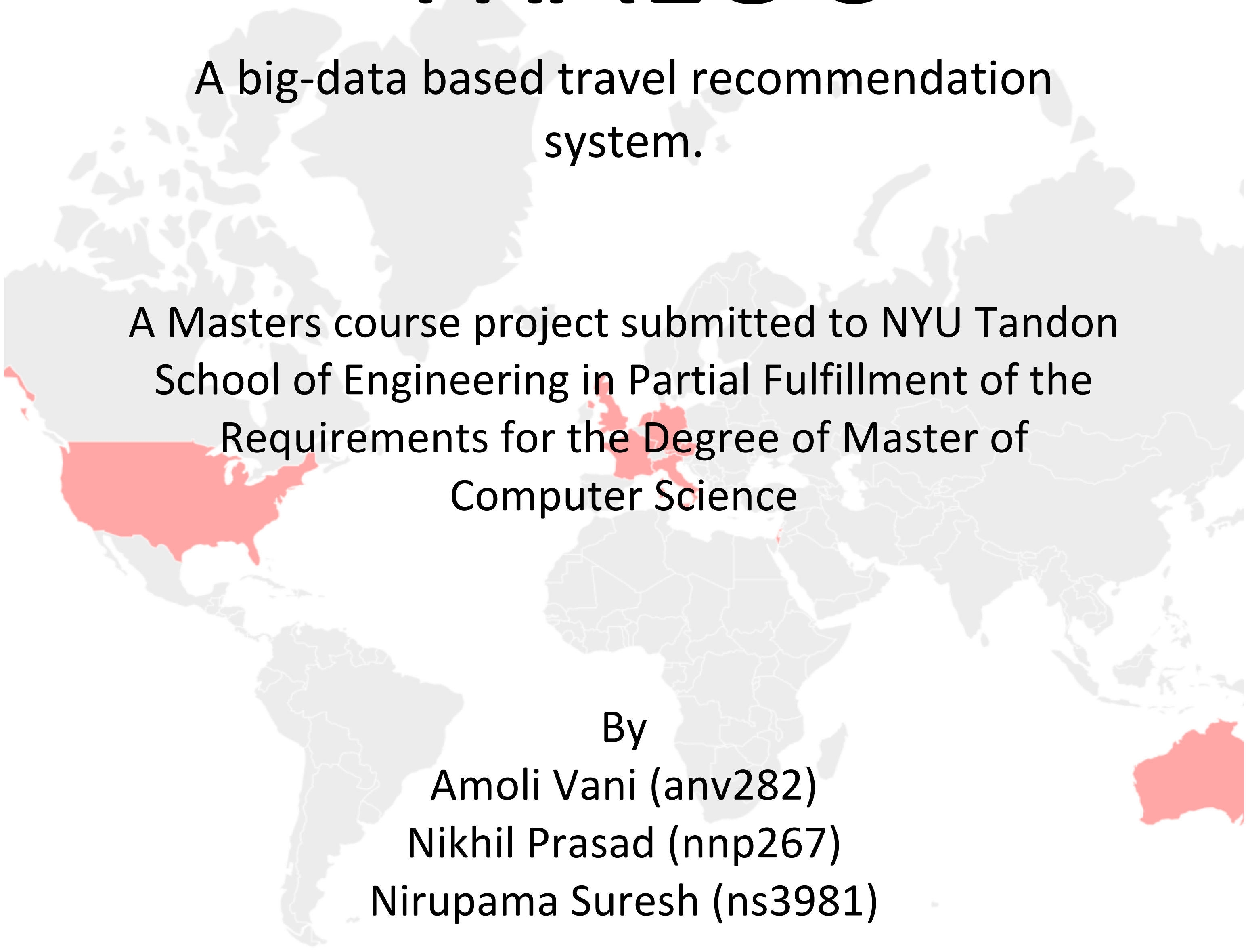
TRALOC

A big-data based travel recommendation system.



A Masters course project submitted to NYU Tandon School of Engineering in Partial Fulfillment of the Requirements for the Degree of Master of Computer Science

By
Amoli Vani (anv282)
Nikhil Prasad (nnp267)
Nirupama Suresh (ns3981)

May 2018

Acknowledgement

We would like to thank Professor Juan Rodriguez for his teaching, enthusiasm and motivating us to learn new technologies throughout the course. We would also like to thank the TAs for their guidance and assistance as well.

Abstract

Everyone loves going on vacation, even if it is just for a short while. Some holidays are planned for weeks in advance and some are as impromptu as they can be. The planning for this can be quite tedious, however. People often need personalized information about destinations, accommodations, flights, activities, food, etc. whilst wanting to remain on a budget. Current tourist recommendation systems are not able to collect all information from different resources.

Examining travel blogs is a step in the right direction as they often contain personalized details of tourist destinations, and the aforementioned requirements for a new traveler. We propose a Travel Recommendation System (TRS) that solves the above problems by examining a plethora of blog posts by various travel bloggers, use technologies such as map-reduce and natural language processing to process the vast amount of information, and deliver results that caters to all the requirements of a traveler.

Keywords: Travel Recommendation System (TRS), map-reduce, natural language processing.

Contents

List of Figures

1 Introduction

The need for a recommender system arose due to the fact that we often rely on other peoples' experience and recommendations when confronted with a new field of expertise, in which we do not have a broad knowledge of all the facts. Recommendations are a common means of planning in the fields of tourism and traveling, and thus, are an intuitive and valuable extension to tourism information systems.

The tourists of today are very demanding and have complex, multi-layered desires and needs. Users choose their destinations among various channels and compare tourism offerings critically. They are no longer satisfied with standardized tourism “products” or “packages”.
The “postmodern tourist” with differentiated life-styles (e.g. shorter trips), individual motives (e.g. business travelers, elderly persons, culture tourists, day-tourists) and specific interests (e.g. focus on special sports) demands products tailored accordingly to stated preferences.

Before embarking on a journey, the primary interest for the traveler is to get information on the main destination, required budget, and the duration of travel. While in the booking stage, they are seeking to get recommendations on the accommodation and the transportation. Finally, during the travel, they require current information about the events, restaurants, activities and shopping centers. The usual route taken by the traveler is to obtain travel information from different sources such as travel websites, friends, and experienced travelers, which can be a cumbersome task. Additionally, travelers’ preferences, and information related to destination, accommodation, flight, and so forth change rapidly[1].

We propose to tackle some of these issues by our Travel Recommendation System(TRS), named TRALOC, by extracting up-to-date information from various travel blogs, and presenting them to the traveler using an easy-to-navigate user interface. Travel blogs are an untapped source of information, as their authors visits a country or a city other than his/her hometown for a period of less than 12 months, face different decision-making challenges during their travel, and are frequently overwhelmed by a plethora of questions and travel information. The methods of preparation, and how they eventually overcame their difficulties during travel are key points which will be presented to the user. A content-based filtering approach[2] is implemented to obtain only the necessary information, so as not to burden the user. Multiple travel blogs have been surveyed to cover all aspects of the travel, and to provide the traveler with different viewpoints about the same destination (as he/she might not resonate with a single travel blog record). The proposed system attempts to emulate offline travel agents by providing users with knowledgeable travel suggestions to facilitate their decision-making processes.

The report is organized as follows: Chapter 2 talks about the requirements for running the product seamlessly, chapter 3 talks about the data collection process, chapter 4 explains the architecture of the product, chapter 5 explains the API architecture that we have implemented, chapter 6 explains the user interface in detail, followed by conclusions and future enhancements.

2 Requirements

2.1 Apache Cassandra (v3.1 or higher)

Apache Cassandra is a free and open-source distributed NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous masterless replication allowing low latency operations for all clients.

The main features of Cassandra are as follows:

Decentralized: Every node in the cluster has the same role. There is no single point of failure. Data is distributed across the cluster (so each node contains different data), but there is no master as every node can service any request.

Supports replication and multi data center replication: Replication strategies are configurable. Cassandra is designed as a distributed system, for deployment of large numbers of nodes across multiple data centers. Key features of Cassandra's distributed architecture are specifically tailored for multiple-data center deployment, for redundancy, for failover and disaster recovery.

Scalability: Designed to have read and write throughput both increase linearly as new machines are added, with the aim of no downtime or interruption to applications.

Fault-tolerant: Data is automatically replicated to multiple nodes for fault-tolerance. Replication across multiple data centers is supported. Failed nodes can be replaced with no downtime.

Tunable consistency: Writes and reads offer a tunable level of consistency, all the way from "writes never fail" to "block for all replicas to be readable", with the quorum level in the middle.

MapReduce support: Cassandra has Hadoop integration, with MapReduce support. There is support also for Apache Pig and Apache Hive.

Query language: Cassandra introduced the Cassandra Query Language (CQL). CQL is a simple interface for accessing Cassandra, as an alternative to the traditional Structured Query Language (SQL). CQL adds an abstraction layer that hides implementation details of this structure and provides native syntaxes for collections and other common encodings[13].

