TECHNICAL REPORT: IDB1

Sochi 2014 Winter Olympics

by “4 Is Better Than 6”

Members:

- Austin Hooper
- Ardhimas Kamdani
- Stewart Johnson
- Nickson Dalimarta

INTRODUCTION

Description:

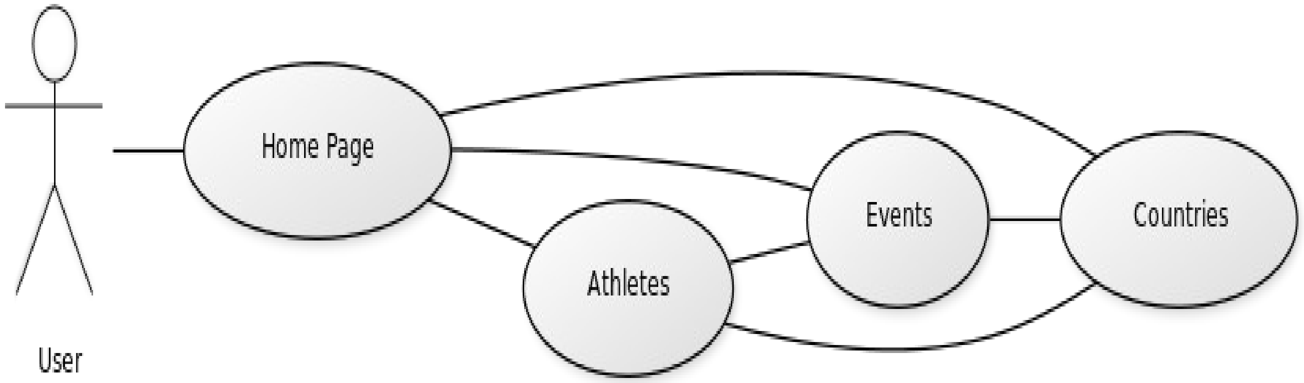
Have you ever found yourself wanting to know the events that went on during the 2014 Sochi Winter Olympics? Or the athletes that took part in said games? How about how well your favorite country did in said events? Then we have a site that is right up your alley!

Use cases (Phase 1) :

* From the Home Page a user can use the navigation bar to be directed to any Athlete, Event, and Country page.

* Navigation Bar exists in all pages. In phase 1, because of limited pages, a user may go to any page by making use of the navigation bar.

Use Case Diagram



DESIGN

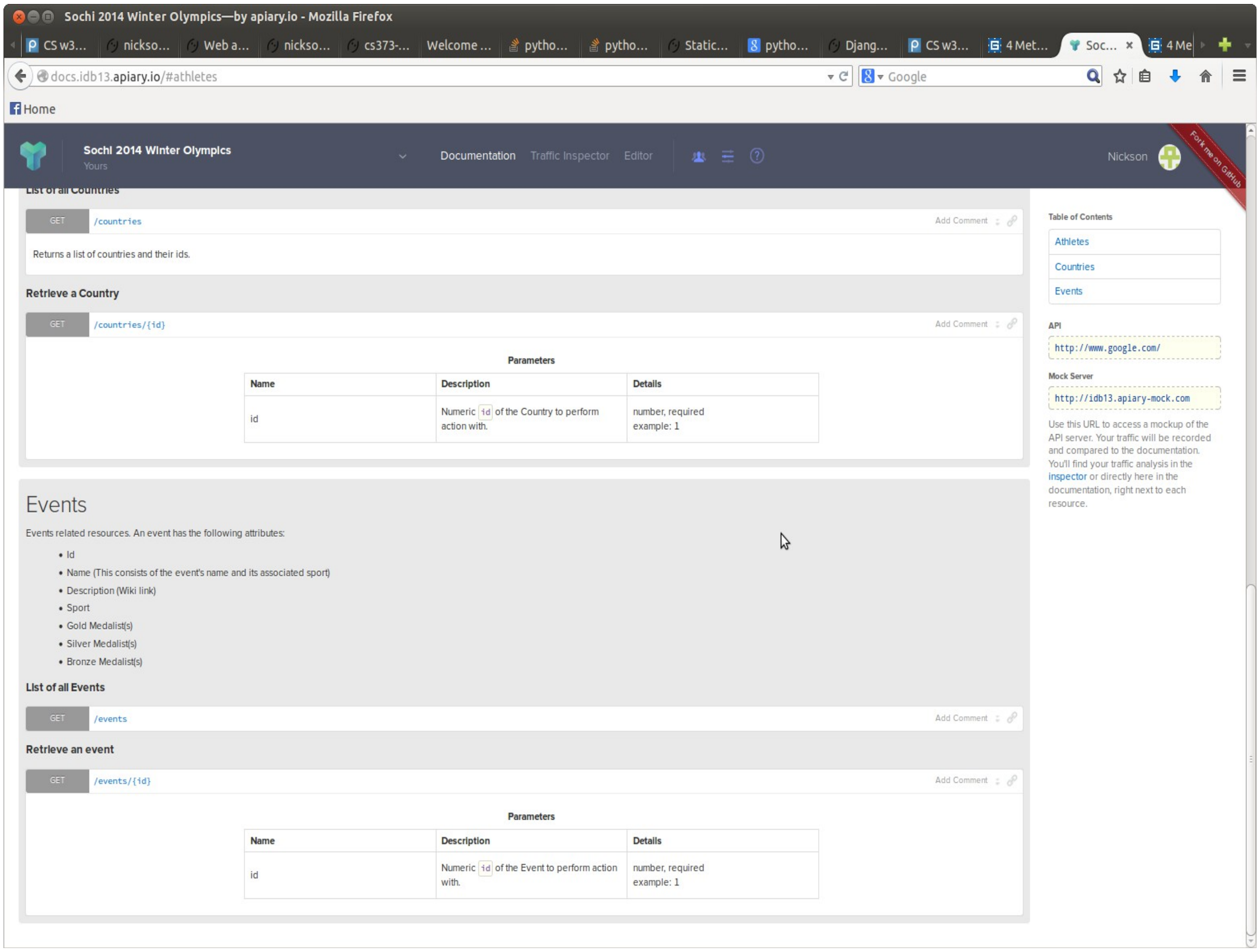
RESTful API:

