCameraResolution() {
        return cameraResolution;
    }

    public void setCameraResolution(String cameraResolution) {
        this.cameraResolution = cameraResolution;
    }
}
package com.klef.jfsd.exam;

import javax.persistence.*;

@Entity
@Table(name = "tablet")
public class Tablet extends Device {

    @Column(name = "screen_size")
    private double screenSize;

    @Column(name = "battery_life")
    private double batteryLife;

    // Getters and Setters
    public double getScreenSize() {
        return screenSize;
    }
```

```
    public void setScreenSize(double screenSize) {
        this.screenSize = screenSize;
    }

    public double getBatteryLife() {
        return batteryLife;
    }

    public void setBatteryLife(double batteryLife) {
        this.batteryLife = batteryLife;
    }
}
package com.klef.jfsd.exam;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class ClientDemo {

    public static void main(String[] args) {

        // Setup Hibernate session factory
        SessionFactory factory = new Configuration()
                .configure("hibernate.cfg.xml")
                .addAnnotatedClass(Device.class)
                .addAnnotatedClass(Smartphone.class)
                .addAnnotatedClass(Tablet.class)
                .buildSessionFactory();

        Session session = factory.getCurrentSession();

        try {
            // Create Device object
            Device device = new Device();
            device.setBrand("Generic");
            device.setModel("Device 1");
            device.setPrice(100);

            // Create Smartphone object
            Smartphone smartphone = new Smartphone();
            smartphone.setBrand("Samsung");
            smartphone.setModel("Galaxy S22");
            smartphone.setPrice(799.99);
            smartphone.setOperatingSystem("Android");
            smartphone.setCameraResolution("108 MP");

            // Create Tablet object
            Tablet tablet = new Tablet();
            tablet.setBrand("Apple");
            tablet.setModel("iPad Pro");
            tablet.setPrice(1099.99);
            tablet.setScreenSize(12.9);
            tablet.setBatteryLife(10);

            // Start a transaction
            session.beginTransaction();

            // Save the devices (smartphone and tablet will be saved through
```

```
inheritance)
            session.save(device);
            session.save(smartphone);
            session.save(tablet);

            // Commit the transaction
            session.getTransaction().commit();

            System.out.println("Records saved successfully!");

        } finally {
            factory.close();
        }
    }
}
```