2.2 Java Runtime Environment (JRE – v1.8)

The Java Runtime Environment (JRE) released by Oracle is a freely available software distribution containing a stand-alone JVM (HotSpot), the Java standard library (Java Class Library), a configuration tool, and—until its discontinuation in JDK 9—a browser plug-in. It is the most common Java environment installed on personal computers in the laptop and desktop form factor. Mobile phones including feature phones and early smartphones that ship with a JVM are most likely to include a JVM meant to run applications targeting Micro Edition of the Java platform. The JVM specification gives a lot of leeway to implementors regarding the implementation details. Since Java 1.3, JRE from Oracle contains a JVM called HotSpot. It has been designed to be a high-performance JVM. To speed-up code execution, HotSpot relies on just-in-time compilation. To speed-up object allocation and garbage collection, HotSpot uses generational heap.

A Java virtual machine (JVM) is a virtual machine that enables a computer to run Java programs as well as programs written in other languages and compiled to Java bytecode. The JVM is detailed by a specification that formally describes what is required of a JVM implementation. Having a specification ensures interoperability of Java programs across different implementations so that program authors using the Java Development Kit (JDK) need not worry about idiosyncrasies of the underlying hardware platform. The JVM reference implementation is developed by the OpenJDK project as open source code and includes a JIT compiler called HotSpot. The commercially supported Java releases available from Oracle Corporation are based on the OpenJDK runtime[13].

2.3 Apache Maven (v3.5.3)

Maven is a build automation tool used primarily for Java projects.

Maven addresses two aspects of building software: first, it describes how software is built, and second, it describes its dependencies. It uses conventions for the build procedure, and only exceptions need to be written down. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins. It comes with pre-defined targets for performing certain well-defined tasks such as compilation of code and its packaging.

Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. This local cache of downloaded artifacts can also be updated with artifacts created by local projects. Public repositories can also be updated. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages.

Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input. Theoretically, this would allow anyone to write plugins to interface with build tools (compilers, unit test tools, etc.) for any other language. In reality, support and use for languages other than Java has been minimal. Currently a plugin for the .NET framework exists and is maintained, and a C/C++ native plugin is maintained for Maven 2[13].

2.4 Spark Java API

The Spark Java API exposes all the Spark features available in the Scala version to Java.

The Spark Java API is defined in the org.apache.spark.api.java package, and includes a JavaSparkContext for initializing Spark and JavaRDD classes, which support the same methods as their Scala counterparts but take Java functions and return Java data and collection types. The main differences have to do with passing functions to RDD operations (e.g. map) and handling RDDs of different types.

A few key differences between the Java and Scala APIs are as follows:

- Java does not support anonymous or first-class functions, so functions must be implemented by extending the org.apache.spark.api.java.function, Function, Function2, etc. classes.
- To maintain type safety, the Java API defines specialized Function and RDD classes for key-value pairs and doubles. For example, JavaPairRDD stores key-value pairs.
- RDD methods like collect() and countByKey() return Java collections types, such as java.util.List and java.util.Map.
- Key-value pairs, which are simply written as (key, value) in Scala, are represented by the scala.Tuple2 class, and need to be created using: new Tuple2<K, V>(key, value) [13].

2.5 Gson

Gson (also known as Google Gson) is an open source Java library to serialize and deserialize Java objects to (and from) JSON. Features of Gson include:

- Gson can handle collections, generic types and nested classes (including inner classes, this cannot be done by default though)
- When deserializing, Gson is navigating the type tree of the object being deserialized. This results in ignoring extra fields present in the JSON input.
- User can write a custom serializer and/or deserializer so that they can control the whole process and even (de)serialize instances of classes for which the source code is not accessible.
- User can write an InstanceCreator which allows them to deserialize instances of classes without a defined no-args constructor.
- Gson is highly customizable, you can specify:
    - Compact/pretty printing (whether you want compact or readable output).
    - How to handle null object fields - by default they are not present in the output.
    - Rules of what fields are intended to be excluded from (de)serialization.
    - How to convert Java field names[13].

2.6 Apache Spark

Apache Spark is an open-source cluster-computing framework. Spark provides an interface for programming entire clusters with implicit data parallelism and fault tolerance.

Apache Spark has as its architectural foundation the resilient distributed dataset (RDD), a read-only multiset of data items distributed over a cluster of machines, that is maintained in a fault-tolerant way. Spark and its RDDs were developed in 2012 in response to limitations in the MapReduce cluster computing paradigm, which forces a particular linear dataflow structure on distributed programs: MapReduce programs read input data from disk, map a function across the data, reduce the results of the map, and store reduction results on disk. Spark's RDDs function as a working set for distributed programs that offers a (deliberately) restricted form of distributed shared memory.

Apache Spark requires a cluster manager and a distributed storage system. For cluster management, Spark supports standalone (native Spark cluster), Hadoop YARN, or Apache Mesos. For distributed storage, Spark can interface with a wide variety, including Hadoop Distributed File System (HDFS), MapR File System (MapR-FS), Cassandra, OpenStack Swift, Amazon S3, Kudu, or a custom solution can be implemented. Spark also supports a pseudo-distributed local mode, usually used only for development or testing purposes, where distributed storage is not required and the local file system can be used instead; in such a scenario, Spark is run on a single machine with one executor per CPU core.

Spark Core is the foundation of the overall project. It provides distributed task dispatching, scheduling, and basic I/O functionalities, exposed through an application programming interface (for Java, Python, Scala, and R) centered on the RDD abstraction (the Java API is available for other JVM languages, but is also usable for some other non-JVM languages, such as Julia, that can connect to the JVM). This interface mirrors a functional/higher-order model of programming: a "driver" program invokes parallel operations such as map, filter or reduce on an RDD by passing a function to Spark, which then schedules the function's execution in parallel on the cluster. These operations, and additional ones such as joins, take RDDs as input and produce new RDDs. RDDs are immutable and their operations are lazy; fault-tolerance is achieved by keeping track of the "lineage" of each RDD (the sequence of operations that produced it) so that it can be reconstructed in the case of data loss. RDDs can contain any type of Python, Java, or Scala objects[13].

2.7 Spark SQL

Spark SQL is a component on top of Spark Core that introduced a data abstraction called DataFrames, which provides support for structured and semi-structured data. Spark SQL provides a domain-specific language (DSL) to manipulate DataFrames in Scala, Java, or Python. It also provides SQL language support, with command-line interfaces and ODBC/JDBC server. Although DataFrames lack the compile-time type-checking afforded by RDDs, as of Spark 2.0, the strongly typed DataSet is fully supported by Spark SQL as well[13].

2.8 Spark-Cassandra Connector

The Spark Cassandra Connector Java API allows you to create Java applications that use Spark to analyze database data. This library lets you expose Cassandra tables as Spark RDDs, write Spark RDDs to Cassandra tables, and execute arbitrary CQL queries in your Spark applications[13].

2.9 Opencsv

Opencsv is an easy-to-use CSV (comma-separated values) parser library for Java.

2.10 Stanford CoreNLP

Stanford CoreNLP provides a set of natural language analysis tools written in Java. It can take raw human language text input and give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize and interpret dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases or word dependencies, and indicate which noun phrases refer to the same entities. It was originally developed for English, but now also provides varying levels of support for (Modern Standard) Arabic, (mainland) Chinese, French, German, and Spanish. Stanford CoreNLP is an integrated framework, which make it very easy to apply a bunch of language analysis tools to a piece of text. Starting from plain text, you can run all the tools with just two lines of code. Its analyses provide the foundational building blocks for higher-level and domain-specific text understanding applications. Stanford CoreNLP is a set of stable and well-tested natural language processing tools, widely used by various groups in academia, industry, and government. The tools variously use rule-based, probabilistic machine learning, and deep learning components.

The Stanford CoreNLP code is written in Java and licensed under the GNU General Public License (v3 or later).

2.11 Google Guava

Google Guava is an open-source set of common libraries for Java, mainly developed by Google engineers. Google Guava can be roughly divided into three components: basic utilities to reduce menial labors to implement common methods and behaviors, an extension to the Java collections framework (JCF) formerly called the Google Collections Library, and other utilities which provide convenient and productive features such as functional programming, graphs, caching, range objects, and hashing[13].

3 Data Collection

Data collection is an important aspect of any big-data application. The more data the application has access to, the more accurate and refined the results will be. The application will find it easier to establish a pattern between the data parameters, if there is a large amount of it. The data, how it was collected, and its sources are listed below:

Blog data: This part was tricky, as different blogs are written differently. The need of a common script that could extract various blog data was necessary. Fortunately, such a script was written (in Python 3.5), with the help of a library called BeautifulSoup. Using this, data from nearly 576 blog posts were scrapped. The .html files were then converted to plain text using an online conversion tool, and then had to be manually trimmed down, in order to remove redundant data, unrecognized symbols, private addresses and phone numbers of blog authors, and so on. The url of the blog and its corresponding data were added to a comprehensive 2-column csv file, and finally fed to the application for processing. This entire process from web scrapping, to the final data took around 12 – 14 days.

Country-specific data: Country-specific data was obtained from a generous user's github repository[3], which contained world data in separate json files, each file containing a country column, and a specific attribute. Necessary attributes, were chosen, and the corresponding json files were combined to produce a single comprehensive world dataset. This contains the attributes: Country, Abbreviation, Capital, Latitude, Longitude, Continent, Region, Language(s), Area, Elevation, Total population, Population density, Currency, Currency code, National symbol, National dish, and Temperature.