API exists for retrieving information of all available athletes/ events/ countries. Due to the limited functionality, there are currently only 2 GET methods for each model. One to retrieve a list of all available datas for a model. Each element is a key-value pair of the model's id and name, separated by an equal sign. The other method is used retrieve the details of an object of a specified id (the only parameter of this method). A user who would like to get the information on Ireen Wust, for example, may first use the first method to get this list

```
"ID=Athletes":["1=Meryl Davis","2=Tobias Arit","3=Ireen Wust"]
```

He can then use the second method to get these details

```
{
    "Id":3,
    "First Name":"Ireen",
    "Last Name":"Wust",
    "Gender":"Female",
    "Nationality":"Netherlands",
    "Date of Birth":"1 April 1986",
    "Age":28,
    "Height":"1.68m (5'6")",
    "Weight":"63 kg (139 lbs)",
    "Place of birth":"Goirle, Netherlands",
    "Total Medals":5,
    "Gold Medals":["Speed Skating:Ladies' 3000m", "Speed Skating:Ladies' Team
      Pursuit"]
    "Silver Medals":["Speed Skating:Ladies' 1000m", "Speed Skating:Ladies' 1500m",  "Speed
Skating:Ladies' 5000m"]
    "Bronze Medals":[]
}
```



List of all Countries

GET /countries

Returns a list of countries and their ids.

Retrieve a Country

GET /countries/{id}

Parameters

Name	Description	Details
id	Numeric id of the Country to perform action with.	number, required example: 1

Events

Events related resources. An event has the following attributes:

- Id
- Name (This consists of the event's name and its associated sport)
- Description (Wiki link)
- Sport
- Gold Medalist(s)
- Silver Medalist(s)
- Bronze Medalist(s)

List of all Events

GET /events

Retrieve an event

GET /events/{id}

Parameters

Name	Description	Details
id	Numeric id of the Event to perform action with.	number, required example: 1

