Databases Advanced Exam
Most Wanted(MVC)

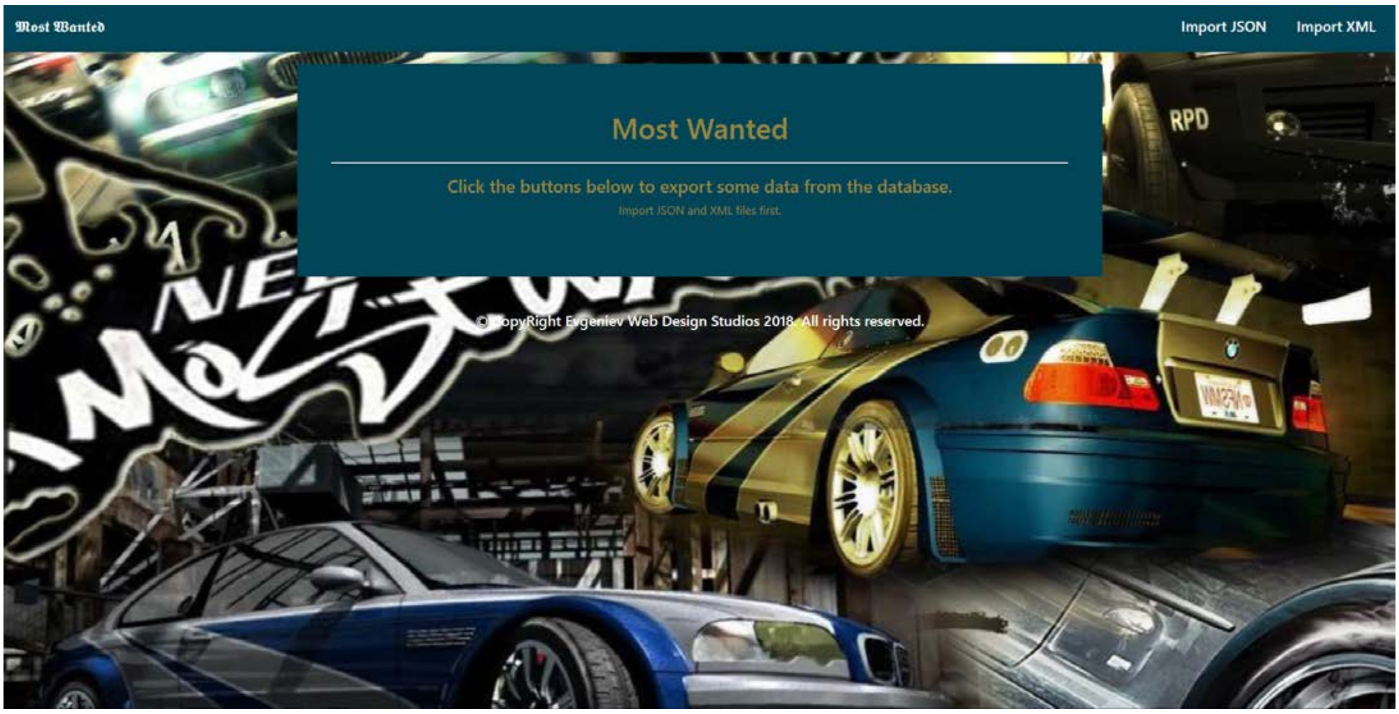
Racing has become an essential part of some people's life. It's the energy that flows through their veins. However, some races get out of control creating massive traffic disasters, which forced the Laws to create The ARPU – Anti-Race Police Unit… A specialized unit, which does not spare even a penny, when it comes to stopping races. Their performance tuned cars are on par with most racers' cars and they aren't afraid of crashing them.

1. Functionality Overview

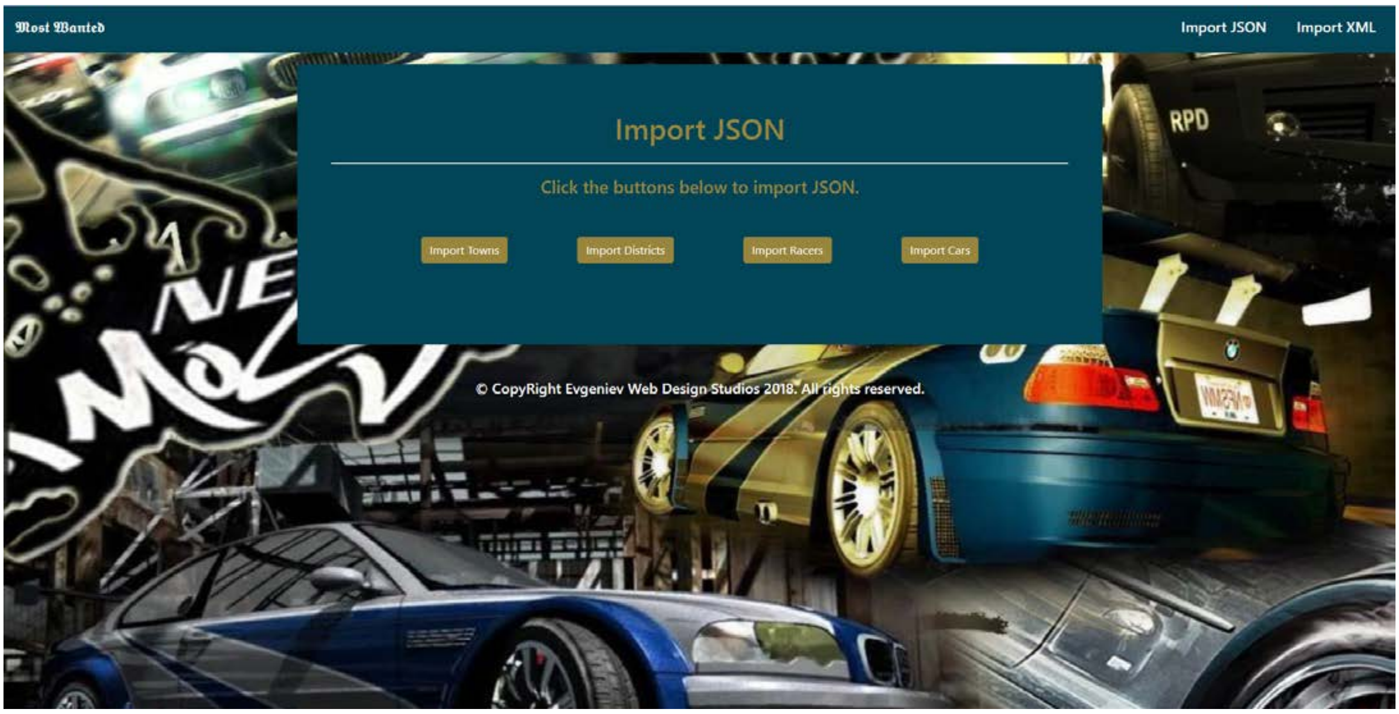
The ARPU has hired you as their database developer, to implement a **database application**. The application should be able to easily **import** hard-formatted data from **XML** and **JSON** and **support functionality** for also **exporting** into the imported data. The application is called – **Most Wanted**.