Restaurant data: Restaurant data was obtained from the kaggle competition "Zomato Restaurants Data"[4], which in-turn was collected from the Zomato API in the form of .json files (raw data). The csv version from the kaggle competition was fed to the application.

4 Architecture

The backend consists of a java project, which uses maven as a build tool to include various dependencies. The data is stored in Apache Cassandra and the processing of text is done using Stanford NLP. The architecture is shown is **Fig 4.1**. The flow of the project is as follows:

1) **Data Layer**

   The data layer consists of reading the CSV files[8] for blog, country and restaurant information and storing it in the keyspace 'travel' in table raw_data, country and restaurant_data respectively. The following commands[5] were used to create the keyspace and the tables:

```
create keyspace travel with replication = {'class':'SimpleStrategy', 'replication_factor':3};

create table travel.raw_data (id UUID PRIMARY KEY, url text, content text, country text);

create table travel.country(Country text primary key,Abbreviation text,Capital text,North text,South text, East text, West text, Continent text, Region text, Languages text,Area text, Elevation text, Population text, Population_density text, Currency_name text, Currency_code text, National_symbol text, Symbol_name text, National_dish text, Average_temperature text, Measurement text);

create table travel.restaurant_data(id UUID primary key, country text, name text, cuisine text, cost_for_two double, rating double);

create table travel.transformed_data(country text primary key, positive list<text>);
```

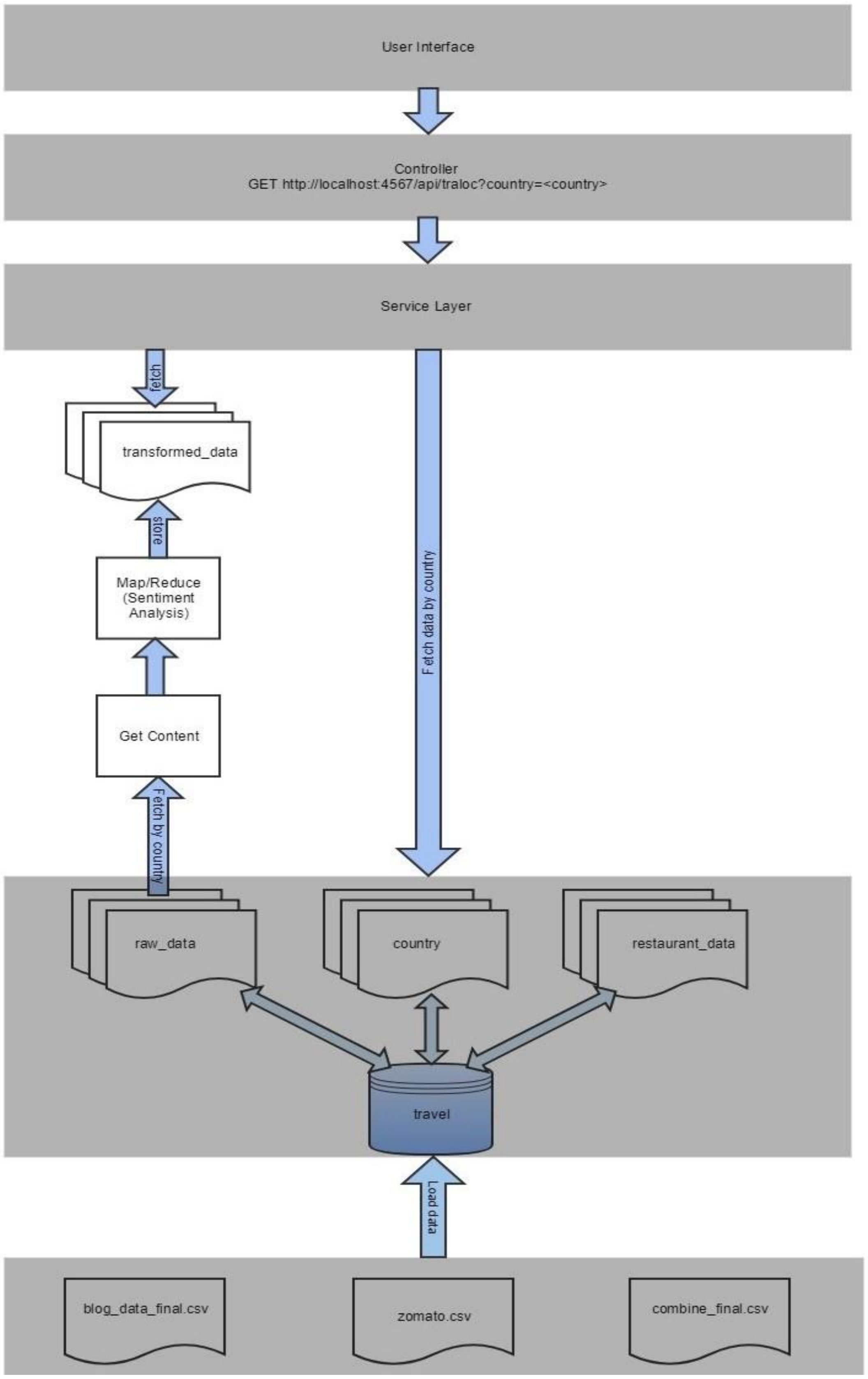


Fig 4.1: Architecture Diagram[6]

Sample data are as follows:

- raw_data

  id| content| country | url

  f1e17d60-c3cb-45fb-b50c-2726d7527c96 | What makes Phnom Penh a perfect destination for travelers seeking a glamorous trip? Phnom Penh may not be known as a glamorous destination, but that is quickly changing. An influx of foreign investment and expats has meant a lot of new businesses geared toward foreigners. ... They have events seven night a week, including ladies night, international DJs and even drag shows! We big on the club scene, but we do frequent some great bars around town. One of our favorites is Che Culo , Italian for ΓÇÿlucky bastardΓÇÖ, for the ambiance and great selection of small bites and tasty drinks. | cambodia | https://hippie-inheels.com/luxury-guide-to-phnom-penh/

- country

  country | abbreviation | area | average_temperature | capital | continent | currency_code | currency_name | east | elevation | languages | measurement | national_dish | national_symbol | north | population | population_density | region | south | symbol_name | west

  United States | US | 9363520 | 8.55 | Washington | North America | USD | US Dollar | -66.954811 | 760m | Chinese,English,French,German,Italian,Japanese,Korean,Polish,Portuguese,Spanish,Tagalog,Vietnamese | celsius | Hamburger, Hot dog, Fried chicken, Buffalo wings, Apple pie | | 49.388611 | 278357000 | 34.56 | North America | 24.544245 | | -124.733253

- restaurant_data

  id | cost_for_two | country | cuisine | name | rating

  3f34c461-85ca-4c55-895c-6d21e9dc727e | 350 | India | Bakery, Chinese, Fast Food | Bakery Wala - The Cake Shop | 3.4

2) Processing:

The data is queried from the “raw_data” table and filtered by country. The ‘content’ field from the list of RawData objects is fetched and split into sentences. The sentences are then sent for sentiment analysis. The sentiment analysis process takes each sentence as an input and performs the following tasks:

- tokenize: tokenizes it into tokens
- ssplit: splits the tokens into sentences
- parse: provides full syntactic analysis and a phrase-structure tree of sentences
- sentiment: predicts the sentiment

The pipeline processes the sentence and fetches the predicted class (integer) using Recurrent Neural Networks, per phrase. For a sentence, the sentiments are summed up to get a final score for the sentence. This sum is returned as the main sentiment for the input string. The result is a sentence and its sentiment. The code snippet [7] for this task is as follows:

```
JavaPairRDD<String, Integer> sentiments = rdd
    .flatMap(s -> Arrays.asList(s.getContent().split("(?<!\\w\\.\\w.)(?<![A-Z][a-z]\\.)(?<=\\.|\\?)\\s")).iterator())
```

```
.mapToPair(sentence -> new Tuple2<>(sentence, getSentiment(sentence)))
.reduceByKey((a, b) -> a + b);
```

This returns a map which is then sorted based on its value and the top 5 scoring sentences (positive sentiments) are then stored along with the country name in “transformed_data” table.

- transformed_data
  country | positive
  poland |['Now that the sunΓÇÖs gone down, fill us on the best places to go for some evening drinks or a great pre-gaming spot.', 'Krakow has more bars per square metre than anywhere else in the world.', 'There isnΓÇÖt any other place in Krakow with this beautiful sunset.', 'There are many 5-star hotels in Krakow, but we would choose Hotel Stary .', 'Tourist from all over to world visit Krakow to see beautiful Rynek (Old Town) and go for one-day trip to neighbouring Wieliczka, famous salt-mine.']

3) **Service Layer:**

The service layer fetches data from the tables based on a country. It then combines the data to include country information, blog post urls, top restaurants in that country and top 5 sentences from all blogs based on that country.

4) **Controller:**

The controller is a REST API (Application Programming Interface) of type GET that gets called when the user selects a country of their choice.

5) **User Interface:**

The UI is built using HTML, CSS and Javascript. The tab in the centre of the page has a search option to pick the country name you want to search as the destination from a dropdown. The second section of the first page displays a world map as an SVG and the last section displays a Contact form. The second page displays the restaurants, cuisines, cost for two, ratings, geographical details of the country, the highlights of the country and the links to the blogs. It also has a menu tab for ease of navigation.