Table of Contents

- Athletes
- Countries
- Events

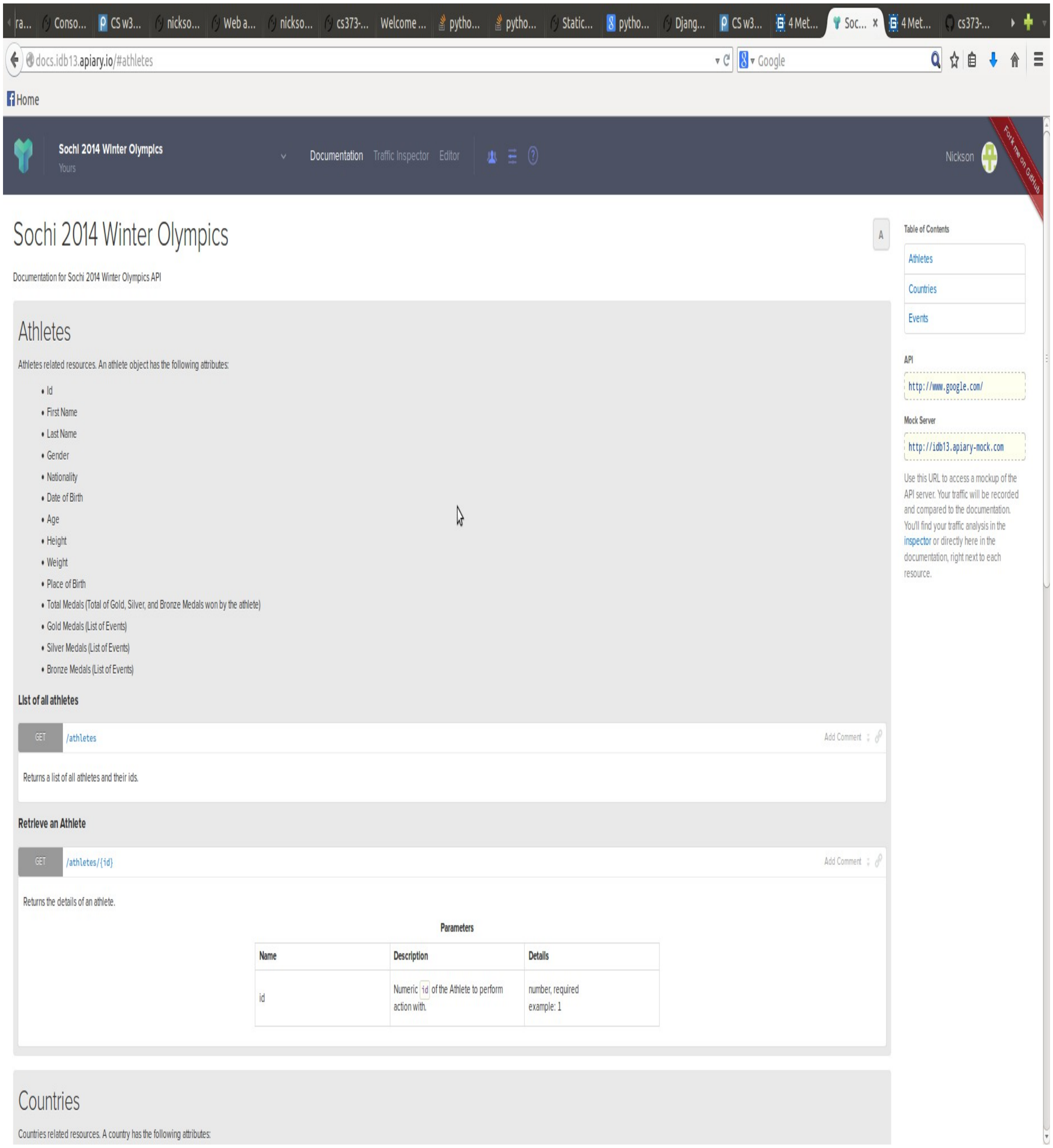
API

`http://www.google.com/`

Mock Server

`http://idb13.apiary-mock.com`

Use this URL to access a mockup of the API server. Your traffic will be recorded and compared to the documentation. You'll find your traffic analysis in the inspector or directly here in the documentation, right next to each resource.



Sochi 2014 Winter Olympics

Documentation for Sochi 2014 Winter Olympics API

Athletes

Athletes related resources. An athlete object has the following attributes:

- Id
- First Name
- Last Name
- Gender
- Nationality
- Date of Birth
- Age
- Height
- Weight
- Place of Birth
- Total Medals (Total of Gold, Silver, and Bronze Medals won by the athlete)
- Gold Medals (List of Events)
- Silver Medals (List of Events)
- Bronze Medals (List of Events)

List of all athletes

GET /athletes

Returns a list of all athletes and their ids.

Retrieve an Athlete

GET /athletes/{id}

Returns the details of an athlete.

