Spring Data Retake Exam – 13 August 2021

Laptop Shop

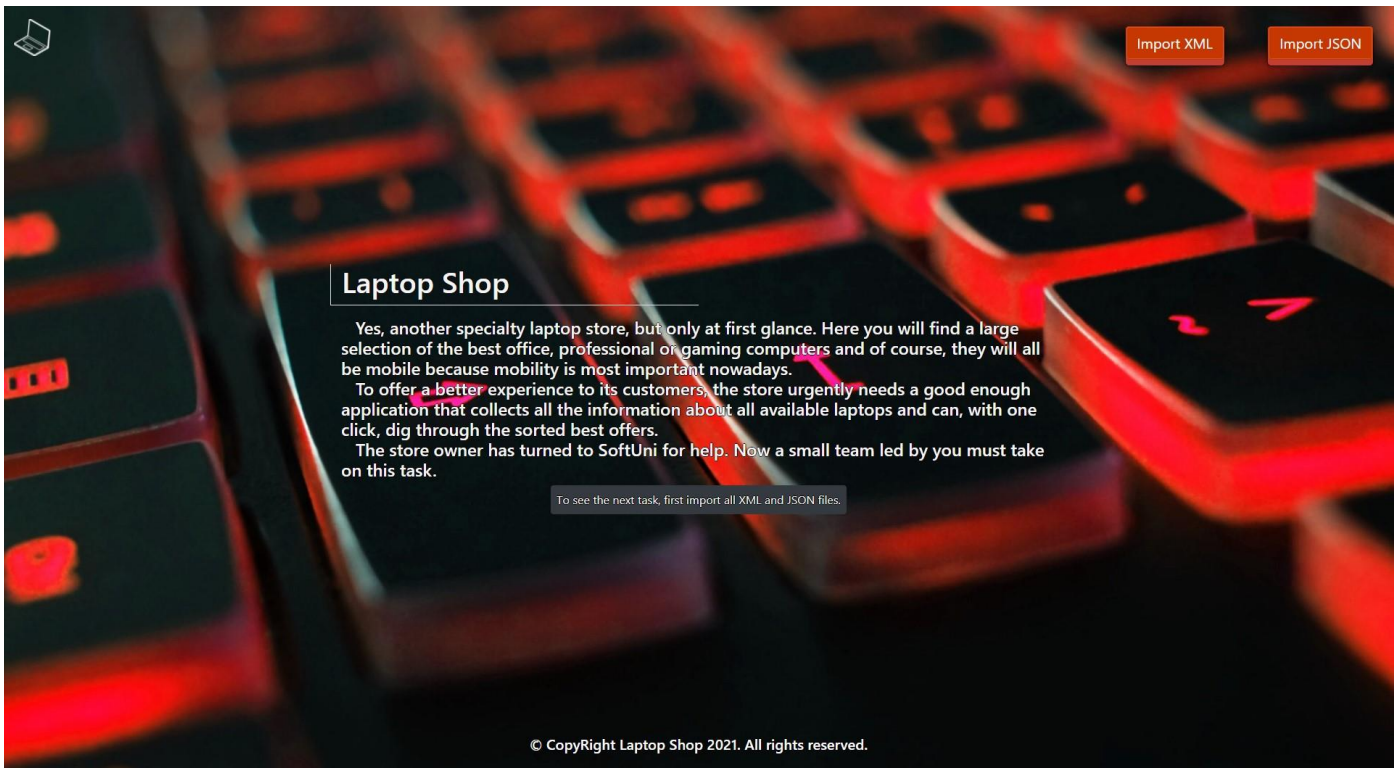
Yes, another specialty laptop store, but only at first glance. Here you will find a large selection of the best office, professional or gaming computers and of course, they will all be mobile because mobility is most important nowadays. To offer a better experience to its customers, the store urgently needs a good enough application that collects all the information about all available laptops and can, with one click, dig through the sorted best offers. The store owner has turned to SoftUni for help. Now a small team led by you must take on this task.

1. Functionality Overview

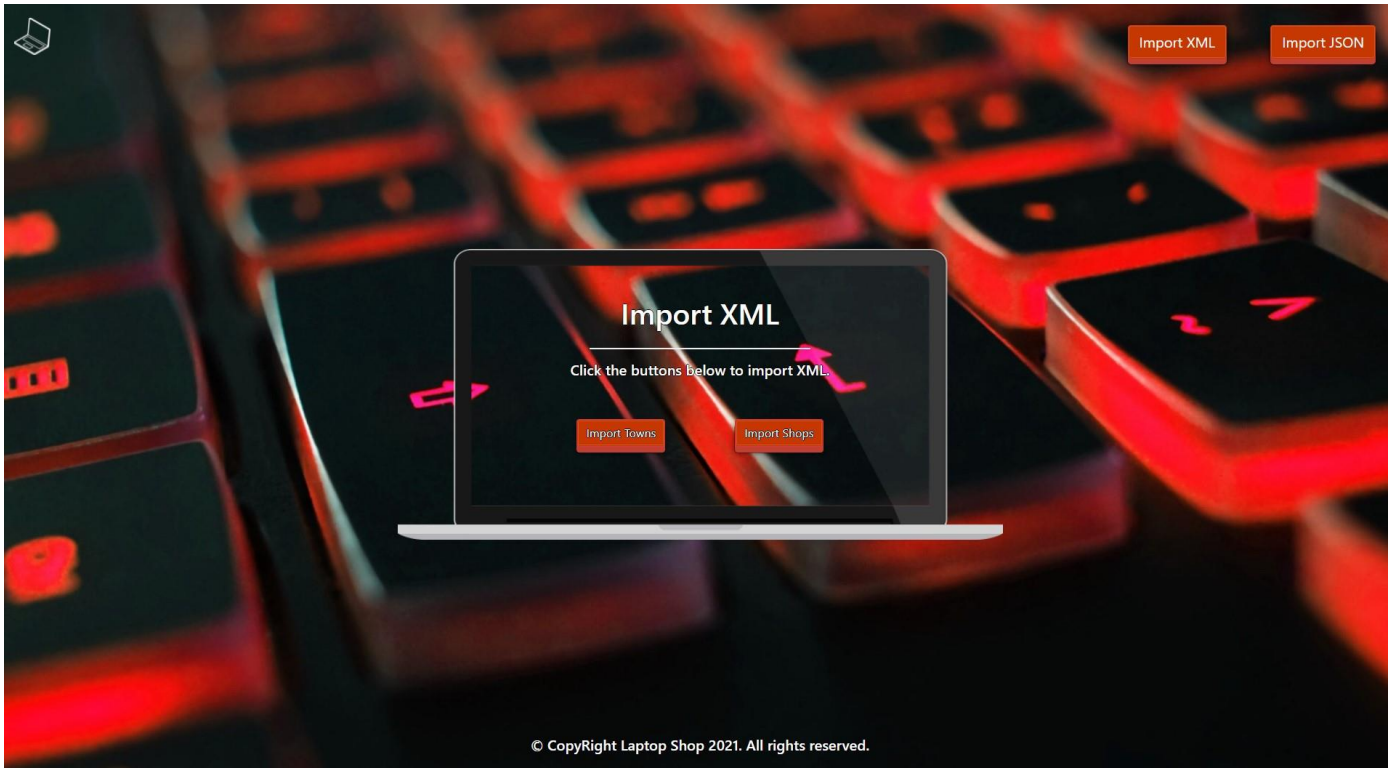
The application should be able to easily **import** hard-formatted data and **support functionalities** for also **exporting** the imported data. The application is called – **Laptop Shop**.