5 API Architecture

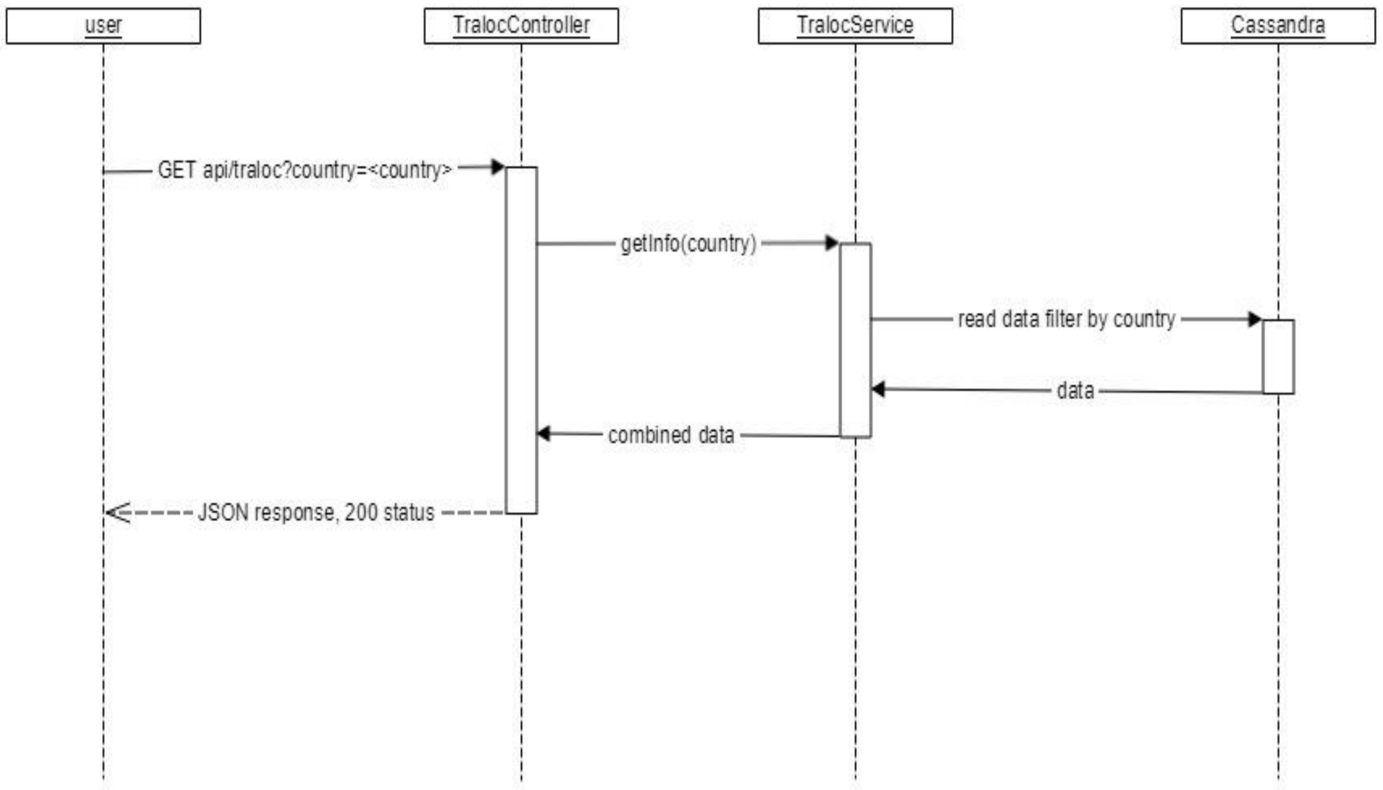

user
TralocController
TralocService
Cassandra
GET api/traloc?country=<country>
getInfo(country)
read data filter by country
data
combined data
JSON response, 200 status

Fig 5.1 Sequence Diagram

The URL for the homepage is http://localhost:4567/travel.html which gives a search box. When clicked on, it shows a dropdown with the available countries. Once a country is chosen, a GET API gets triggered.

The API[9] URL is http://localhost:4567/api/traloc?country=<country> that accepts one query parameter, country and calls a service. This TralocService combines the data from the tables country, transformed_data, restaurant_data and raw_data into one object and returns it. The controller then returns this JSON object with status code 200.

Following is a sample JSON response from the API:

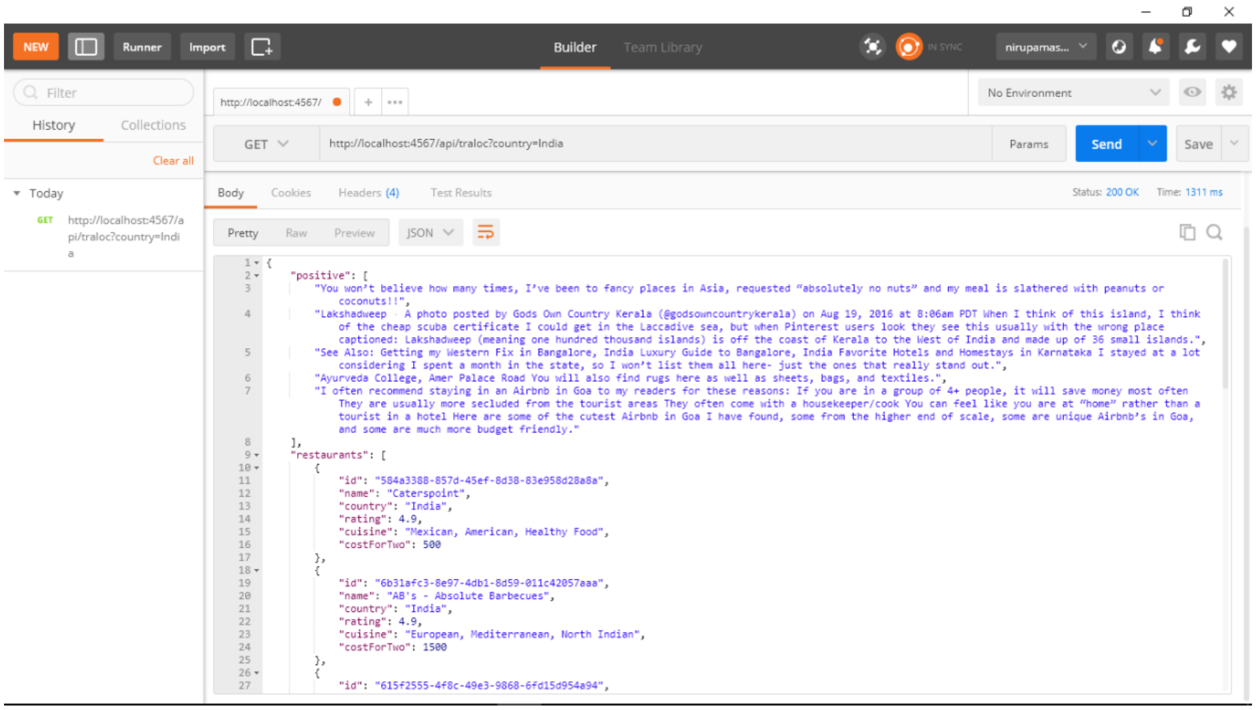
```
{
    "positive": [
        "You won’t believe how many times, I’ve been to fancy places in Asia, requested “absolutely no nuts” and my meal is slathered with peanuts or coconuts!!",
        "Lakshadweep    A photo posted by Gods Own Country Kerala (@godsowncountrykerala) on Aug 19, 2016 at 8:06am PDT When I think of this island, I think of the cheap scuba certificate I could get in the Laccadive sea, but when Pinterest users look they see this usually with the wrong place captioned: Lakshadweep (meaning one hundred thousand islands) is off the coast of Kerala to the West of India and made up of 36 small islands.",
        "See Also: Getting my Western Fix in Bangalore, India Luxury Guide to Bangalore, India Favorite Hotels and Homestays in Karnataka I stayed at a lot considering I spent a month in the state, so I won’t list them all here- just the ones that really stand out.",
        "Ayurveda College, Amer Palace Road You will also find rugs here as well as sheets, bags, and textiles.",
        "I often recommend staying in an Airbnb in Goa to my readers for these reasons: If you are in a group of 4+ people, it will save money most often They are usually more secluded from the tourist areas They often come with a housekeeper/cook You can feel like you are at “home” rather than a tourist in a hotel Here are some of the cutest Airbnb in Goa I have found, some from the higher end of scale, some are unique Airbnb’s in Goa, and some are much more budget friendly."
    ],
    "restaurants": [
        {
            "id": "584a3388-857d-45ef-8d38-83e958d28a8a",
            "name": "Caterspoint",
            "country": "India",
        "rating": 4.9,
        "cuisine": "Mexican, American, Healthy Food",
        "costForTwo": 500
    },
    {
        "id": "6b31afc3-8e97-4db1-8d59-011c42057aaa",
        "name": "AB's - Absolute Barbecues",
        "country": "India",
        "rating": 4.9,
        "cuisine": "European, Mediterranean, North Indian",
        "costForTwo": 1500
    },
    {
        "id": "615f2555-4f8c-49e3-9868-6fd15d954a94",
        "name": "Barbeque Nation",
        "country": "India",
        "rating": 4.9,
        "cuisine": "North Indian, Chinese",
        "costForTwo": 1600
    },
    {
        "id": "62d814d0-5bae-43b1-a30a-6ac5283d7c5f",
        "name": "Zolocrust - Hotel Clarks Amer",
        "country": "India",
        "rating": 4.9,
        "cuisine": "Italian, Bakery, Continental",
        "costForTwo": 2000
    },
    {
```

```
                "id": "c08738e0-d5b1-4cf9-98e9-8926059258ca",
                "name": "Barbeque Nation",
                "country": "India",
                "rating": 4.9,
                "cuisine": "North Indian, Chinese, Mediterranean",
                "costForTwo": 1600
            }
        ],
        "urls": [
            "https://hippie-inheels.com/things-to-do-in-udaipur/",
            "https://hippie-inheels.com/luxury-guide-to-jaipur/",
            "https://hippie-inheels.com/sangti-valley/",
            "https://hippie-inheels.com/european-cities-india/",
            "https://hippie-inheels.com/guide-to-mysore/",
            "https://hippie-inheels.com/bangalore-local-food-guide/",
            "https://hippie-inheels.com/little-weekend-guide-bangalore/",
            "https://www.thepoortraveler.net/2017/10/india-golden-triangle/",
            "https://www.thepoortraveler.net/2015/04/budget-save-india-tour/",
            "https://hippie-inheels.com/best-indian-dishes-for-travelers/",
            "https://hippie-inheels.com/itanagar-market/",
            "https://hippie-inheels.com/complete-guide-traveling-karnataka-9-hotspots/",
            "https://hippie-inheels.com/luxurious-indian-islands/",
            "https://hippie-inheels.com/best-indian-honeymoon-destinations/",
            "https://hippie-inheels.com/best-places-to-visit-in-india-from-five-years-of-
exploration/",
            "https://hippie-inheels.com/luxury-guide-mumbai/",
            "https://hippie-inheels.com/goa-nightlife-most-popular-party-places/",
            "https://hippie-inheels.com/little-souvenir-guide-buy-india-region/",
            "https://hippie-inheels.com/find-a-house-in-goa/",
```