Parameters

Name	Description	Details
id	Numeric id of the Athlete to perform action with.	number, required example: 1

Countries

Countries related resources. A country has the following attributes:

Table of Contents

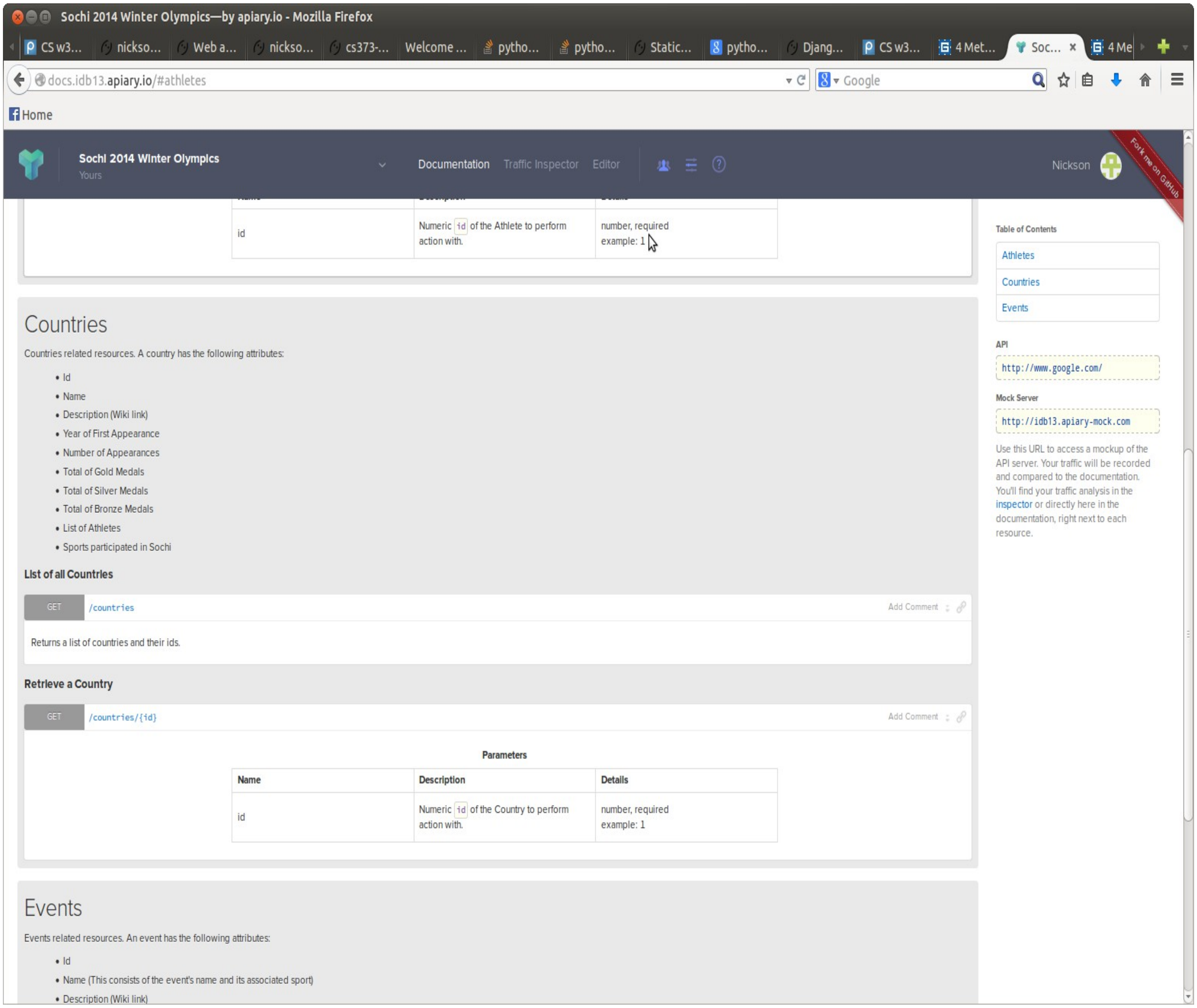
API

http://www.google.com/

Mock Server

http://idb13.apiary-mock.com

Use this URL to access a mockup of the API server. Your traffic will be recorded and compared to the documentation. You'll find your traffic analysis in the inspector or directly here in the documentation, right next to each resource.