Look at the pictures below to see what must happen:

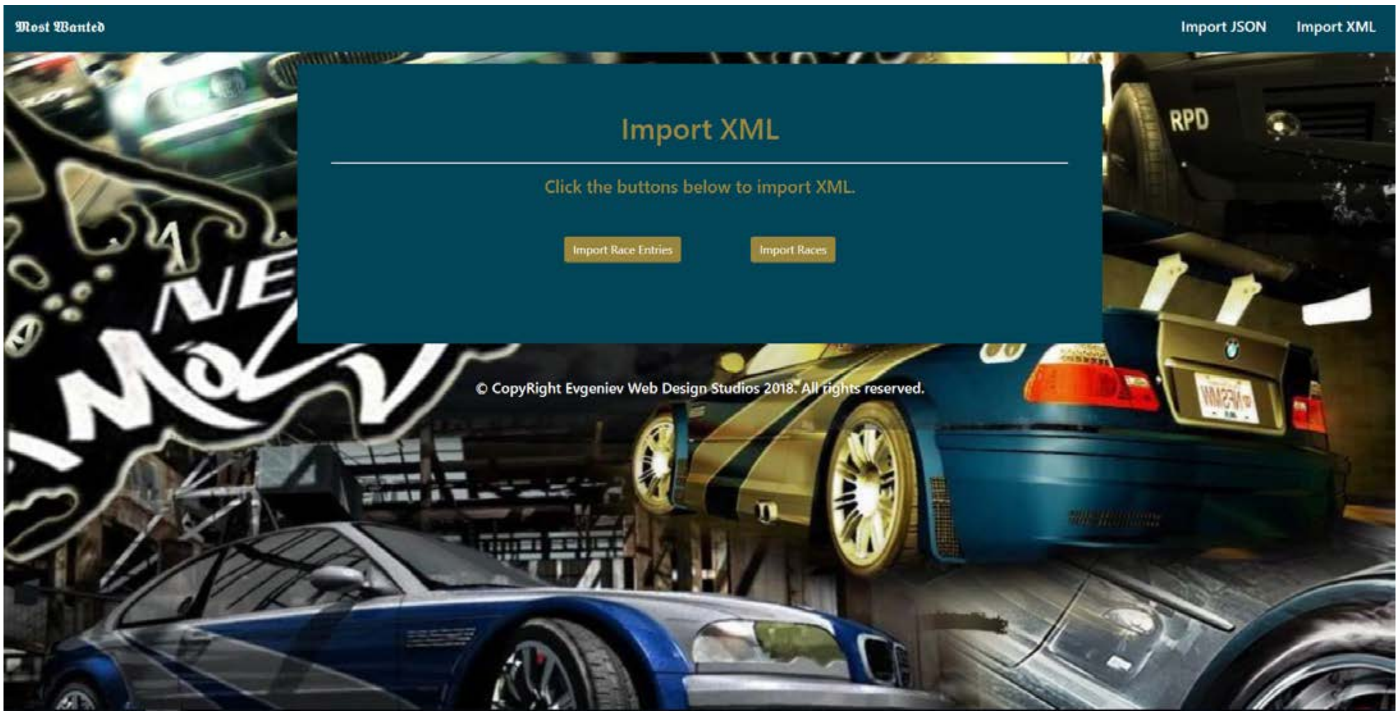
- Home page before importing anything:


Most Wanted
Import JSON
Import XML
Most Wanted
Click the buttons below to export some data from the database.
Import JSON and XML files first.
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

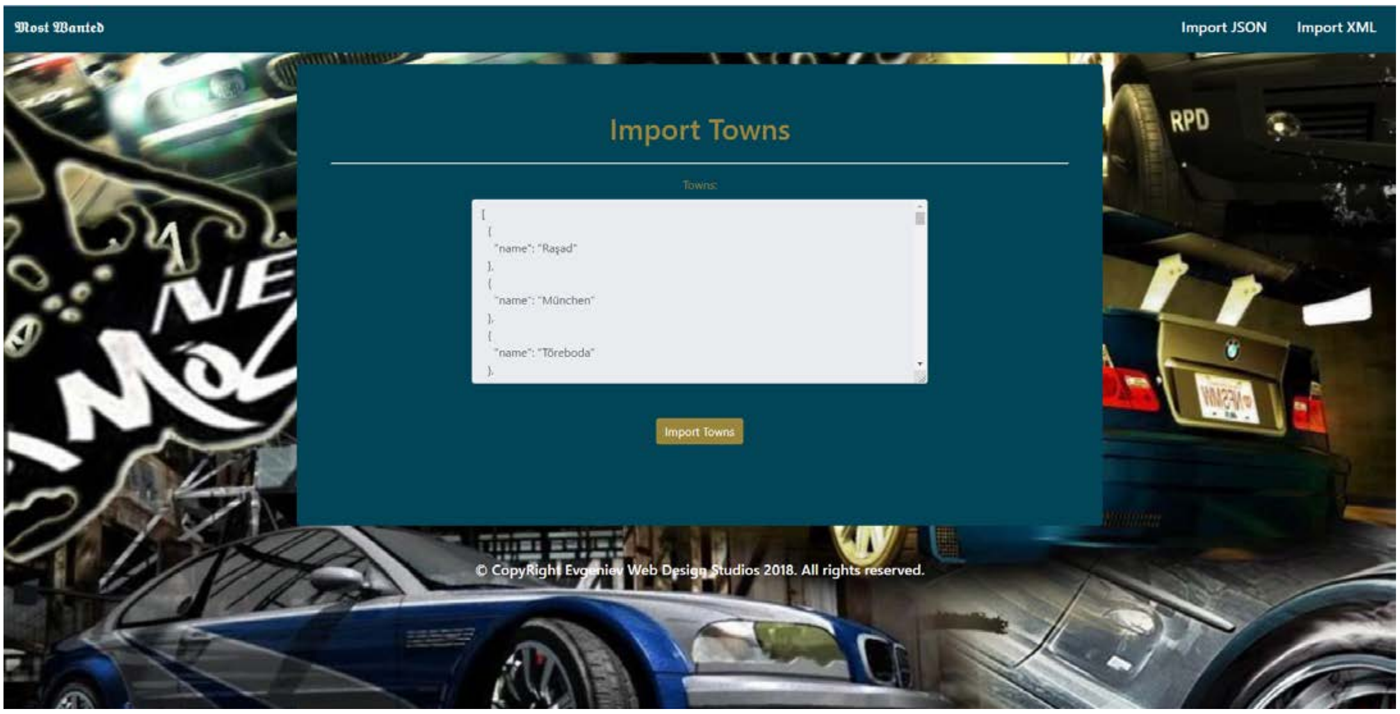
- Import JSON page before importing anything:


Most Wanted
Import JSON
Import XML
Import JSON
Click the buttons below to import JSON.
Import Towns
Import Districts
Import Racers
Import Cars
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Import XML page before importing anything:

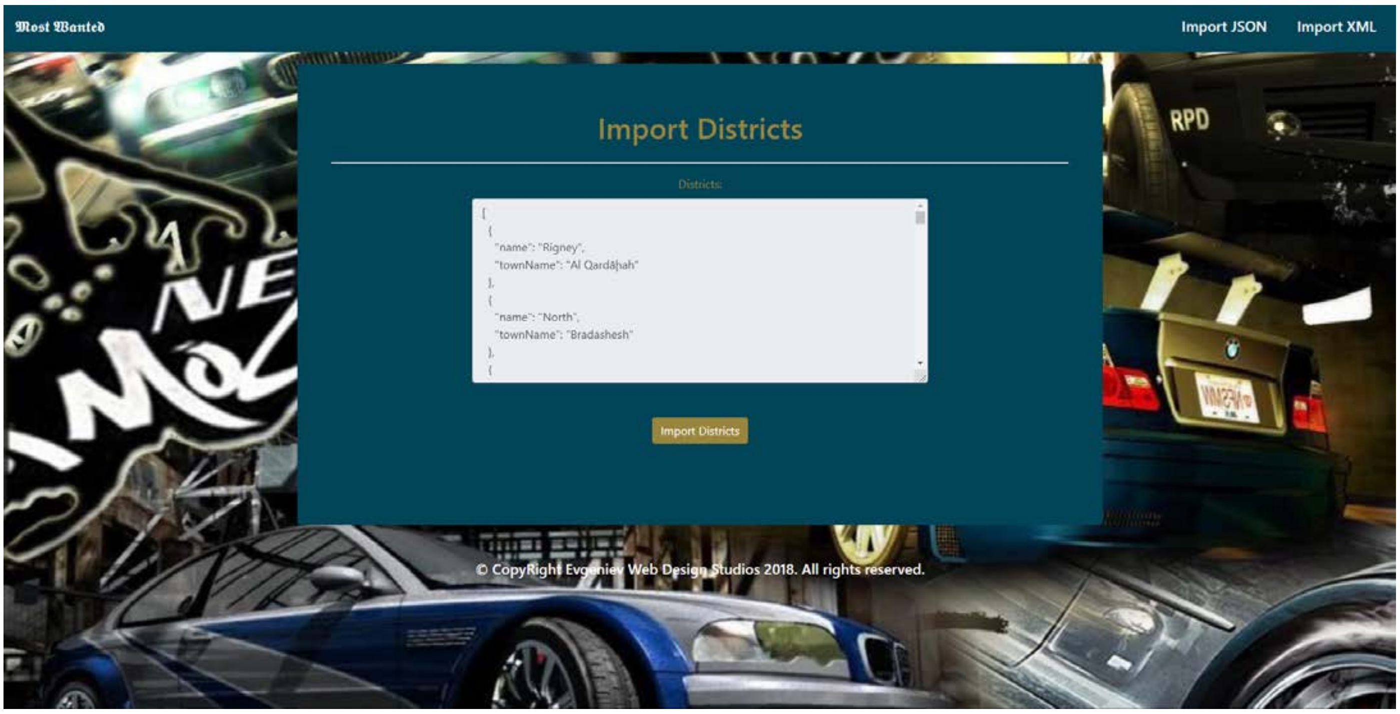

Most Wanted
Import JSON
Import XML
Import XML
Click the buttons below to import XML.
Import Race Entries
Import Races
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Import Towns page after reading the **towns.json** file:


Most Wanted
Import JSON
Import XML
Import Towns
Towns:
Import Towns
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

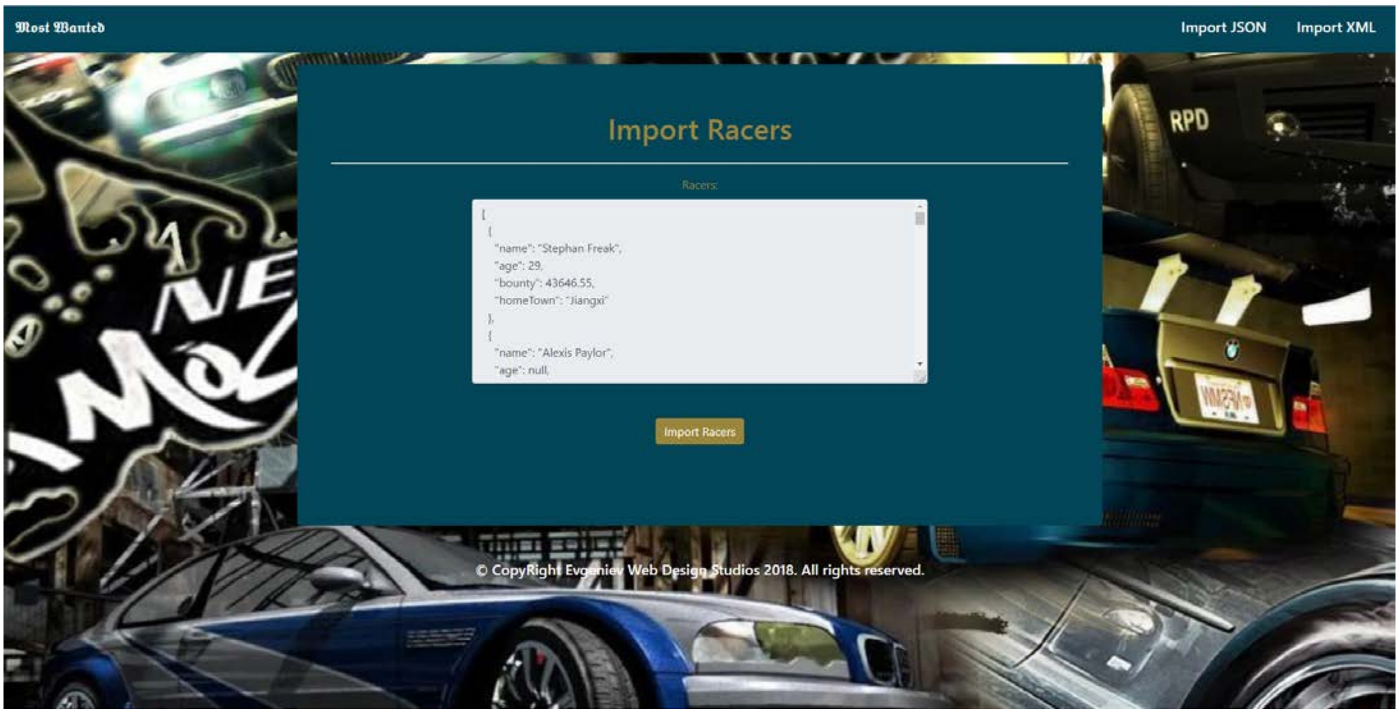
```
[
  {
    "name": "Raşad"
  },
  {
    "name": "München"
  },
  {
    "name": "Töreboda"
  },
```

- Import Districts page after reading the **districts.json** file:



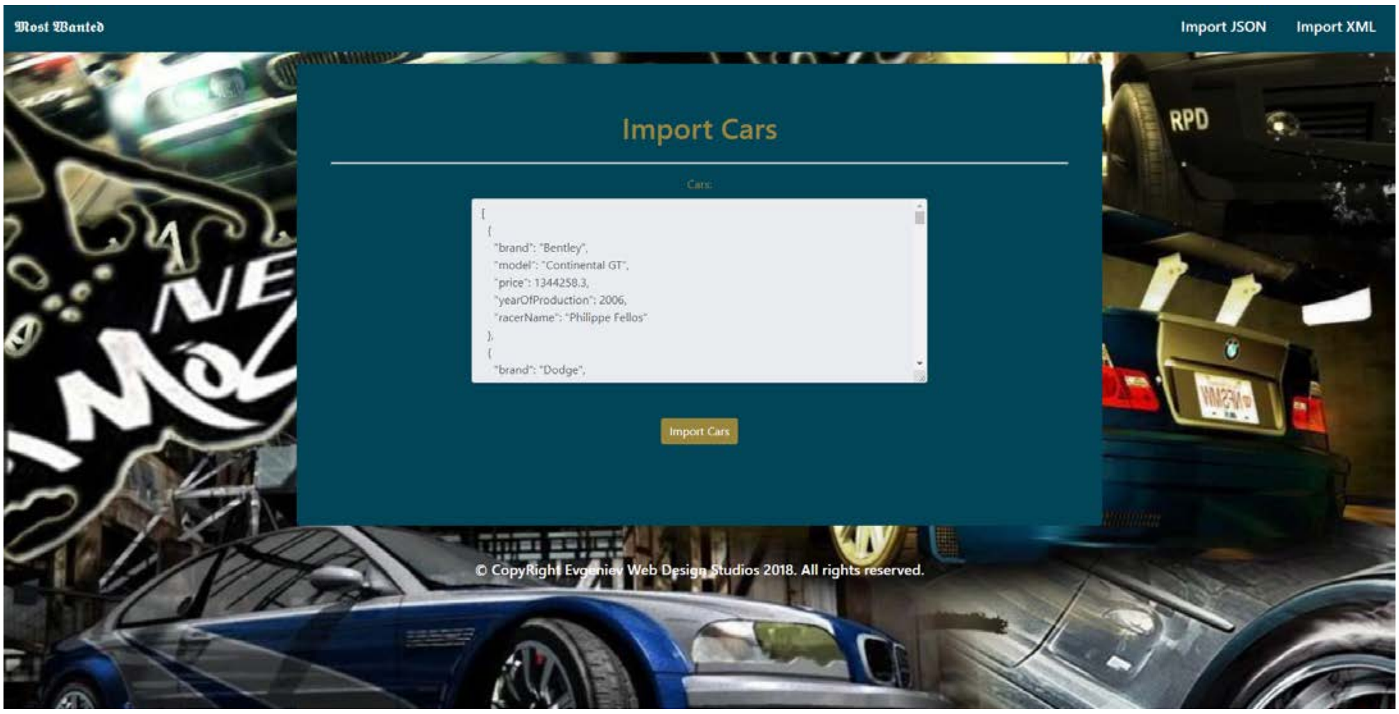
```
[
  {
    "name": "Rigney",
    "townName": "Al Qardāḩah"
  },
  {
    "name": "North",
    "townName": "Bradashesh"
  },
  {
```