"https://hippie-inheels.com/places-to-visit-in-jaipur/",
"https://hippie-inheels.com/10-days-kerala-backpacking-tips-itinerary/",
"https://hippie-inheels.com/dont-miss-places-visit-bomdila/",
"https://www.thepoortraveler.net/2016/05/hilarious-backpacking-experiences-round-the-world/",
"https://hippie-inheels.com/places-visit-goa-tourists/",
"https://hippie-inheels.com/one-month-backpacking-india/",
"https://hippie-inheels.com/things-to-do-in-coorg/",
"https://hippie-inheels.com/my-little-guide-to-jodhpur-indias-blue-city/"
],
"country": {
"country": "India",
"abbreviation": "IN",
"capital": "New Delhi",
"north": "35.504223",
"south": "6.747139",
"east": "97.403305",
"west": "68.186691",
"continent": "Asia",
"region": "Southern and Central Asia",
"languages": "Asami, Bengali,Gujarati,Hindi,Kannada,Malajalam,Marathi,Orija,Punjabi,Tamil,Telugu,Urdu",
"area": "3287263",
"elevation": "160m",
"population": "1013662000",
"populationDensity": "421.14",
"currencyName": "Indian Rupee",
"currencyCode": "INR",

```
            "nationalSymbol": "animal",
            "symbolName": "Lioned Capital",
            "nationalDish": "No officially anointed national dish",
            "averageTemperature": "23.65",
            "measurement": "celsius"
      }
}
```

Testing for the API was done using Postman. Here is a screenshot of the testing:



GET http://localhost:4567/api/traloc?country=India

Status: 200 OK Time: 1311 ms

```
{
    "positive": [
        "You won’t believe how many times, I’ve been to fancy places in Asia, requested “absolutely no nuts” and my meal is slathered with peanuts or
            coconuts!!",
        "Lakshadweep · A photo posted by Gods Own Country Kerala (@godsowncountrykerala) on Aug 19, 2016 at 8:06am PDT When I think of this island, I think
            of the cheap scuba certificate I could get in the Laccadive sea, but when Pinterest users look they see this usually with the wrong place
            captioned: Lakshadweep (meaning one hundred thousand islands) is off the coast of Kerala to the West of India and made up of 36 small islands.",
        "See Also: Getting my Western Fix in Bangalore, India Luxury Guide to Bangalore, India Favorite Hotels and Homestays in Karnataka I stayed at a lot
            considering I spent a month in the state, so I won’t list them all here- just the ones that really stand out.",
        "Ayurveda College, Amer Palace Road You will also find rugs here as well as sheets, bags, and textiles.",
        "I often recommend staying in an Airbnb in Goa to my readers for these reasons: If you are in a group of 4+ people, it will save money most often
            They are usually more secluded from the tourist areas They often come with a housekeeper/cook You can feel like you are at “home” rather than a
            tourist in a hotel Here are some of the cutest Airbnb in Goa I have found, some from the higher end of scale, some are unique Airbnb’s in Goa,
            and some are much more budget friendly."
    ],
    "restaurants": [
        {
            "id": "584a3388-857d-45ef-8d38-83e958d28a8a",
            "name": "Caterspoint",
            "country": "India",
            "rating": 4.9,
            "cuisine": "Mexican, American, Healthy Food",
            "costForTwo": 500
        },
        {
            "id": "6b31afc3-8e97-4db1-8d59-011c42057aaa",
            "name": "AB's - Absolute Barbecues",
            "country": "India",
            "rating": 4.9,
            "cuisine": "European, Mediterranean, North Indian",
            "costForTwo": 1500
        },
        {
            "id": "615f2555-4f8c-49e3-9868-6fd15d954a94",
```

Fig 5.2: API Testing

6 User Interface

The HTML, JavaScript and CSS files[11] are present in the "main/resources/static/countries_html" folder. This folder contains HTML pages for each of the countries since the SVG maps[10] available needed to be added per country. The JavaScript[12] files are present in the 'js' folder and CSS in 'css' folder. The 'Countries' folder consists of all the SVGs to display the map. Following are the screenshots of the UI at each step of the process:

1) **Homepage at http://localhost:4567/travel.html**

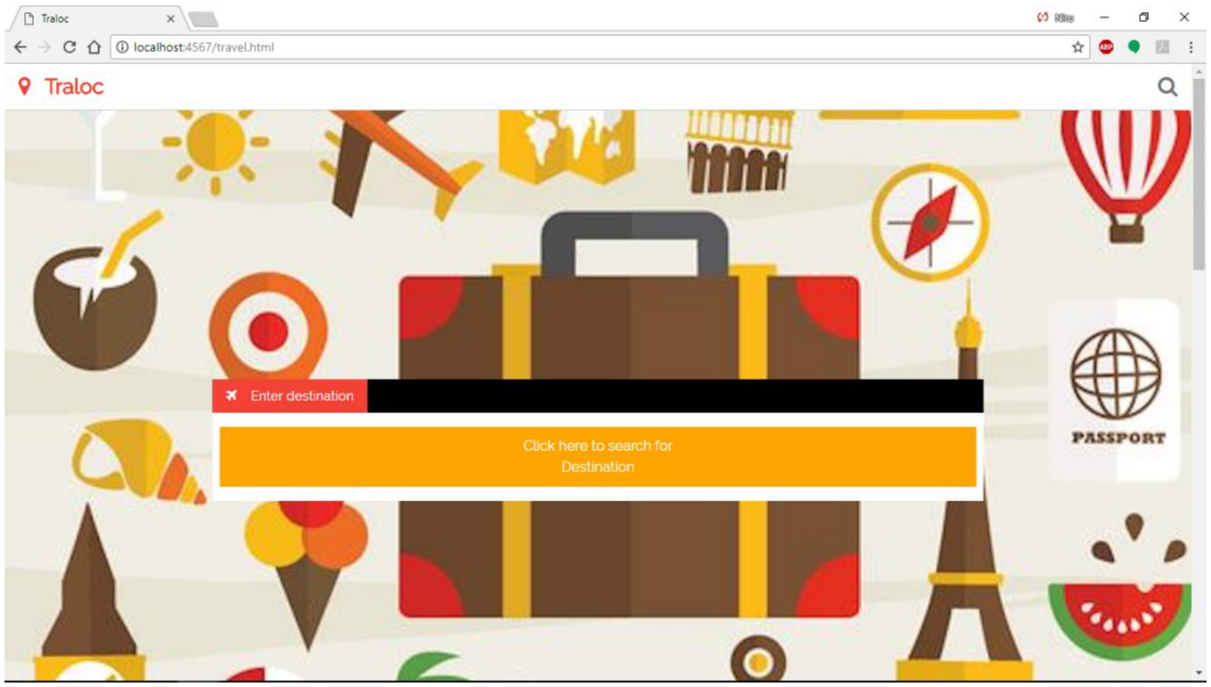
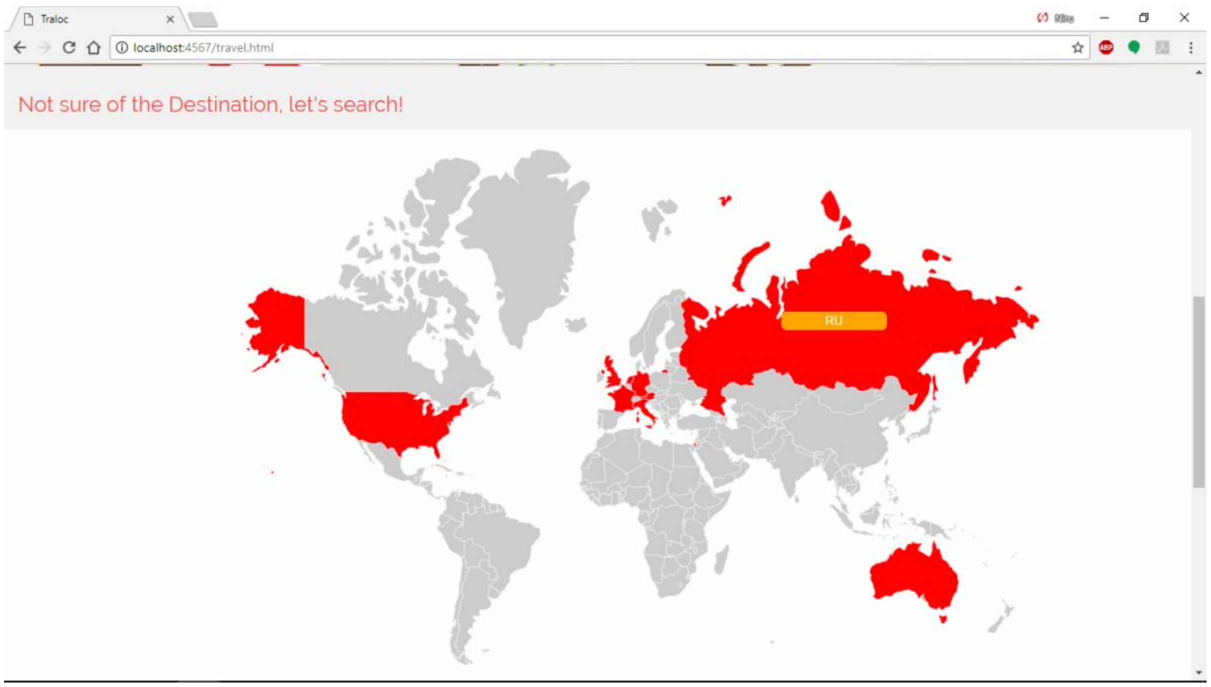