Sochi 2014 Winter Olympics—by apiary.io - Mozilla Firefox
CS w3… | nickso… | Web a… | nickso… | cs373-… | Welcome … | pytho… | pytho… | Static… | pytho… | Djang… | CS w3… | 4 Met… | Soc… | 4 Me
docs.idb13.apiary.io/#athletes
Google
Home
Sochi 2014 Winter Olympics
Yours
Documentation
Traffic Inspector
Editor
Nickson
Fork me on GitHub

id	Numeric `id` of the Athlete to perform action with.	number, required example: 1

Countries

Countries related resources. A country has the following attributes:

- Id
- Name
- Description (Wiki link)
- Year of First Appearance
- Number of Appearances
- Total of Gold Medals
- Total of Silver Medals
- Total of Bronze Medals
- List of Athletes
- Sports participated in Sochi

List of all Countries

GET `/countries`

Returns a list of countries and their ids.

Retrieve a Country

GET `/countries/{id}`

Parameters

Name	Description	Details
id	Numeric `id` of the Country to perform action with.	number, required example: 1

Events

Events related resources. An event has the following attributes:

- Id
- Name (This consists of the event's name and its associated sport)
- Description (Wiki link)

API

`http://www.google.com/`

Mock Server

`http://idb13.apiary-mock.com`

Use this URL to access a mockup of the API server. Your traffic will be recorded and compared to the documentation. You'll find your traffic analysis in the inspector or directly here in the documentation, right next to each resource.

Django Models & Attributes:

1. ATHLETES

   * First Name

     - Primary Key

     - Athlete's first name

     - Type: String

   * Last Name

     - Primary Key

     - Athlete's last name

     - String

   * Country

     - Athlete's Country

     - String

   * Gender

     - Athlete's Gender

     - String

   * Date of Birth

     - Primary Key

     - Athlete's Date of birth

     - Date

   * Gold Medals

- List of events the athlete won the gold medal from

- Foreign Key

* Silver Medals: Events

- List of events the athlete won the silver medal from

- Foreign Key

* Bronze Medals: Events

- List of events the athlete won the bronze medal from

- Foreign Key

FirstName, LastName, and DateofBirth are chosen as the primary keys to prevent conflicts. It is highly unlikely that participating athletes have the same name and date of birth.

2. <u>COUNTRIES</u>

* Name

- Primary Key

- Name of the coutnry

- String

* Athlete List

- List of participating athletes from the country

- String

* Description

  - Brief description of the country
  - String
* Gold Medals Won
  - Total of gold medals won by the country
  - Dynamically calculated from the medals won by the athletes
  - Integer
* Silver Medals Won
  - Total of silver medals won by the country
  - Dynamically calculated from the medals won by the athletes
* Bronze Medals Won
  - Total of bronze medals won by the country
  - Dynamically calculated from the medals won by the athletes

3. EVENTS

* Name
  - Primary Key
  - Name of the associated sport followed by the name of the event
  - String
* Sport
  - Primary Key
  - Name of the associated sport

  - String

* Description

  - A brief description of the event

  - String

* Gold Medalist(s)

  - The name of the gold medalist(s). Can be multiple athletes.

  - Foreign Key

* Silver Medalist(s)

  - The name of the silver medalist(s). Can be multiple athletes.