Look at the pictures below to see what must happen:

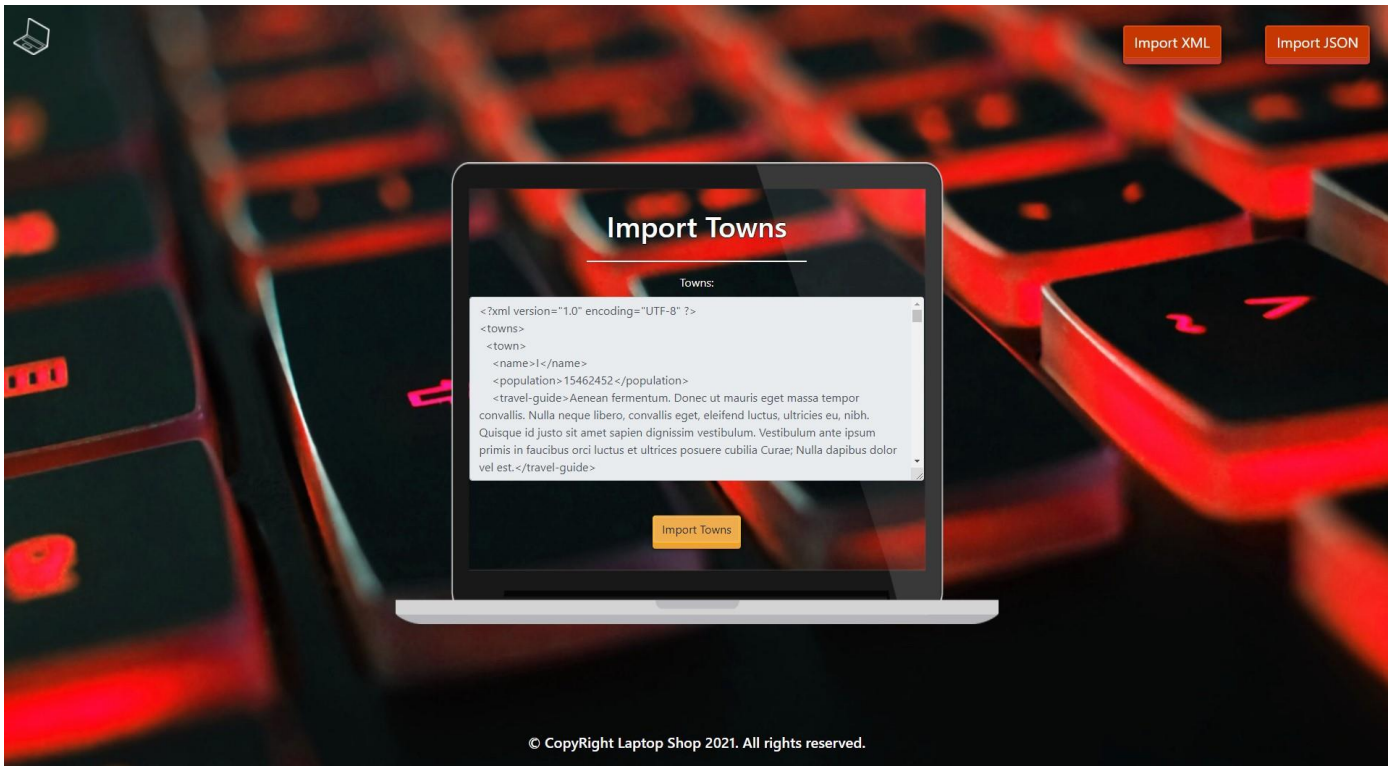
- The home page before importing anything:



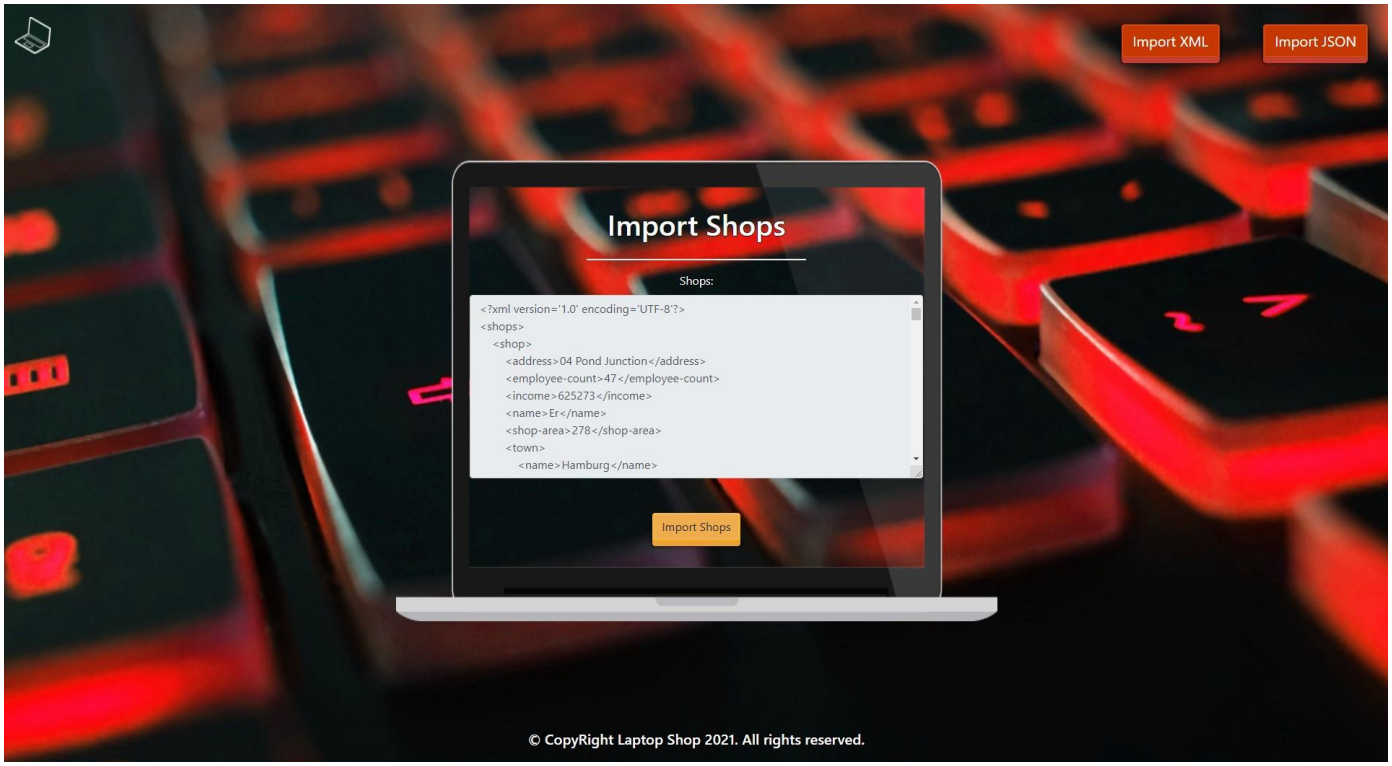
- The import XML page before importing anything:



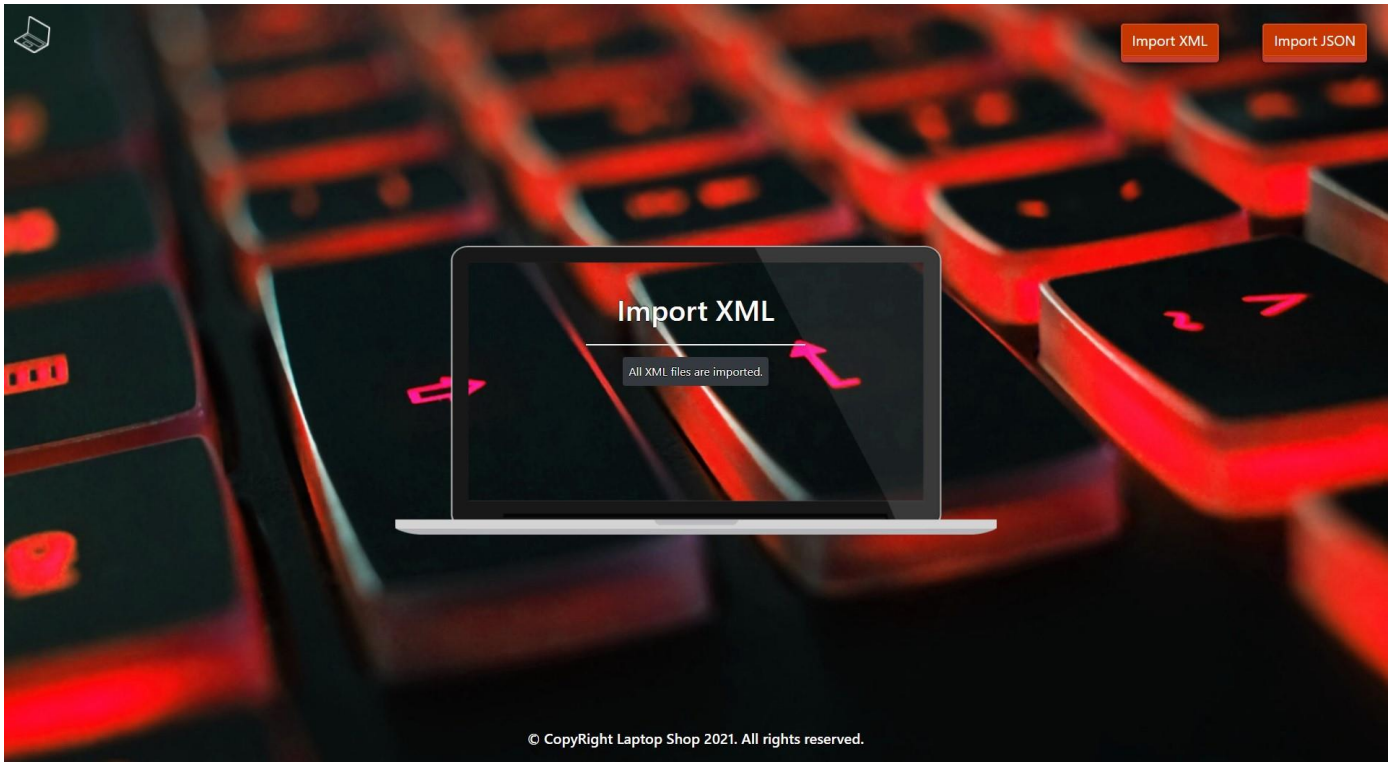
- Import the towns first:



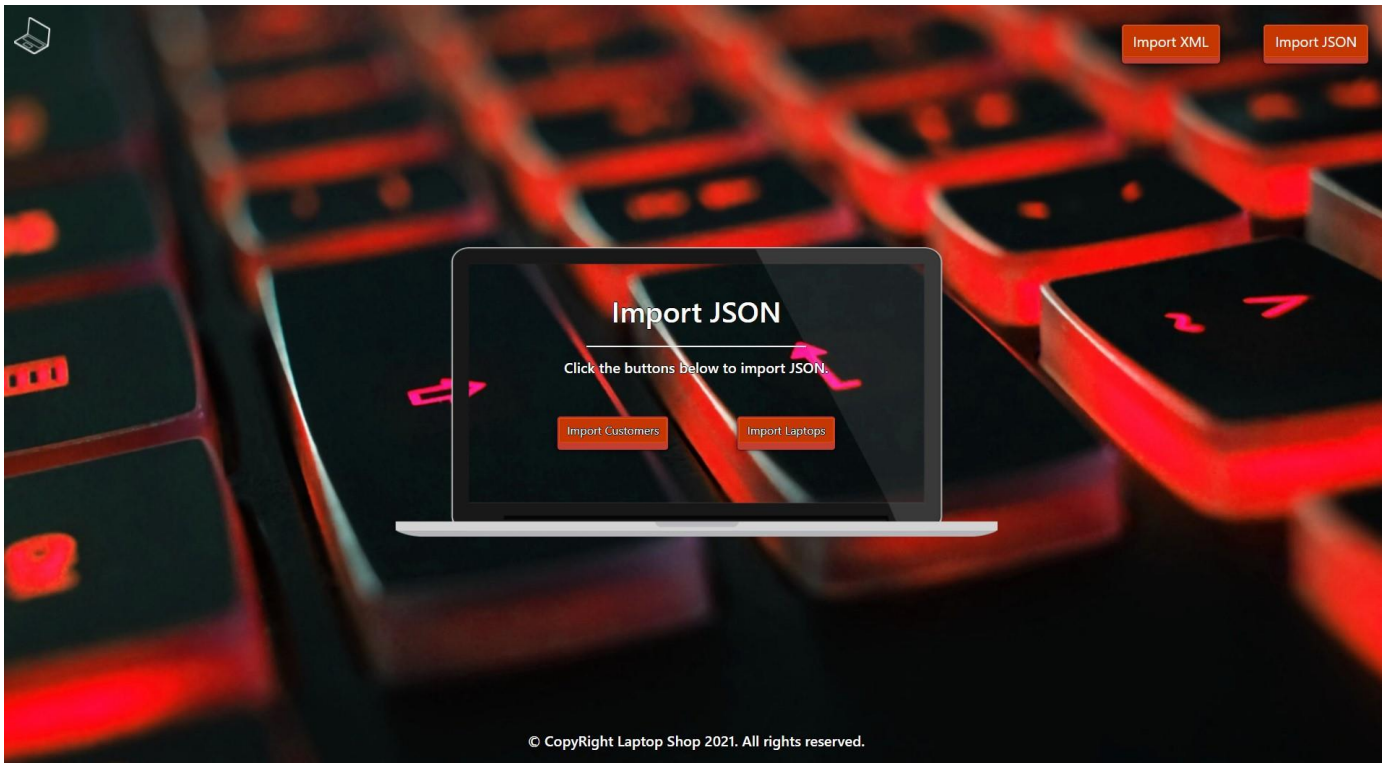
- Import the shops second:



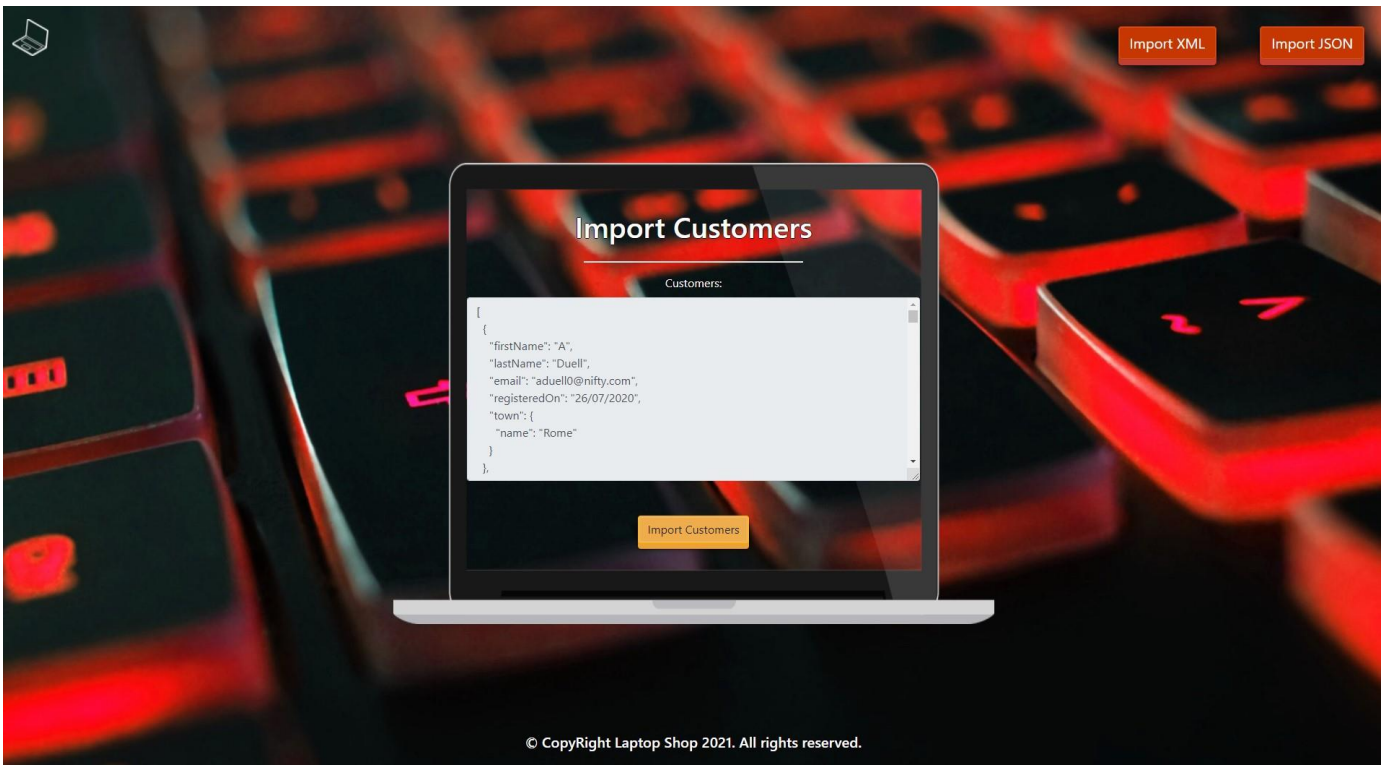
- The import XML page after importing both files:



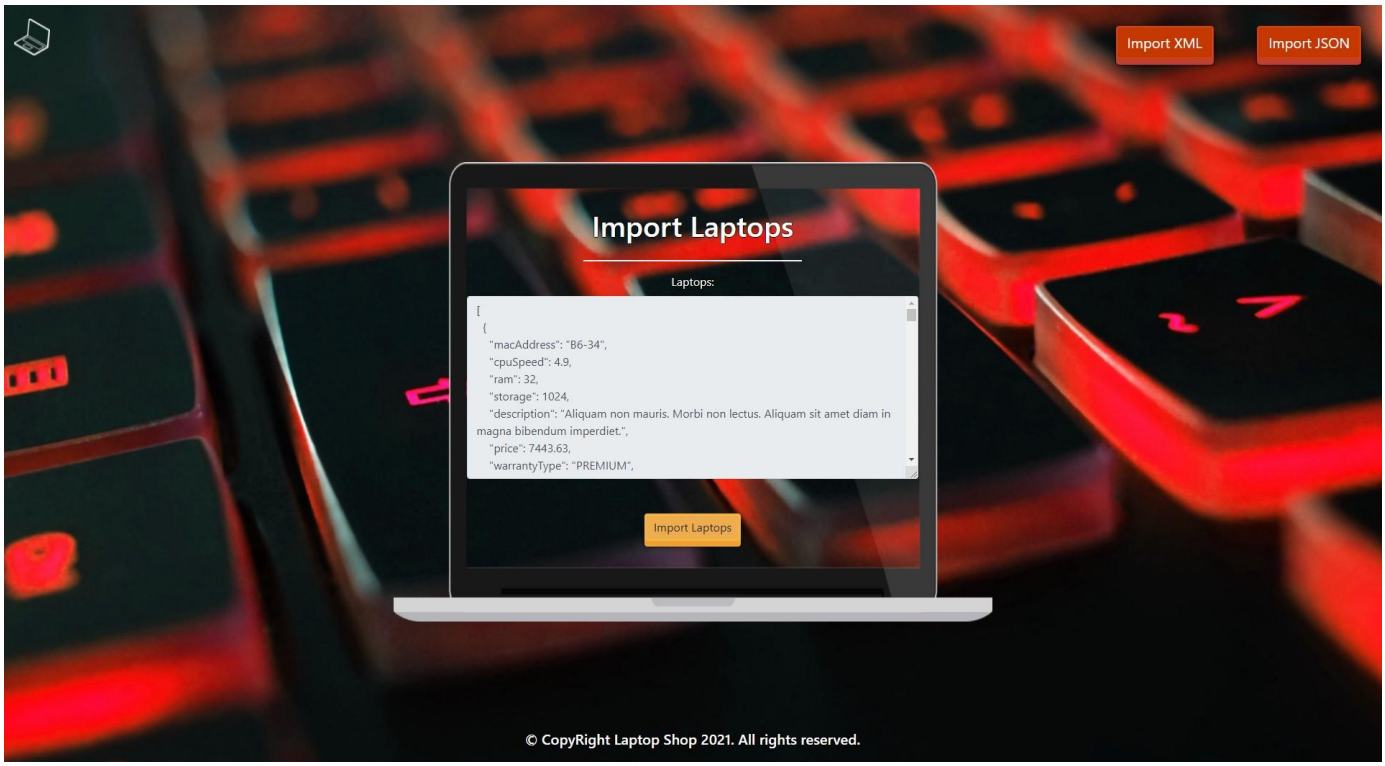
- The import JSON page before importing the given data:



- Import the customers' data:

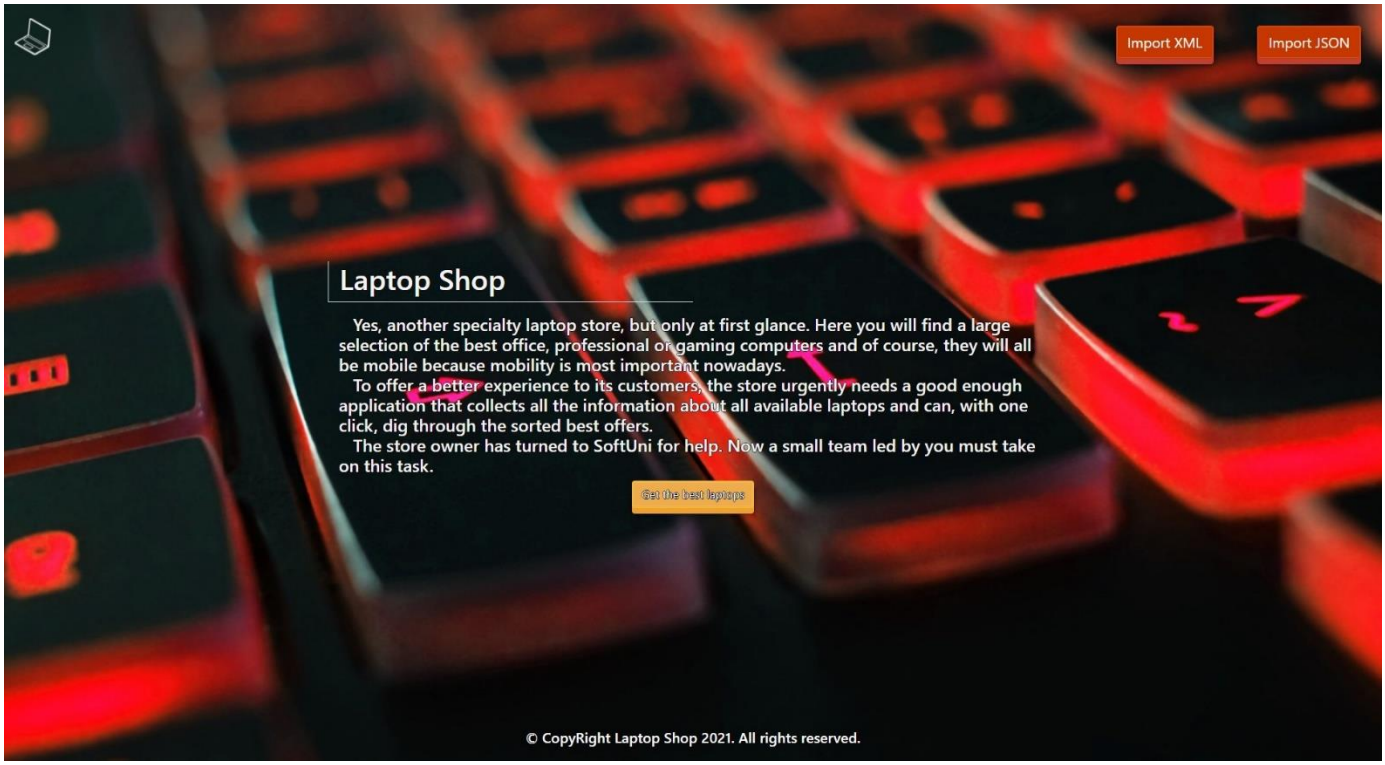


- Import the laptops' data:

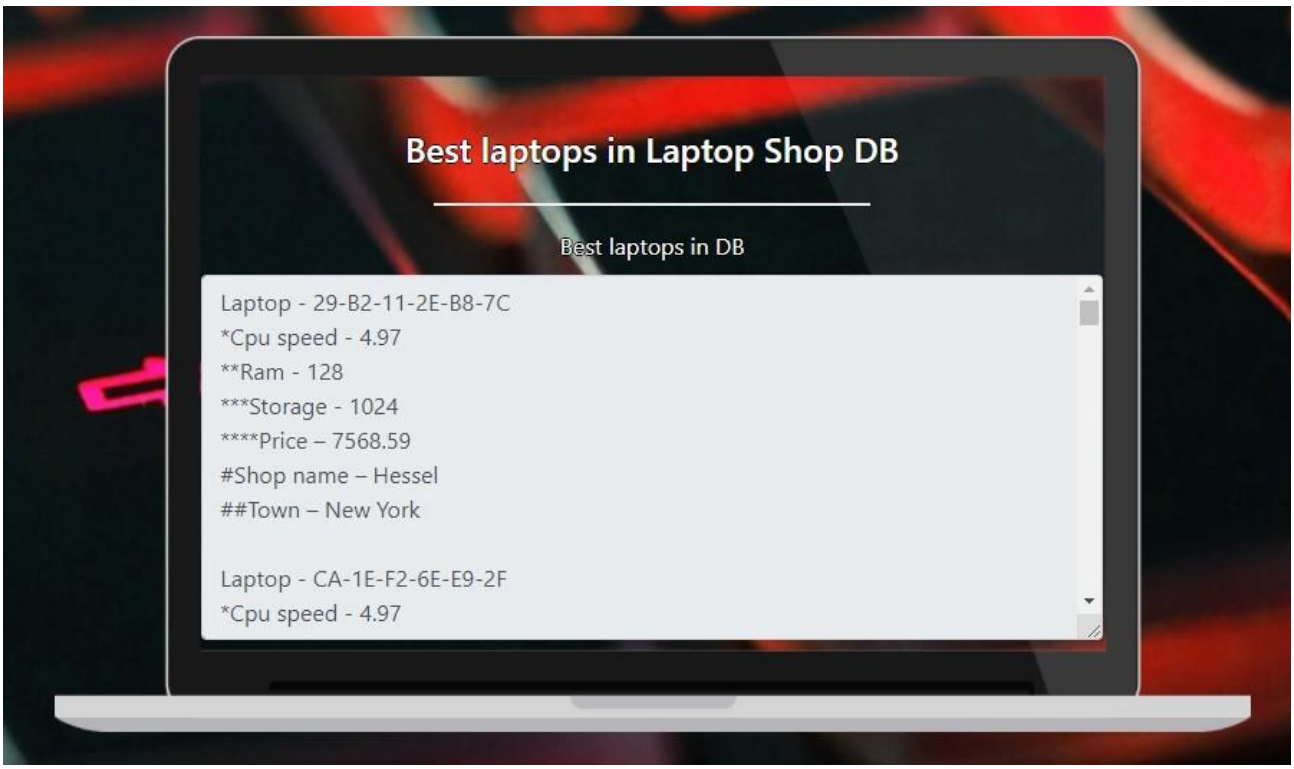


- The import JSON page after importing the data:

- The home page after the data is imported:



- Export laptops by their characteristics:



2. Project Skeleton Overview

You will be given a **skeleton**, containing a **certain architecture (MVC)** with **several classes**, some of which are completely empty. The **Skeleton** will include the **files** with which you will **seed** the **database**.

3. Model Definition

There are 4 main models that the **Laptop Shop database** application should contain in its functionality.

Design them in the **most appropriate** way, considering the following **data constraints**:

Town

- **id** – accepts **integer** values, a **primary identification field, an auto incremented field**.
- **name** – accepts **char sequences** as values where their character length value **higher than or equal to 2.** The values are **unique in the database**.
- **population** – accepts **number** values (must be positive), 0 as a value is **exclusive**.
- **travel guide – a long and detailed description** of all known places with a character length value **higher than or equal to 10.**

Shop

- **id** – accepts **integer** values, a **primary identification field, an auto incremented field**.
- **name** – accepts **char sequences** as values where their character length value **higher than or equal to 4.** The values are **unique in the database**.
- **income** – accepts **number** values that are **more** than or **equal** to **20000.**
- **address** – accepts **char sequences** as values where their character length value **higher than or equal to 4.**
- **employee count** – accepts **number** values that are between 1 and 50
    - **(Larger than** or **equal** to **1** and **less than** or **equal** to **50).**
- **shop area** – accepts **number** values that are **more** than or **equal** to **150.**
- **Constraint:** The shops table has a relation with the towns table.

Customer

- **id** – accepts **integer** values, a **primary identification field, an auto incremented field**.
- **first name –** accepts **char sequences** as values where their character length value **higher than or equal to 2.**
- **last name –** accepts **char sequences** as values where their character length value **higher than or equal to 2.**
- **email –** accepts valid **email addresses** (must contains '@' and '.' – a dot). The values are **unique in the database.**
- **registered on – a date when a customer registers in the shop.**
- **Constraint:** The customers table has a relation with the towns table.