Traloc
localhost:4567/travel.html
Traloc
Enter destination
Click here to search for Destination
PASSPORT

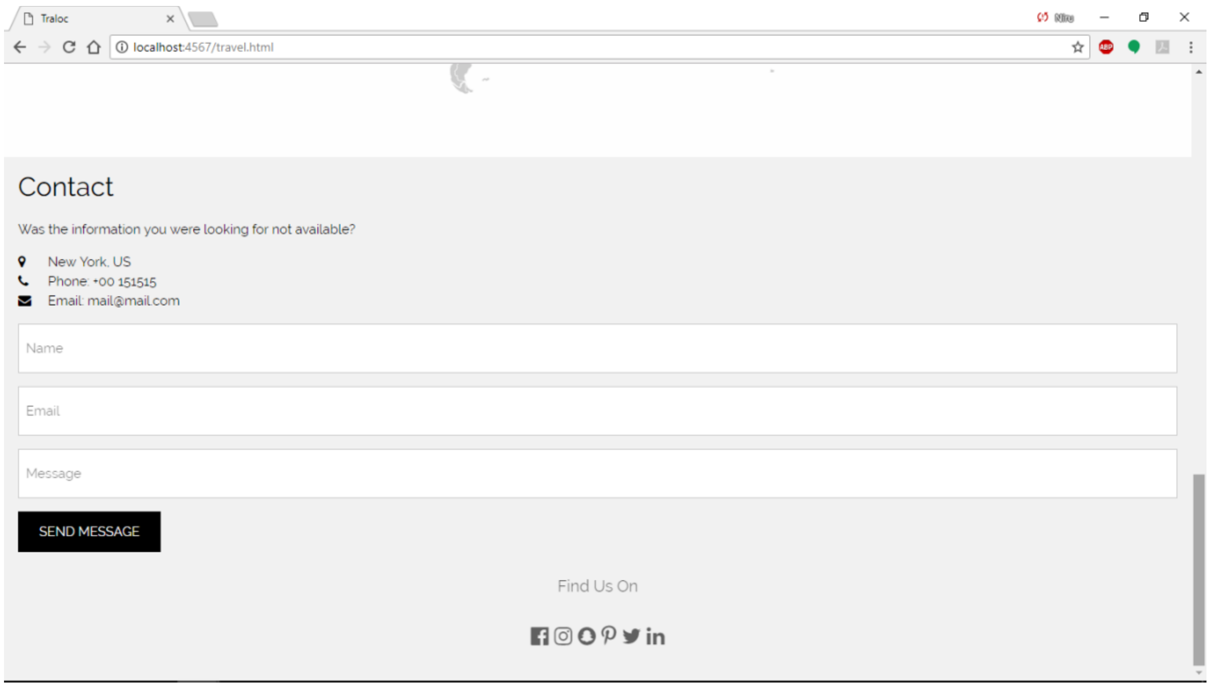
Fig 6.1: Homepage Part 1

The initial screen presented to the user is shown in **Fig 6.1**. Here, the user gets to select a country of their choice, either by typing the name of the country, or selecting it from the drop-down menu. The user is also presented with an interactive world map, using which they can select a destination as well. This is shown is **Fig 6.2**. At the end of the home page, contact information is available to the user, using which they can tell us about a missing country, or just to provide feedback. This is shown is **Fig 6.3**.



Not sure of the Destination, let's search!

Fig 6.2: Homepage Part 2



Contact

Was the information you were looking for not available?

New York, US
Phone: +00 151515
Email: mail@mail.com

Name

Email

Message

SEND MESSAGE

Find Us On

Fig 6.3: Homepage Part 3

2) **Choosing a destination**

As stated above, the user can either select their destination by typing it in the search box, or by selecting it from the drop-down menu. This is shown in **Fig 6.4** and **Fig 6.5.**