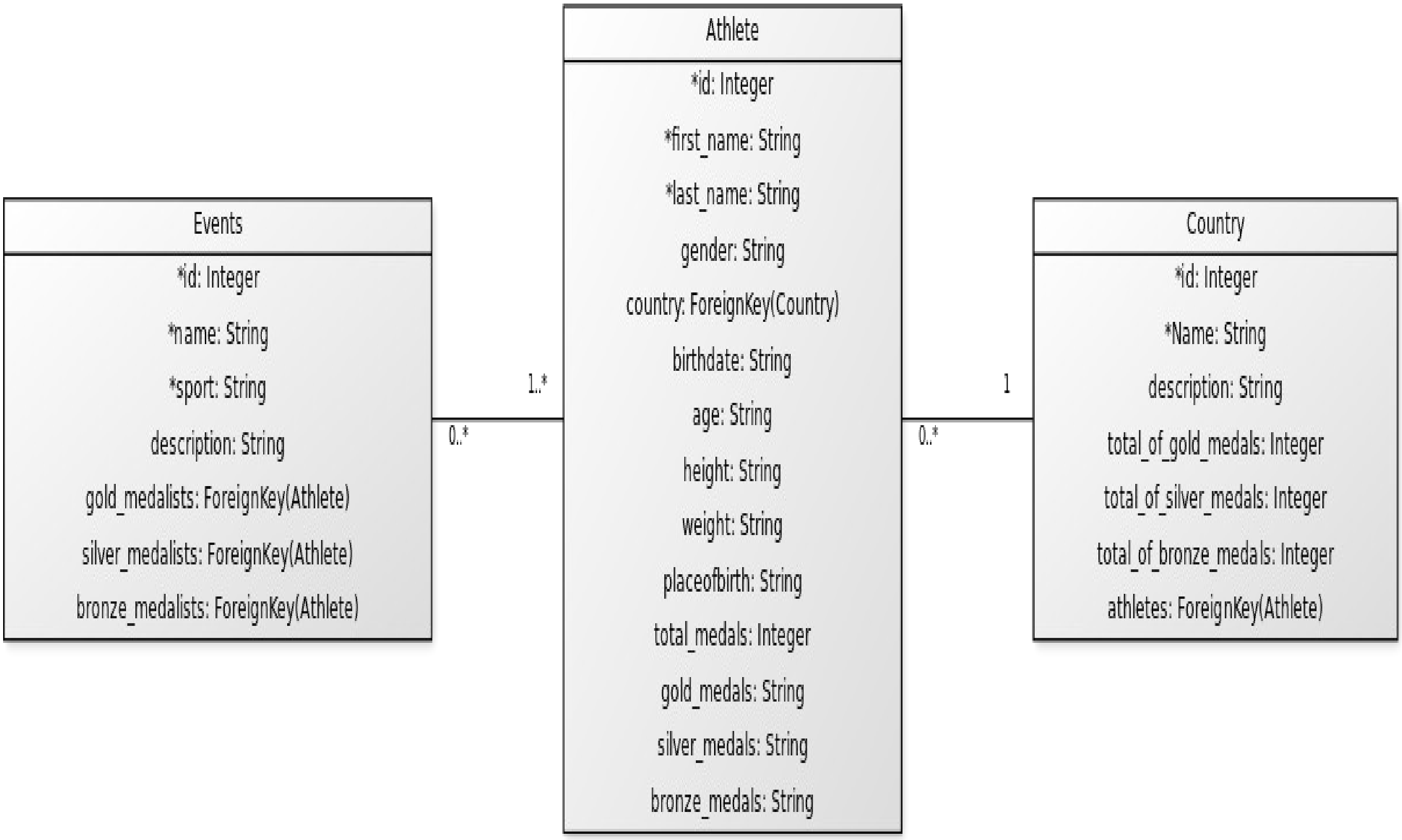
  - Foreign Key

* Bronze Medalist(s)

- The name of the bronze medalist(s). Can be multiple athletes.