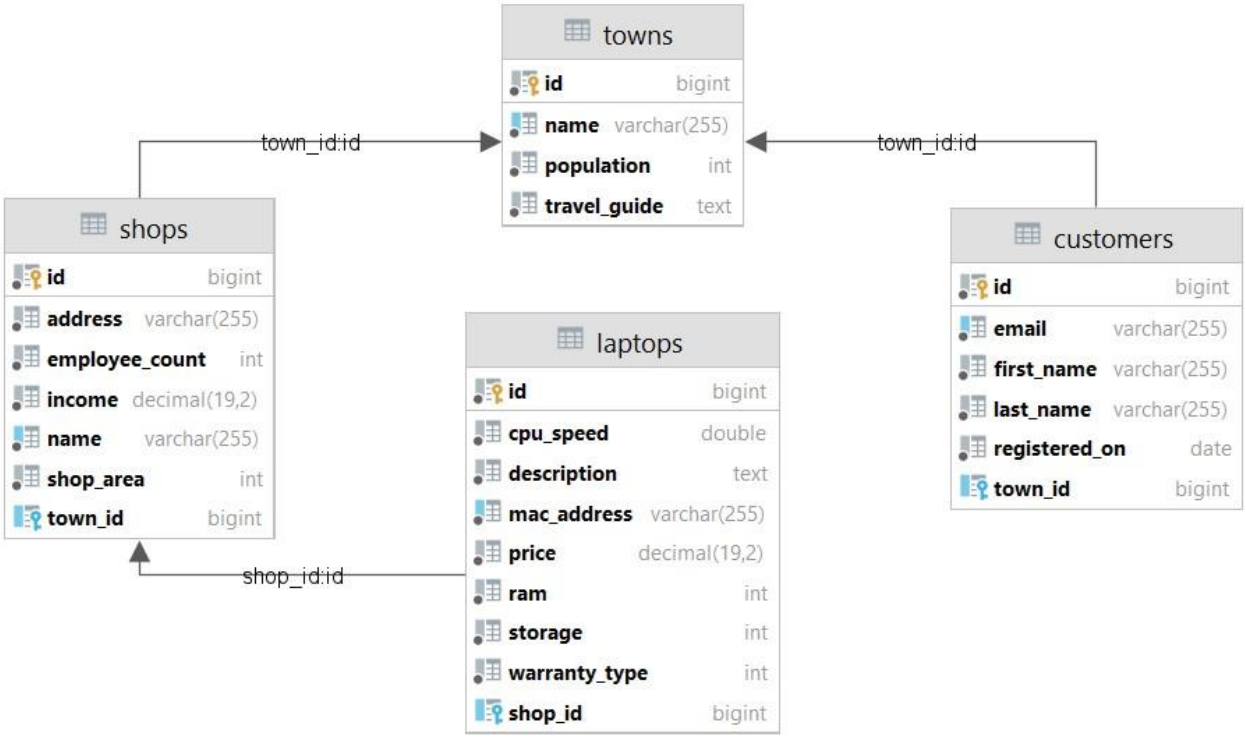
Laptop

- **id** – accepts **integer** values, a **primary identification field, an auto incremented field**.
- **mac address** – accepts **char sequences** as values where their character length value **higher than 8.** The values are **unique in the database.**
- **cpu speed** – accepts positive floating-point numbers.
- **ram** – accepts **number** values that are **more** than or **equal** to **8 and less than or equal to 128**
- **storage** – accepts **number** values that are **more** than or **equal** to **128 and less than or equal to 1024**

- **description** – **a long and detailed description** of all known places with a character length value **higher than or equal to 10**.
- **price** – accepts a **positive number**.
- **warranty type** – the enumeration, one of the following – **BASIC, PREMIUM, LIFETIME**.
- **Constraint:** The laptops table has a relation with the shops table.

Constraint:

- Name the entities and their class members **exactly** in the **format stated** above.
- All fields are **NOT NULL** unless explicitly stated to be nullable.



4. Data Import

Use the provided files to populate the database with data. Import all the information from those files into the database.

You are not allowed to modify the provided files.

ANY INCORRECT data should be **ignored** and a message:

"**Invalid {Town / Shop / Customer/ Laptop}**" should be printed.

When the import is finished:

"**Successfully imported {Town/ Shop / Customer/ Laptop} {town name / shop name - income / first name – last name - email / laptop mac address – cpu – ram -hdd}**"

XML Import

Your new colleagues have prepared some XML data for you to import.

Towns (towns.xml)

```
<?xml version="1.0" encoding="UTF-8" ?>
<towns>
  <town>
    <name>I</name>
    <population>15462452</population>
    <travel-guide>Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque libero, convallis eget, eleifend luctus, ultricies eu, nibh. Quisque id justo sit amet sapien dignissim vestibulum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nulla dapibus dolor vel est.</travel-guide>
  </town>
  <town>
    <name>Moscow</name>
    <population>12195221</population>
    <travel-guide>Suspendisse potenti. Cras in purus eu magna vulputate luctus. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Vivamus vestibulum sagittis sapien. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam vel augue. Vestibulum rutrum rutrum neque. Aenean auctor gravida sem. Praesent id massa id nisl venenatis lacinia. Aenean sit amet justo.</travel-guide>
  </town>
  <town>
    <name>London</name>
    <population>9126366</population>
    <travel-guide>Short.</travel-guide>
  </town>
  <town>
    <name>Saint Petersburg</name>
    <population>-5383890</population>
    <travel-guide>Nullam varius. Nulla facilisi. Cras non velit nec nisi vulputate nonummy. Maecenas tincidunt lacus at velit. Vivamus vel nulla eget eros elementum pellentesque. Quisque porta volutpat erat. Quisque erat eros, viverra eget, congue eget, semper rutrum, nulla.</travel-guide>
  </town>
  <town>
    <name>Berlin</name>
    <population>3748148</population>
    <travel-guide>Maecenas tincidunt lacus at velit. Vivamus vel nulla eget eros elementum pellentesque.</travel-guide>
  </town>
...
```

```
Invalid town
Successfully imported Town Moscow
Invalid town
Invalid town
Successfully imported Town Berlin
…
```

Constraint:

- **If the shop's name already exists in the DB return "Invalid Shop".**
- **The provided town names will always be valid.**

Shops (shops.xml)

```
<?xml version='1.0' encoding='UTF-8'?>
<shops>
    <shop>
        <address>04 Pond Junction</address>
        <employee-count>47</employee-count>
        <income>625273</income>
        <name>Er</name>
        <shop-area>278</shop-area>
        <town>
            <name>Hamburg</name>
        </town>
    </shop>
    <shop>
        <address>779 Parkside Park</address>
        <employee-count>48</employee-count>
        <income>10</income>
        <name>Raynor</name>
        <shop-area>108</shop-area>
        <town>
            <name>Birmingham</name>
        </town>
    </shop>
    <shop>
        <address>15c</address>
        <employee-count>40</employee-count>
        <income>2710190</income>
        <name>Lockman-Stroman</name>
        <shop-area>419</shop-area>
        <town>
            <name>New York</name>
        </town>
    </shop>
    <shop>
        <address>6 Kennedy Lane</address>
        <employee-count>47</employee-count>
        <income>2078882</income>
        <name>Nader-Wehner</name>
        <shop-area>50</shop-area>
        <town>
            <name>Barcelona</name>
        </town>
    </shop>
    <shop>
        <address>95 Delaware Avenue</address>
        <employee-count>0</employee-count>
        <income>387613</income>
        <name>Langosh-Mraz</name>
```

```
        <shop-area>445</shop-area>
        <town>
            <name>New York</name>
        </town>
    </shop>
    <shop>
        <address>340 Londonderry Junction</address>
        <employee-count>31</employee-count>
        <income>1511525</income>
        <name>Treutel</name>
        <shop-area>164</shop-area>
        <town>
            <name>Moscow</name>
        </town>
    </shop>
...
```

```
Invalid shop
Invalid shop
Invalid shop
Invalid shop
Invalid shop
Successfully imported Shop Treutel - 1511525
...
```

JSON Import

Your new colleagues have prepared some JSON data for you to import.