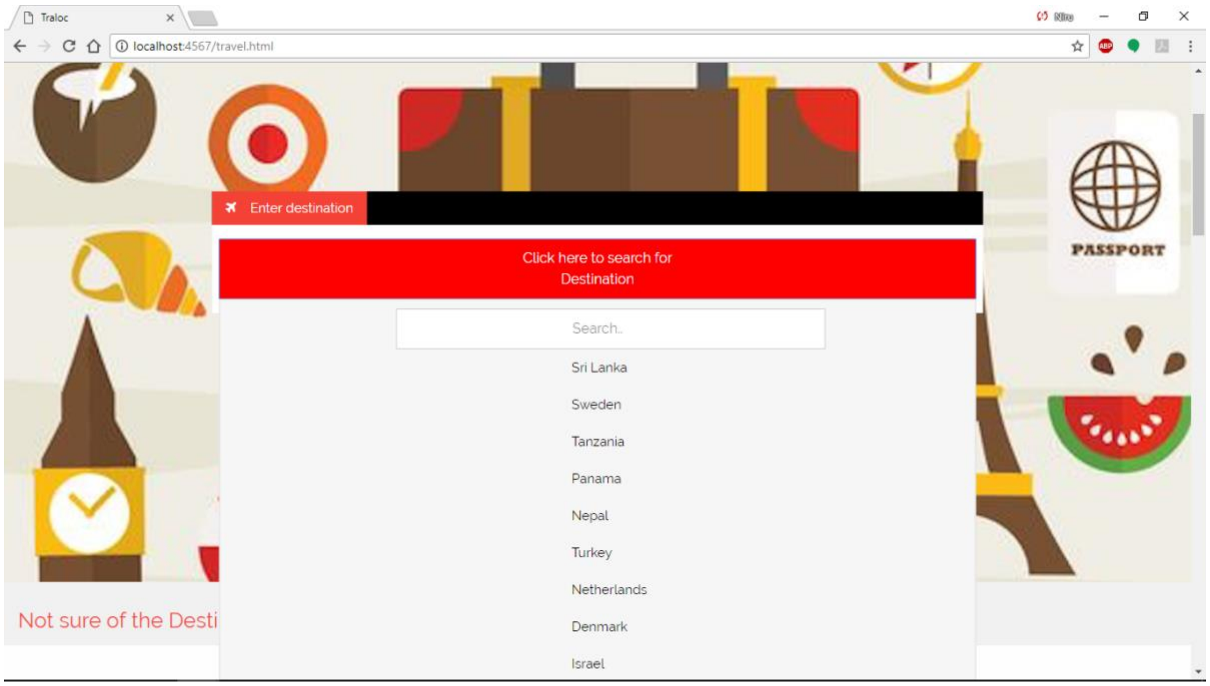


Fig 6.4: Dropdown Menu

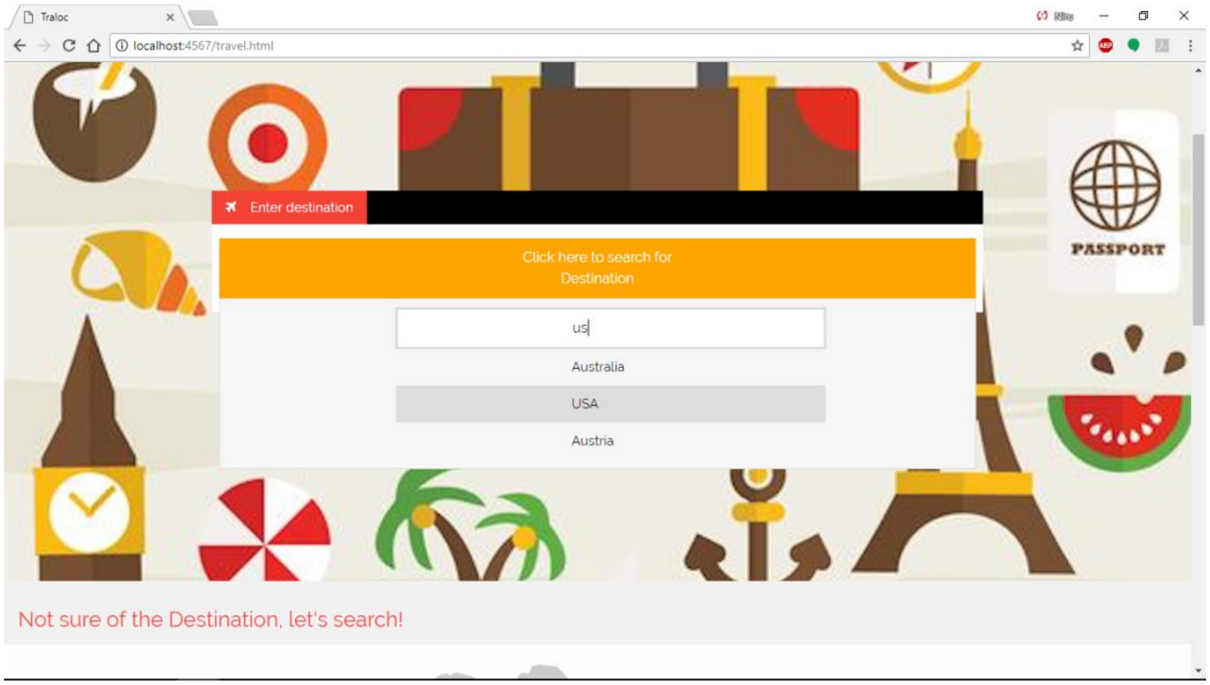
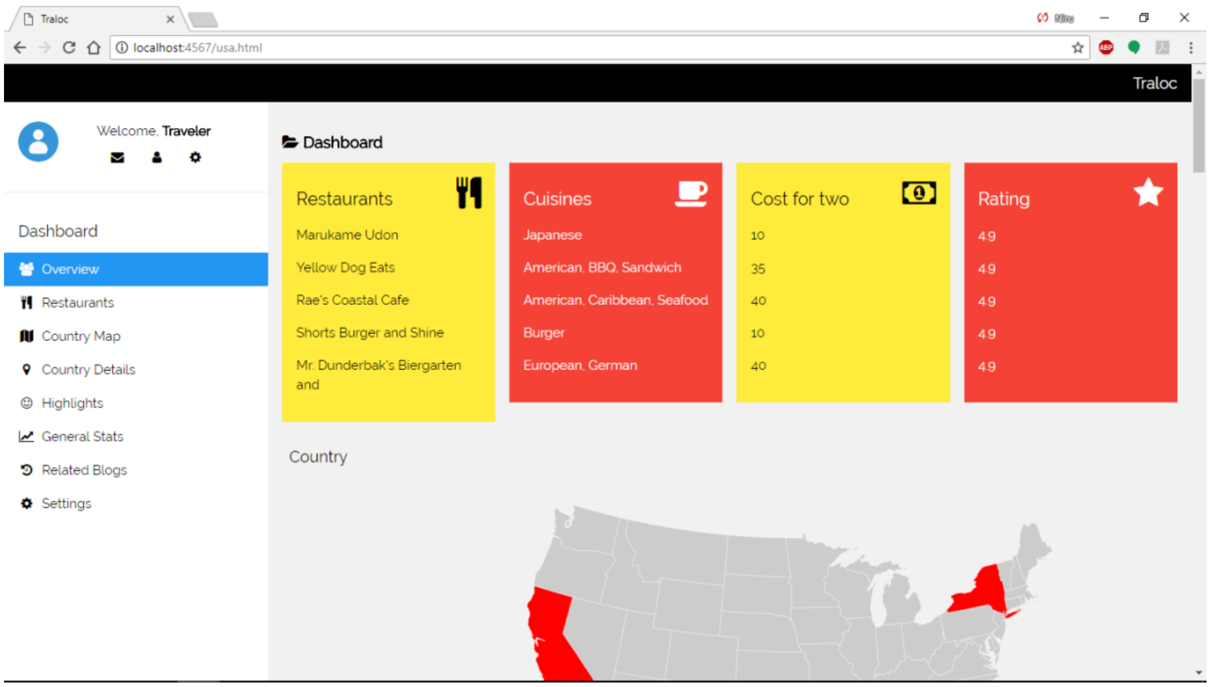

Traloc
localhost:4567/travel.html
PASSPORT
Enter destination
Click here to search for Destination
us
Australia
USA
Austria
Not sure of the Destination, let's search!

Fig 6.5: Selecting a Country

3) Once the user chooses the country, the user gets redirected to the page http://localhost:4567/<country_name>.html of that country. Here, the user is provided with all the information required to make an informed decision about the travel destination. The different types of information provided for the user are as follows:

- The Dashboard displays a top-5 list of local restaurants, cuisines, the cost for two, and their ratings (**Fig 6.6**).
- An interactive map of the selected country is also displayed (**Fig 6.7**).
- General information about the country such as its capital city, temperature, currency, is displayed, along with the highlights of visiting that country (**Fig 6.8**).
- A link to various blog posts is also provided for the user where they can obtain a more personalized account of going on vacation to the selected country (**Fig 6.9**)



Restaurants	Cuisines	Cost for two	Rating
Marukame Udon	Japanese	10	4.9
Yellow Dog Eats	American, BBQ, Sandwich	35	4.9
Rae's Coastal Cafe	American, Caribbean, Seafood	40	4.9
Shorts Burger and Shine	Burger	10	4.9
Mr. Dunderbak's Biergarten and	European, German	40	4.9