- Import Racers page after reading the **racers.json** file:


Most Wanted
Import JSON
Import XML
Import Racers
Racers:
Import Racers
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

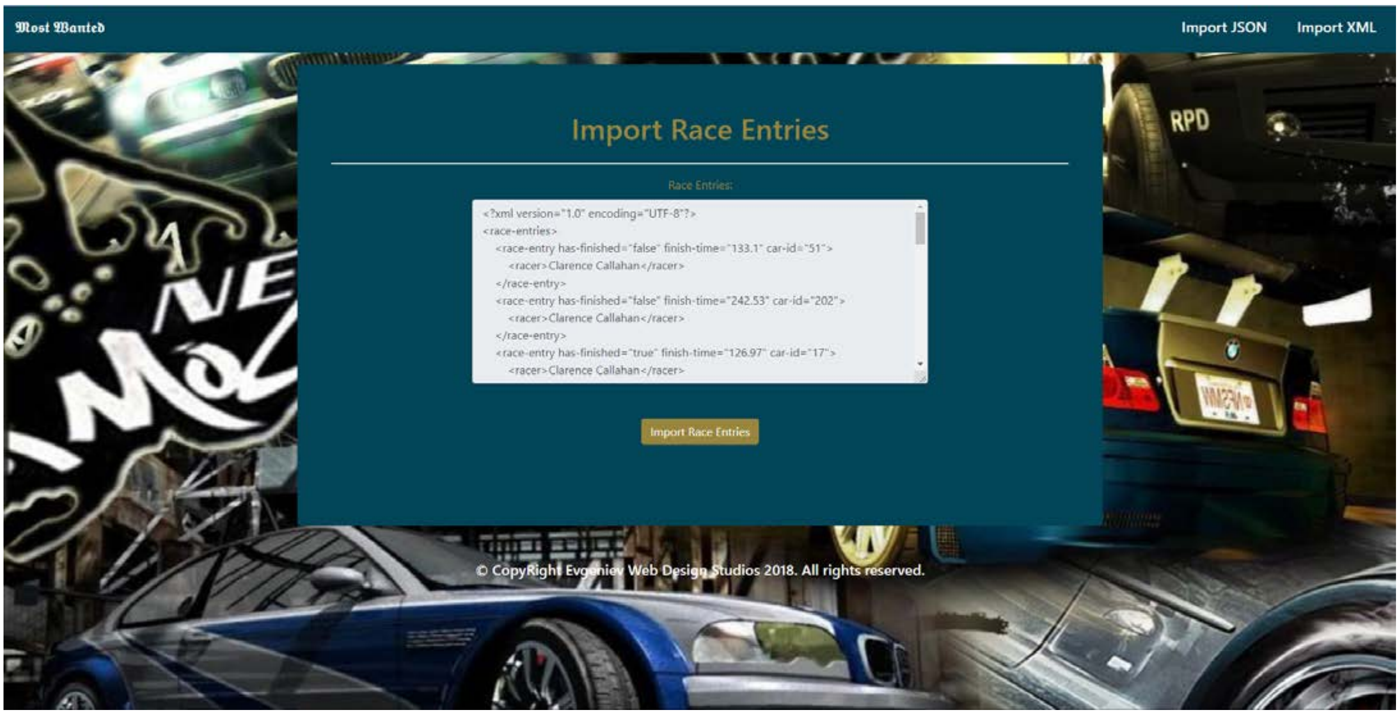
```
[
  {
    "name": "Stephan Freak",
    "age": 29,
    "bounty": 43646.55,
    "homeTown": "Jiangxi"
  },
  {
    "name": "Alexis Paylor",
    "age": null,
```

- Import Cars page after reading the **cars.json** file:


Most Wanted
Import JSON
Import XML
Import Cars
Cars:
Import Cars
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

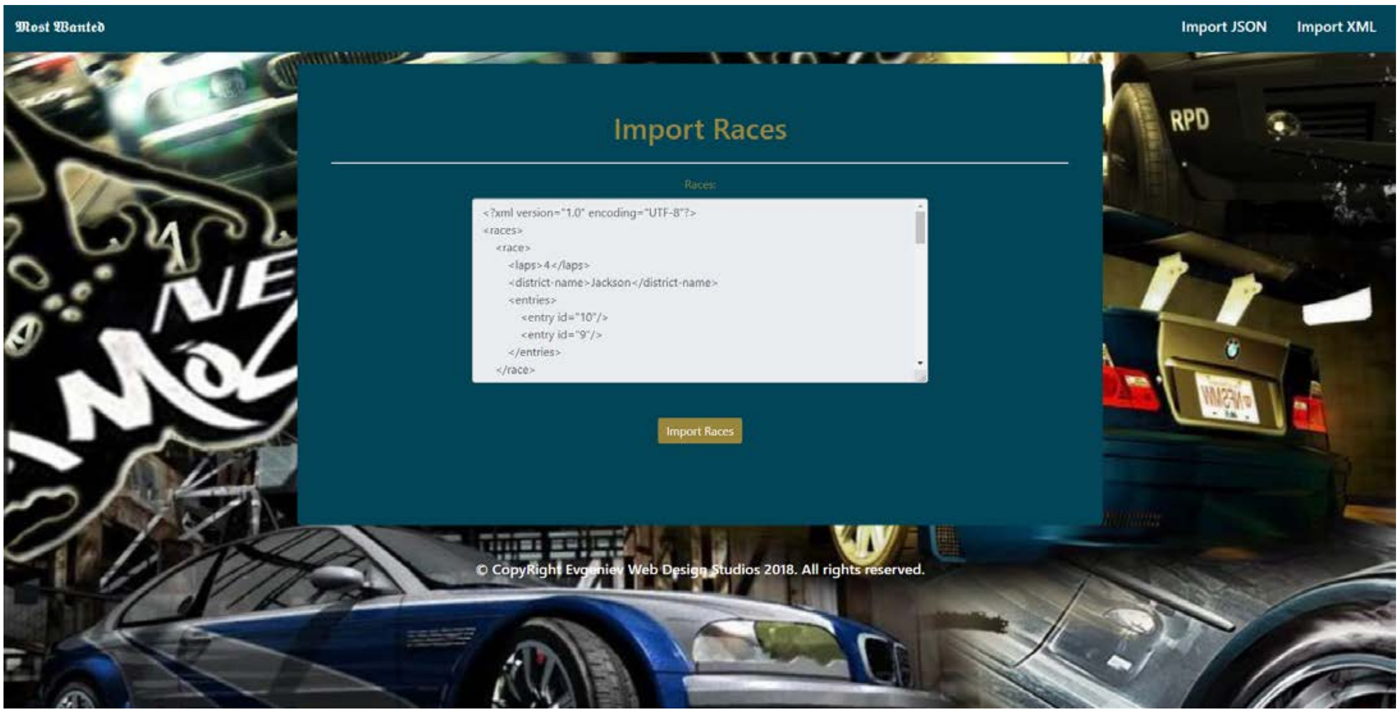
```
[
  {
    "brand": "Bentley",
    "model": "Continental GT",
    "price": 1344258.3,
    "yearOfProduction": 2006,
    "racerName": "Philippe Fellos"
  },
  {
    "brand": "Dodge",
```

- Import Race Entries page after reading the **race-entries.xml** file:



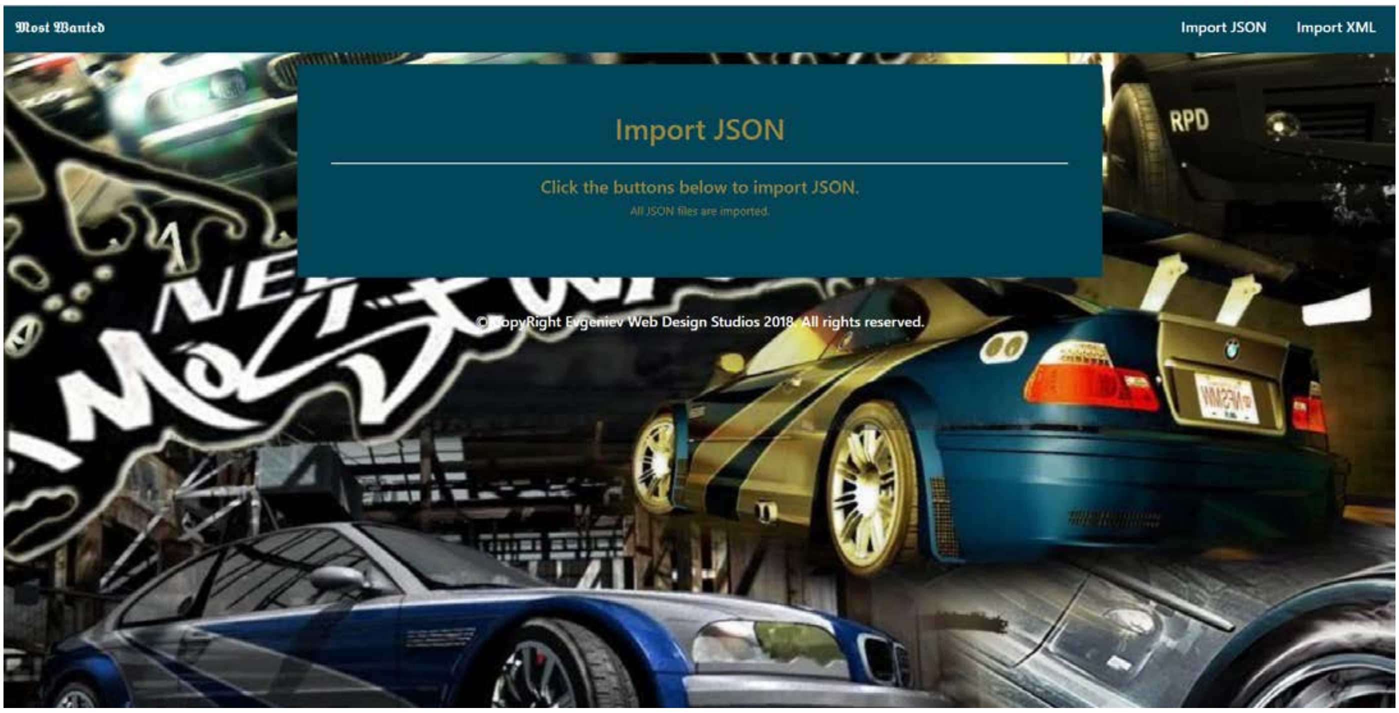
```
<?xml version="1.0" encoding="UTF-8"?>
<race-entries>
  <race-entry has-finished="false" finish-time="133.1" car-id="51">
    <racer>Clarence Callahan</racer>
  </race-entry>
  <race-entry has-finished="false" finish-time="242.53" car-id="202">
    <racer>Clarence Callahan</racer>
  </race-entry>
  <race-entry has-finished="true" finish-time="126.97" car-id="17">
    <racer>Clarence Callahan</racer>
```

- Import Races page after reading the **races.xml** file:


Most Wanted
Import JSON
Import XML
Import Races
Races:
Import Races
© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

```
<?xml version="1.0" encoding="UTF-8"?>
<races>
  <race>
    <laps>4</laps>
    <district-name>Jackson</district-name>
    <entries>
      <entry id="10"/>
      <entry id="9"/>
    </entries>
  </race>
```

- Import JSON page after importing the given data:



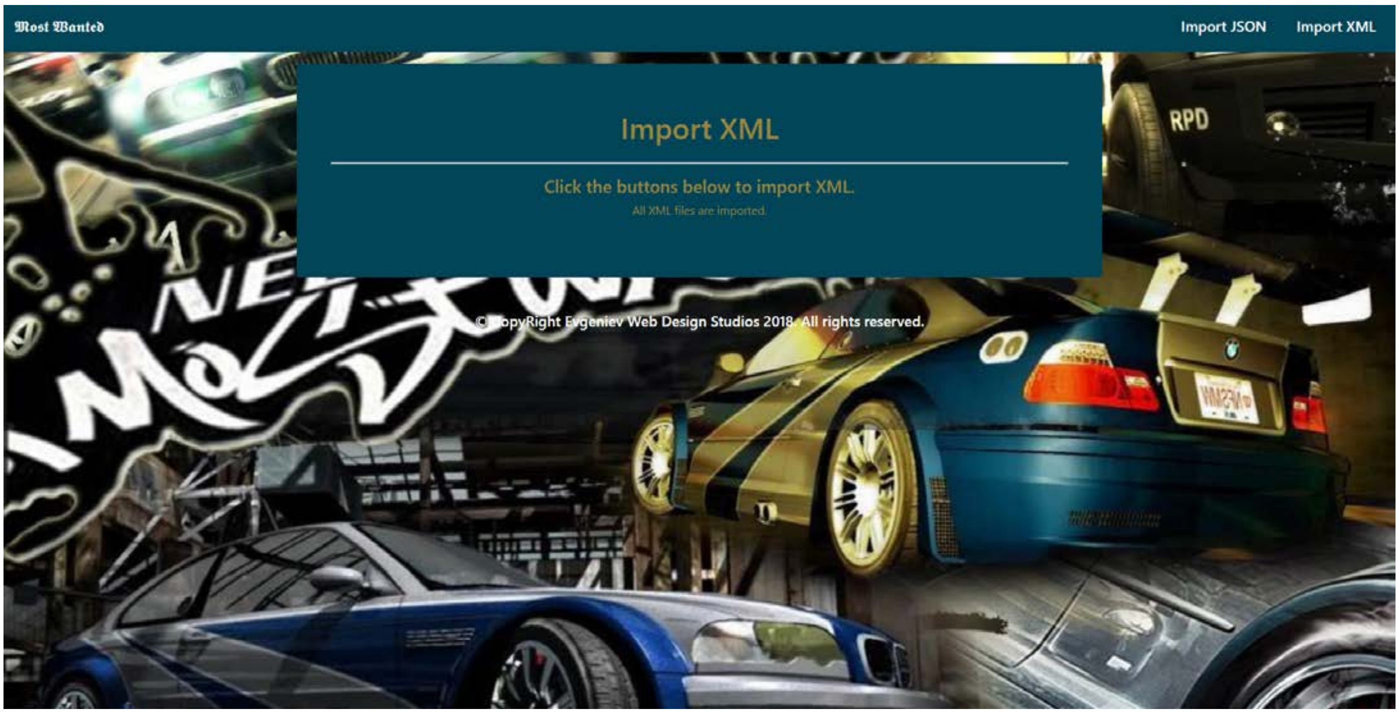
Import JSON

Click the buttons below to import JSON.

All JSON files are imported.

© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Import XML page after importing the given data:



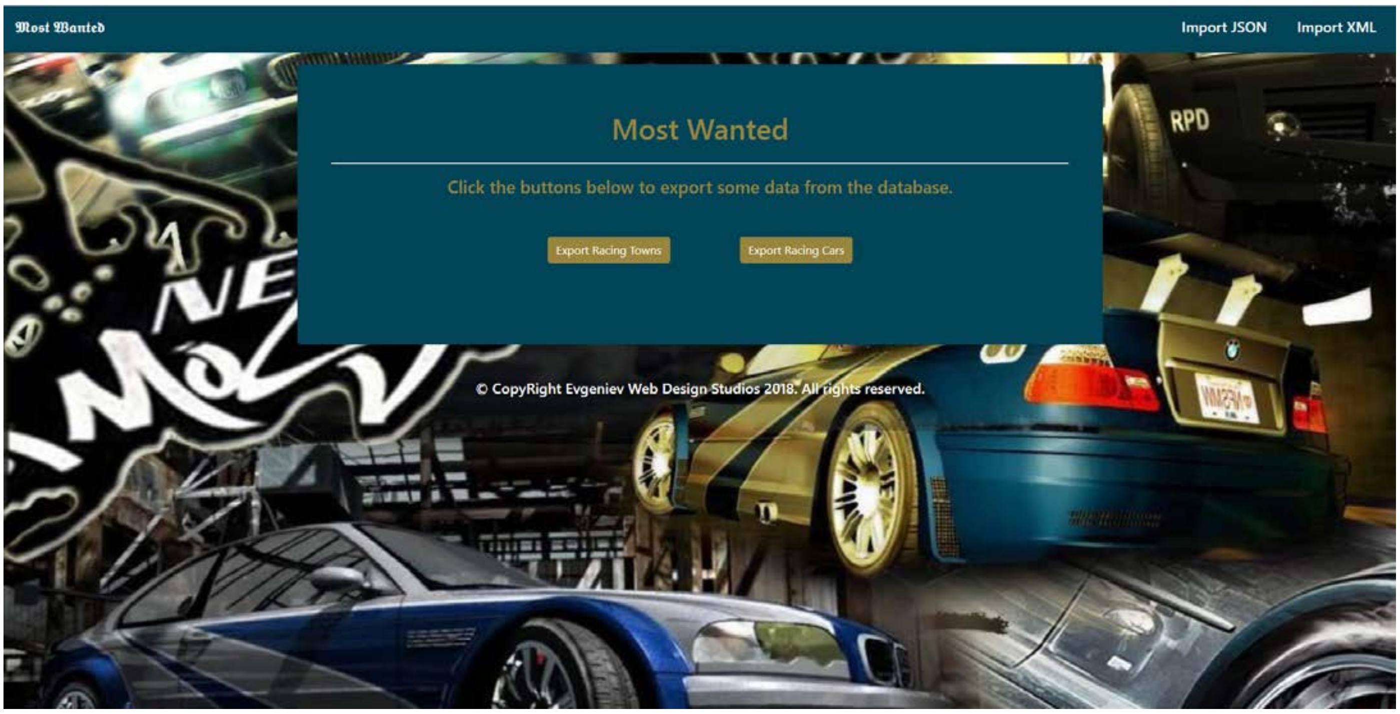
Import XML

Click the buttons below to import XML.

All XML files are imported.

© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Home page after importing the given data:

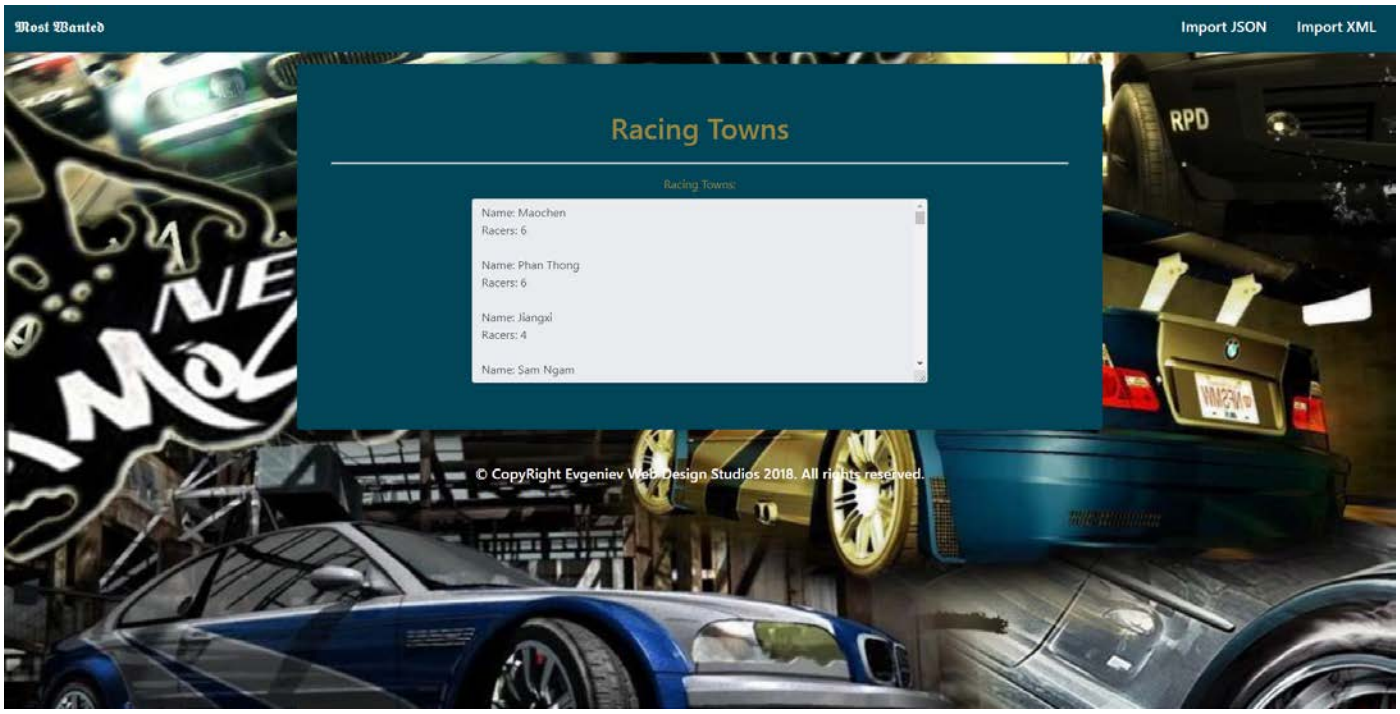


Most Wanted

Click the buttons below to export some data from the database.

© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Export Racing Towns page:

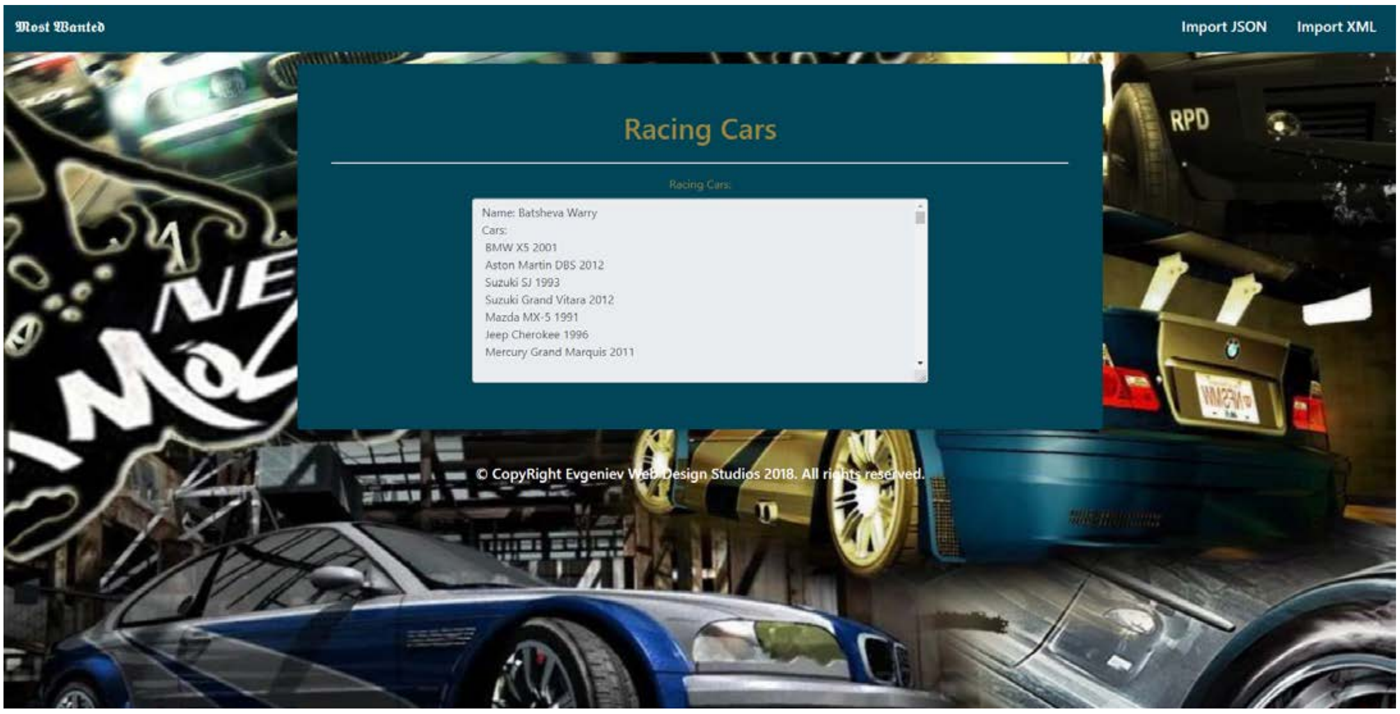


Racing Towns

```
Name: Maochen
Racers: 6

Name: Phan Thong
Racers: 6

Name: Jiangxi
Racers: 4

Name: Sam Ngam
```

© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

- Export Racing Cars page:



```
Name: Batsheva Warry
Cars:
 BMW X5 2001
 Aston Martin DBS 2012
 Suzuki SJ 1993
 Suzuki Grand Vitara 2012
 Mazda MX-5 1991
 Jeep Cherokee 1996
 Mercury Grand Marquis 2011
```

© CopyRight Evgeniev Web Design Studios 2018. All rights reserved.

2. Project Skeleton Overview

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

3. Model Definition

There are 6 main models that the **Most Wanted** database application should contain in its functionality.

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

Town

- **id** – **integer** number, **primary identification field**.
- **name** – a **string** (**required, unique**).

District

- **id** – **integer** number, **primary identification field**.
- **name** – a **string** (**required, unique**).
- **town** – a **Town** entity.

Racer

- **id** – **integer** number, **primary identification field**.
- **name** – a **string** (**required, unique**).
- **age** – an **integer** number.
- **bounty** – a **decimal** data type.
- **homeTown** – a **Town** entity.
- **cars** – a collection of **Car** entity.

Car

- `id` – **integer** number, **primary identification field**.
- `brand` – a **string** (**required**).
- `model` – a **string** (**required**).
- `price` – a **decimal** data type.
- `yearOfProduction` – an **integer** number (**required**).
- `maxSpeed` – a **floating-point** data type.
- `zeroToSixty` – a **floating-point** data type.
- `racer` – a **`Racer`** entity.

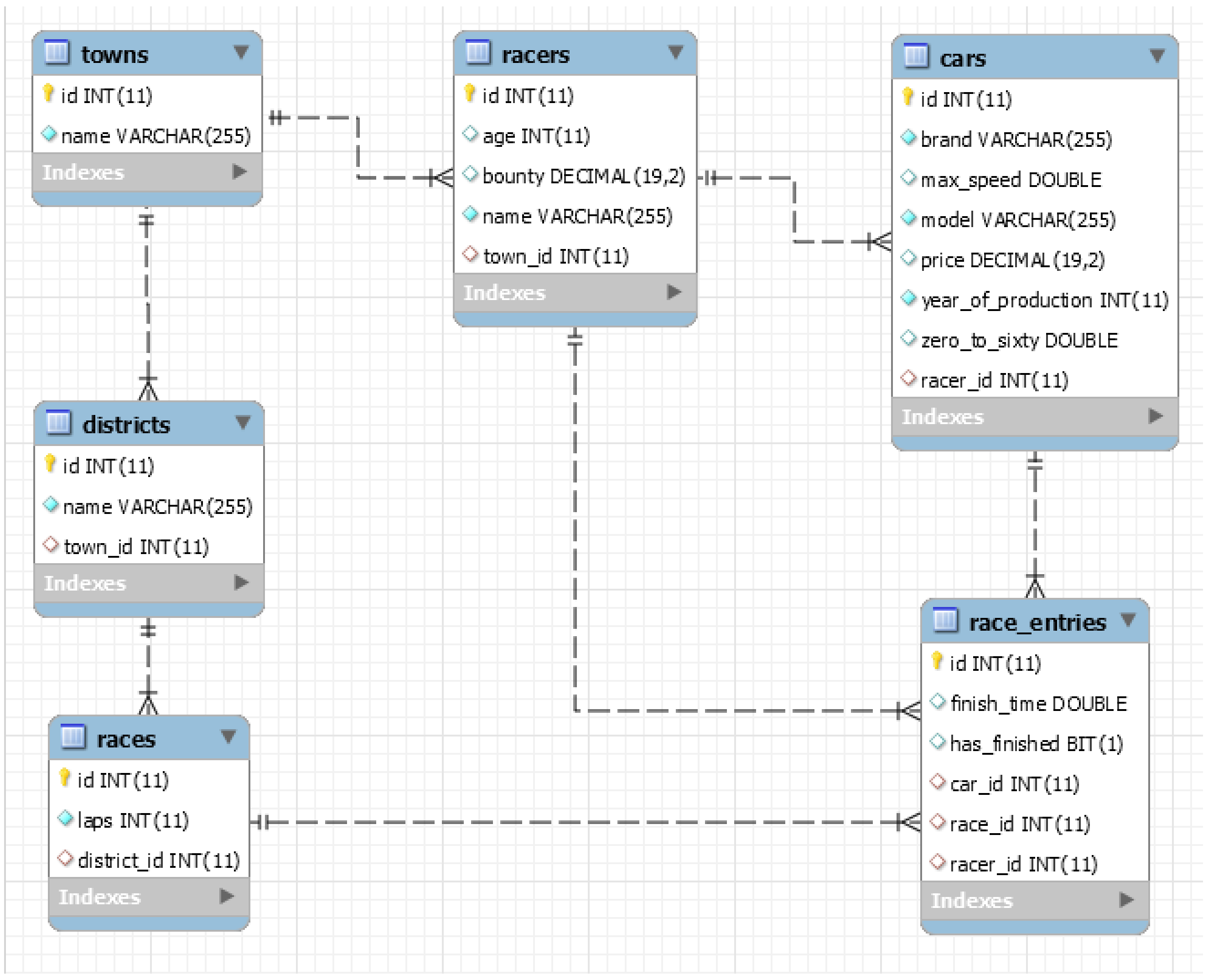
Race

- `id` – **integer** number, **primary identification field**.
- `laps` – **integer** number (**required**, **default – 0**)
- `district` – a **`District`** entity (**required**).
- `entries` – a collection of **`RaceEntry`** entity.

RaceEntry

- `id` – **integer** number, **primary identification field**.
- `hasFinished` – a **boolean** value.
- `finishTime` – a **floating-point** data type.
- `car` – a **`Car`** entity.
- `racer` – a **`Racer`** entity.

You are also given an E/R Diagram for better understanding of the database:



4. Data Import

Use the provided **JSON** and **XML** 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 JSON and XML files.**

ANY UNALLOWED DUPLICATE data should be **ignored** and a message "**`Error: Duplicate Data!`**" should be printed.

ANY INCORRECT data should be **ignored** and a message "**`Error: Incorrect Data!`**" should be printed.

ANY SUCCESSFUL data import should **result** in a message "**`Successfully imported {entityClass} – {entityField}.`**".