Constraint:

- **If the Customer email already exists in the DB return "Invalid Customer".**
- **The provided town names will always be valid.**

Customers (Customers.json)

Customers (customers.json)

```
[
  {
    "firstName": "A",
    "lastName": "Duell",
    "email": "aduell0@nifty.com",
    "registeredOn": "26/07/2020",
    "town": {
      "name": "Rome"
    }
  },
  {
    "firstName": "Lorne",
    "lastName": "C",
    "email": "lcurtayne1@nbcnews.com",
    "registeredOn": "27/06/2020",
    "town": {
```

```
      "name": "Kyiv"
    }
  },
  {
    "firstName": "Mil",
    "lastName": "Armitty",
    "email": "marmit2@digg.com",
    "registeredOn": "03/07/2020",
    "town": {
      "name": "Chicago"
    }
  },
  {
    "firstName": "Odella",
    "lastName": "Scully",
    "email": "notValidMail.com",
    "registeredOn": "21/11/2020",
    "town": {
      "name": "Paris"
    }
  },
  {
    "firstName": "L",
    "lastName": "Healks",
    "email": "lhealks4@yahoo.com",
    "registeredOn": "13/12/2020",
    "town": {
      "name": "Chicago"
    }
  },
  {
    "firstName": "Bil",
    "lastName": "Sadat",
    "email": "bsadat5@cyberchimps.com",
    "registeredOn": "21/01/2020",
    "town": {
      "name": "Belgrade"
    }
  },

...
```


```
Invalid Customer
Invalid Customer
Successfully imported Customer Mil Armitty - marmit2@digg.com
Invalid Customer
Invalid Customer
Successfully imported Customer Bil Sadat - bsadat5@cyberchimps.com
...
```

Laptops (laptops.json)

Constraint:

- **If the laptop's mac address already exists in the DB return "Invalid Laptop".**
- **The provided shop names will always be valid.**
- **Format the cpu speed to the second digit after the decimal point.**

Laptops (laptops.json)


```
[
  {
    "macAddress": "B6-34",
    "cpuSpeed": 4.9,
    "ram": 32,
    "storage": 1024,
    "description": "Aliquam non mauris. Morbi non lectus. Aliquam sit amet diam in
magna bibendum imperdiet.",
    "price": 7443.63,
    "warrantyType": "PREMIUM",
    "shop": {
      "name": "Becker"
    }
  },
  {
    "macAddress": "45-F8-0F-D3-A9-FC",
    "cpuSpeed": -5.65,
    "ram": 8,
    "storage": 1024,
    "description": "Phasellus in felis. Donec semper sapien a libero. Nam dui. Proin
leo odio, porttitor id, consequat in, consequat ut, nulla. Sed accumsan felis.",
    "price": 6591.95,
    "warrantyType": "LIFETIME",
    "shop": {
      "name": "Ferry"
    }
  },
  {
    "macAddress": "8C-B5-5E-F9-E1-0E",
    "cpuSpeed": 2.09,
    "ram": 2,
    "storage": 1024,
    "description": "Pellentesque viverra pede ac diam. Cras pellentesque volutpat
dui. Maecenas tristique, est et tempus semper, est quam pharetra magna, ac consequat
metus sapien ut nunc. Vestibulum ante ipsum primis in faucibus orci luctus et
ultrices posuere cubilia Curae; Mauris viverra diam vitae quam.",
    "price": 1103.69,
    "warrantyType": "BASIC",
    "shop": {
      "name": "Towne-Jast"
    }
  },
  {
    "macAddress": "DC-85-FA-B8-55-93",
    "cpuSpeed": 4.21,
    "ram": 8,
    "storage": 64,
    "description": "Vivamus metus arcu, adipiscing molestie, hendrerit at, vulputate
vitae, nisl.",
    "price": 6218.74,
    "warrantyType": "PREMIUM",
    "shop": {
      "name": "Watsica"
    }
  },
  {
    "macAddress": "F0-4F-A5-06-9F-1E",
```

```
    "cpuSpeed": 3.65,
    "ram": 16,
    "storage": 128,
    "description": "Short.",
    "price": 6308.28,
    "warrantyType": "PREMIUM",
    "shop": {
      "name": "O'Reilly"
    }
  },
  {
    "macAddress": "95-FF-4C-B2-E2-25",
    "cpuSpeed": 4.84,
    "ram": 64,
    "storage": 1024,
    "description": "Nulla justo. Aliquam quis turpis eget elit sodales scelerisque.
Mauris sit amet eros. Suspendisse accumsan tortor quis turpis. Sed ante.",
    "price": -50.00,
    "warrantyType": "PREMIUM",
    "shop": {
      "name": "Walker Inc"
    }
  },
  {
    "macAddress": "C4-90-3F-52-5D-83",
    "cpuSpeed": 3.99,
    "ram": 64,
    "storage": 512,
    "description": "Nulla ac enim. In tempor, turpis nec euismod scelerisque, quam
turpis adipiscing lorem, vitae mattis nibh ligula nec sem. Duis aliquam convallis
nunc. Proin at turpis a pede posuere nonummy. Integer non velit.",
    "price": 4401.56,
    "warrantyType": "INVALID WARRANTY TYPE",
    "shop": {
      "name": "Kerluke"
    }
  },
  {
    "macAddress": "B5-42-0A-AC-F0-19",
    "cpuSpeed": 1.46,
    "ram": 128,
    "storage": 128,
    "description": "Aenean lectus. Pellentesque eget nunc.",
    "price": 4081.54,
    "warrantyType": "PREMIUM",
    "shop": {
      "name": "Jacobi and Bayer"
    }
  },
...
```


```
Invalid Laptop
Invalid Laptop
Invalid Laptop
Invalid Laptop
Invalid Laptop
Invalid Laptop
Invalid Laptop
Successfully imported Laptop B5-42-0A-AC-F0-19 - 1.46 - 128 - 128
...
```

5. Data Export

Get ready to export the data you have imported in the previous task. Here you will have some complex database querying. Export the data in the formats specified below.

Export Best Laptops from the Data Base

- Extract from the database, the **mac address**, **CPU speed (to second digit after decimal point)**, **ram**, **storage,** and the **price (to second digit after decimal point)** of the **laptop**. Also, we need to show the **name** of the **shop** and the **name** of the **town**.
- **Order Them by the cpu speed in descending order, Then by the ram in descending order, then by the storage in descending order and finally by the MAC Address**
- Return the information in this format:
- ```
  "Laptop - {mac address}
  *Cpu speed - {cpu speed}
  **Ram - {ram}
  ***Storage - {storage}
  ****Price - {price}
  #Shop name - {name of the shop}
  ##Town - {the name of the town of shop}
  . . . "
  ```
  ```