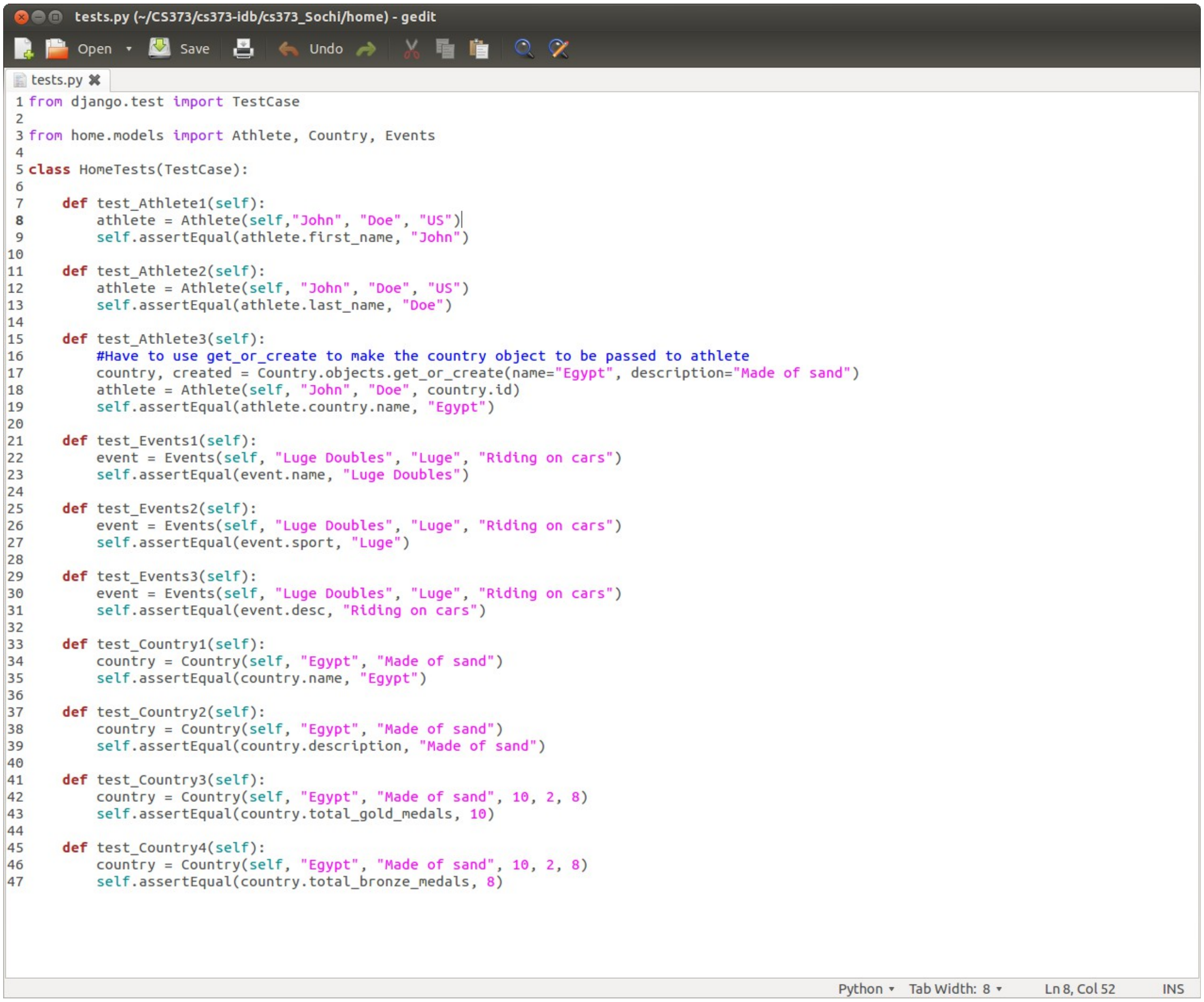
- Foreign Key

Django Models Diagram



TESTS

Django Models:



```python
from django.test import TestCase

from home.models import Athlete, Country, Events

class HomeTests(TestCase):

    def test_Athlete1(self):
        athlete = Athlete(self,"John", "Doe", "US")
        self.assertEqual(athlete.first_name, "John")

    def test_Athlete2(self):
        athlete = Athlete(self, "John", "Doe", "US")
        self.assertEqual(athlete.last_name, "Doe")

    def test_Athlete3(self):
        #Have to use get_or_create to make the country object to be passed to athlete
        country, created = Country.objects.get_or_create(name="Egypt", description="Made of sand")
        athlete = Athlete(self, "John", "Doe", country.id)
        self.assertEqual(athlete.country.name, "Egypt")

    def test_Events1(self):
        event = Events(self, "Luge Doubles", "Luge", "Riding on cars")
        self.assertEqual(event.name, "Luge Doubles")

    def test_Events2(self):
        event = Events(self, "Luge Doubles", "Luge", "Riding on cars")
        self.assertEqual(event.sport, "Luge")

    def test_Events3(self):
        event = Events(self, "Luge Doubles", "Luge", "Riding on cars")
        self.assertEqual(event.desc, "Riding on cars")

    def test_Country1(self):
        country = Country(self, "Egypt", "Made of sand")
        self.assertEqual(country.name, "Egypt")

    def test_Country2(self):
        country = Country(self, "Egypt", "Made of sand")
        self.assertEqual(country.description, "Made of sand")

    def test_Country3(self):
        country = Country(self, "Egypt", "Made of sand", 10, 2, 8)
        self.assertEqual(country.total_gold_medals, 10)

    def test_Country4(self):
        country = Country(self, "Egypt", "Made of sand", 10, 2, 8)
        self.assertEqual(country.total_bronze_medals, 8)
```

Simple tests that test if objects were created successfully.