The **`entityField`** depends on the **`entityClass`**:

- For **`Towns`**, **`Districts`**, **`Racers`** – **`{name}`**.
- For **`Cars`** – a string **composed** in the following format – "**`{brand} {model} @ {yearOfProduction}`**"
- For **`Race`**, **`RaceEntry`** – **`{id}`**;

JSON Import (10 pts)

Towns (towns.json)

towns.json

```
[
      { "name" : "München" },
      { "name" : "Maastricht" },
      . . .
]
```

```
Successfully imported Town - München.
Successfully imported Town - Maastricht.
. . .
```

Districts (districts.json)

districts.json

```
[
      { "name" : "Larry", "townName" : "Maastricht" },
      { "name" : "Transport", "townName" : "München" },
      . . .
]
```

```
Successfully imported District - Larry.
Successfully imported District - Transport.
. . .
```

Racers (racers.json)

racers.json

```
[
      {
            "name" : "Clarence Callahan",
            "age" : 34,
            "bounty" : 164177.22,
            "townName" : "Troyes"
      },
      . . .
]
```

```
Successfully imported Racer - Clarence Callahan.
. . .
```

Cars (cars.json)

cars.json

```
[
      {
            "brand" : "Volvo",
            "model" : "C70",
            "price" : 487452.02,
            "yearOfProduction" : 2010,
            "maxSpeed" : 161.6,
            "zeroToSixty" : 2.24,
            "racerName" : "Brigit Speller"
      },
      . . .
]
```

```
Successfully imported Car - Volvo C70 @ 2010.
. . .
```

XML Import

The ARPU are not very smart, so their data seeds are quite messed up.
You will need to **import** the **RaceEntries** first, and then import all **Races** with their **entries**.

Race Entries (race-entries.xml)

race-entries.xml

```
<?xml version="1.0" encoding="utf-8"?>
<race-entries>
```

```
    <race-entry has-finished="true" finish-time="741.12" car-id="269">
        <racer>Max Philpott</racer>
    </race-entry>
    <race-entry has-finished="false" finish-time="822.96" car-id="242">
        <racer>Wylie Gareisr</racer>
    </race-entry>
    <race-entry has-finished="true" finish-time="156.57" car-id="220">
        <racer>Murial Jedrzejewicz</racer>
    </race-entry>
    ...
</race-entries>
```

```
Successfully imported RaceEntry - 1.
Successfully imported RaceEntry - 2.
Successfully imported RaceEntry - 3.
. . .
```

Races (races.xml)

races.xml

```
<?xml version="1.0" encoding="utf-8"?>
<races>
    <race>
        <laps>4</laps>
        <district-name>Jackson</district-name>
        <entries>
            <entry id="10">
            <entry id="9">
        </entries>
    </race>
    ...
</races>
```

```
Successfully imported Race - 1.
. . .
```

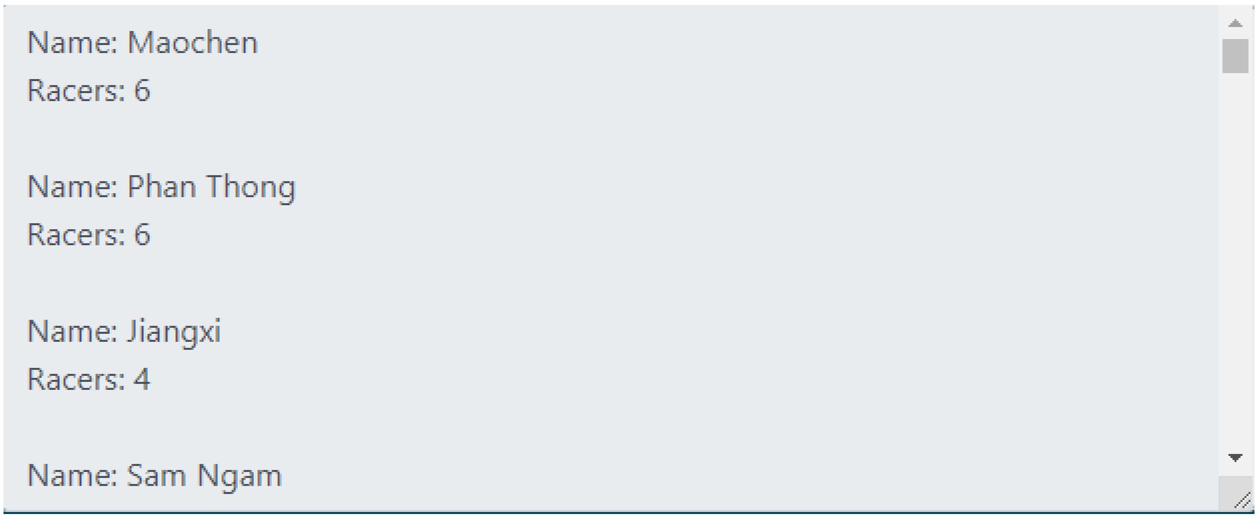
5. Data Export

Get ready to export the data you've imported in the previous task. Here you will have some pretty complex database querying.

Racing Towns

Export all towns which have **any racers** in them:

- **Export** only the **town name** (as **name**) and **count of racers** (as `racers`).
- **Order** them **descending**, by **count** of **racers** they have, and then by **town name alphabetically**.

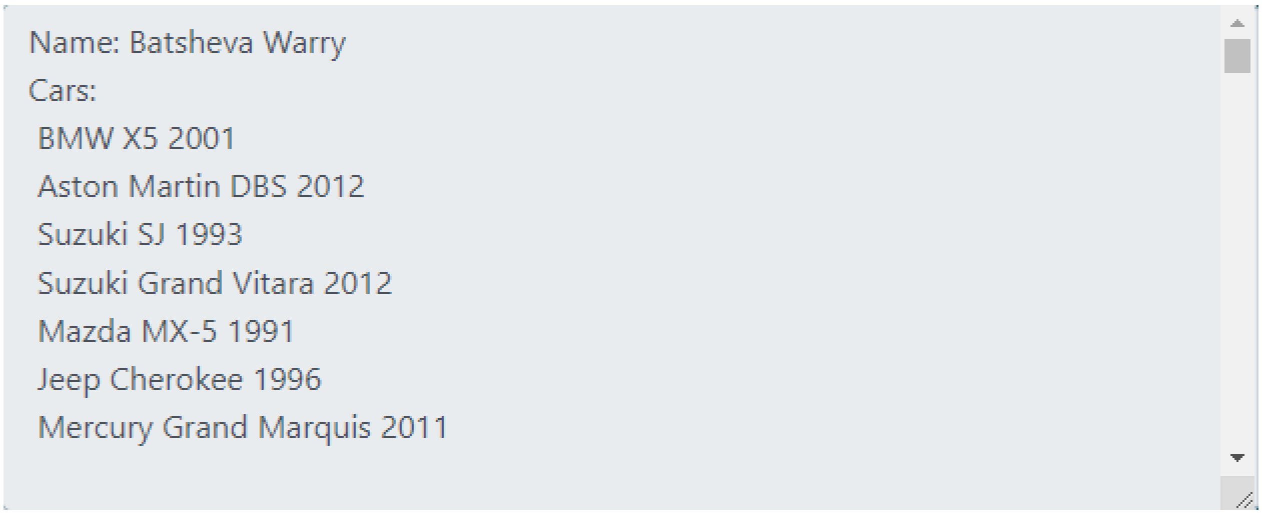


```
Name: Maochen
Racers: 6

Name: Phan Thong
Racers: 6

Name: Jiangxi
Racers: 4

Name: Sam Ngam
```

Racing Cars

Export all racers which **have any cars**:

- **Export** the racer's **name**, **age** (but **ONLY** if it is **NOT NULL**), **list** of **cars**.
  - In case the **racer's age** property is **NULL**, do **NOT** include it.
- The **cars** should be **strings** in the following format: "`{brand} {model} {yearOfProduction}`".
- **Order** them **descending**, by **count** of **cars** they have, and then by **racer name alphabetically**.



```
Name: Batsheva Warry
Cars:
 BMW X5 2001
 Aston Martin DBS 2012
 Suzuki SJ 1993
 Suzuki Grand Vitara 2012
 Mazda MX-5 1991
 Jeep Cherokee 1996
 Mercury Grand Marquis 2011
```