Fig 6.6: Results Part 1

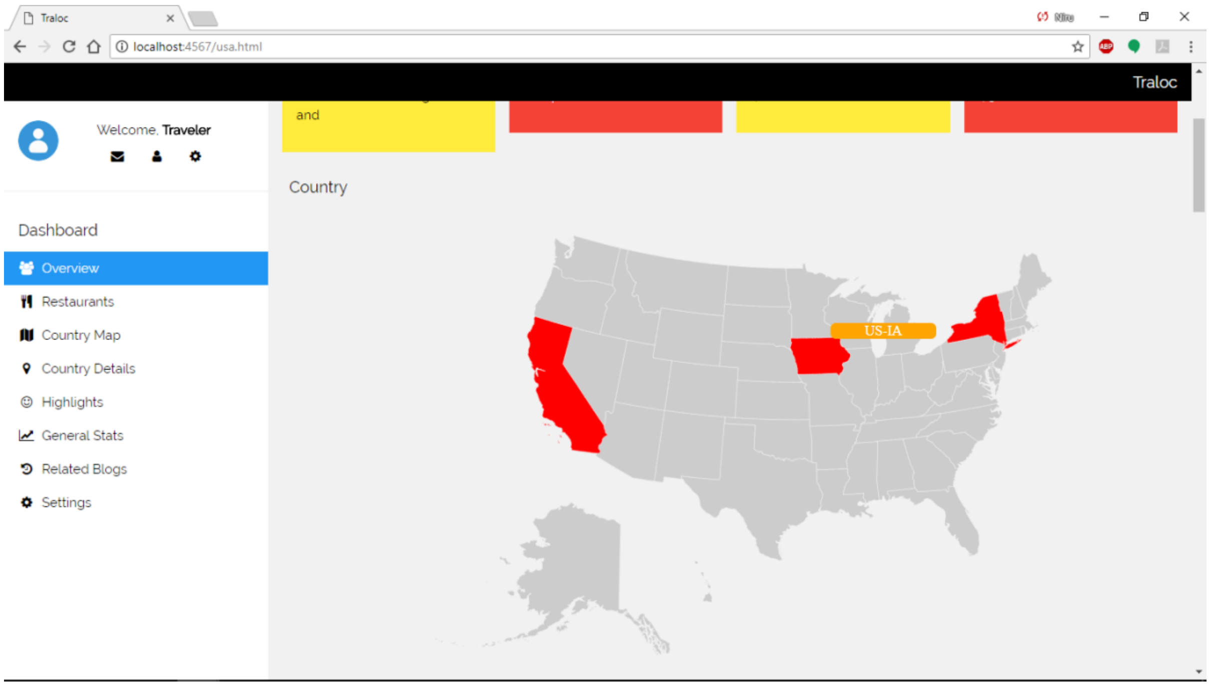
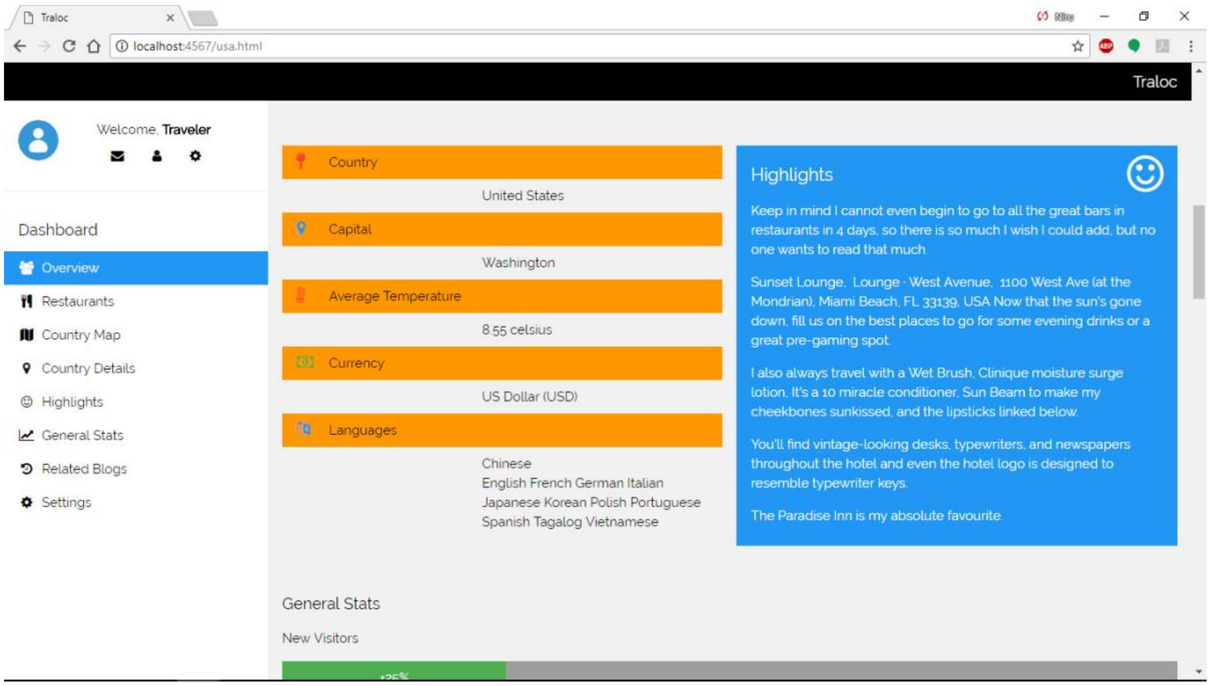


Fig 6.7: Results Part 2



Country
United States
Capital
Washington
Average Temperature
8.55 celsius
Currency
US Dollar (USD)
Languages
Chinese
English French German Italian
Japanese Korean Polish Portuguese
Spanish Tagalog Vietnamese

Highlights

Keep in mind I cannot even begin to go to all the great bars in restaurants in 4 days, so there is so much I wish I could add, but no one wants to read that much.

Sunset Lounge, Lounge · West Avenue, 1100 West Ave (at the Mondrian), Miami Beach, FL 33139, USA Now that the sun's gone down, fill us on the best places to go for some evening drinks or a great pre-gaming spot.

I also always travel with a Wet Brush, Clinique moisture surge lotion, It's a 10 miracle conditioner, Sun Beam to make my cheekbones sunkissed, and the lipsticks linked below.

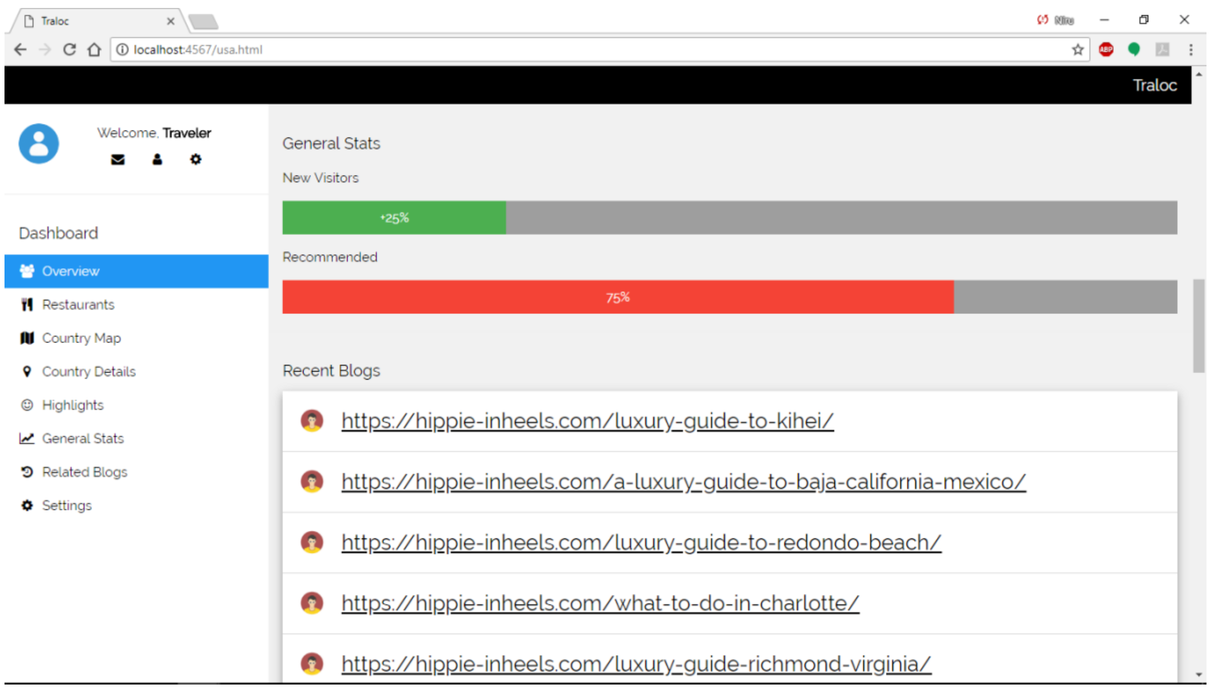
You'll find vintage-looking desks, typewriters, and newspapers throughout the hotel and even the hotel logo is designed to resemble typewriter keys.

The Paradise Inn is my absolute favourite.

General Stats

New Visitors

Fig 6.8: Results Part 3



General Stats

New Visitors

+25%

Recommended

75%

Recent Blogs

https://hippie-inheels.com/luxury-guide-to-kihei/

https://hippie-inheels.com/a-luxury-guide-to-baja-california-mexico/

https://hippie-inheels.com/luxury-guide-to-redondo-beach/

https://hippie-inheels.com/what-to-do-in-charlotte/

https://hippie-inheels.com/luxury-guide-richmond-virginia/

Fig 6.9: Results Part 4

7 Conclusion

The main aim of this project is to make use of big-data technologies learned during the coursework and create a product that provides a traveler with a recommendation system that will provide him the with information about the country they would like to visit.

We achieved this task by creating a product that is interactive as well as visually appealing to the user. In doing so, we successfully learned to use big data technologies from a product perspective. The process of coming up with an idea and finding the right technologies to use to achieve a scalable and available system was one of the challenges we faced.

Along the way, we learned how to use Cassandra as a database storage and connect to it using Spark, using numerous APIs, some familiar, some not. Other new technologies we had to implement that we were unfamiliar with were Gson, Stanford NLP, to name a few.

In conclusion, we hope to build a reliable and a seamless system that becomes a full-blown product in the future, and make it open-source to welcome any contributions from users and developers for its improvement.

8 Future Enhancements

- We hope to host the project on AWS for scalability and push this code into GitHub as a public repository to make it open-source and allow contributions.
- Another enhancement is to include a search feature using Elasticsearch and visualization using Kibana. The user interface still needs some work to make the interactions seamless and efficient.
- The data collection process involved a lot of manual intervention which we hope to eliminate in the future.

References

[1] https://link.springer.com/article/10.1007/s40595-014-0034-5#Sec5

[2] https://pdfs.semanticscholar.org/2457/5d6fca09fe66cee03b1d18c4ba05de62e03d.pdf

[3] https://github.com/samayo/country-json

[4] https://www.kaggle.com/shrutimehta/zomato-restaurants-data/data

[5] https://docs.datastax.com/en/cql/3.1/cql/cql_reference

[6] https://www.gliffy.com/

[7] https://www.datastax.com/dev/blog/accessing-cassandra-from-spark-in-java

[8] https://stackoverflow.com

[9] http://www.baeldung.com/spark-framework-rest-api

[10] https://www.amcharts.com/svg-maps/

[11] https://www.w3schools.com

[12] https://codepen.io/anon/pen/WdjJzz

[13] https://en.wikipedia.org/