World Bank Group 4

Vance Armstrong, Jack Ribarich, Jack Rocca

Cal Poly M.S. Business Analytics

Winter 2022

Introduction

The project explained in this paper focuses on two major points of interest for the World Bank International Comparison Program (ICP) team. The first point being the lack of data available to determine purchasing power parities (PPP) and the second point being how to use this data to determine the PPP in a more accurate and efficient manner. With respect to the lack of data, this project simplifies the process of finding data, especially pricing information, for housing by describing and implementing a web scraper for global real estate. Pertaining to the use of this data after it is collected, this project implements a natural language processing (NLP) model and a simple and easy to use user interface (UI) to meet the goals of the World Bank. The UI allows the user to describe information about a real estate property anywhere in the world and discover price data for that real estate. This process will allow World Bank to determine the price of housing efficiently and accurately for different countries and cities around the world specifically for uses related to the PPP.

Under the hood, the project's model relies heavily on NLP and sentence embedding of the descriptions of properties as well as considering the location and the number of bedrooms and bathrooms each property has. The UI has been built out as a React JavaScript App and will communicate with the back-end model. This is possible through utilizing Amazon Web Services (AWS) tools that include AWS SageMaker, AWS API Gateway, and Amazon S3.

Approach

Data Prep:

Focusing on World Bank's first major hurdle of data availability and accessibility, a web scraper was built to collect global real estate information for free from HomesGoFast.com. In building the web scraper, the structures of URLs on HomesGoFast.com were analyzed and broken down in a way that would allow the scraper to iterate through all property listings. The scraper created unique URLs to navigate to each listing. Specific HTML elements for different property listings were collected including property title, city, country, property URL, property image URL, number of beds, number of baths, property area and metric, price in dollars, price in pounds, price in euros and property description.

While most variables have a uniform format, the property description varies widely in length between each property. Some properties have a description of several hundred words, whereas others have less than 20. In attempt to create a standardized format, a Sentence Summarization model was utilized to transform each property description into a 130-word summary of the original description. By doing so, we reduced the upper bound of description length to 130. The Sentence Summarization model is an NLP model that is trained to create a shorter version of a document, article, or description by capturing all the valuable information and discarding the information it deems as excess. Specifically, the model was extractive, meaning it extracts relevant information from the description rather than generating new text to capture relevant information.

Methodology:

When considering the second issue raised by the World Bank, there are three major parts of the project that allow for the accurate and efficient use of global real estate data. These consist of the front-end UI, the back-end NLP sentence embedding model, and the AWS architecture. Each aspect to the project is vital in producing accurate and verified results to the user with both analytical and computer science concepts applied.

The first aspect of the project is the front-end UI which is a React App built using JavaScript. The React App is home to three different pages which the user will either see or interact with. The home page holds a user submission form for property details, the contacting model page is essentially a loading screen as the model is contacted, and the model results page displays information about the top three comparable properties to the user.

The home page of the front-end provides the model with the information for the type of property the user is querying. A screen capture of the home page can be found in *Figure 1* of the Appendix. The form contains different fields describing a property that include the location of the property consisting of country and city, the number of bedrooms and bathrooms of the property, and a short description of the property that should include any keywords or key features that the user desires for the property. The country field is a drop-down menu that includes the list of countries that the project currently has property data for, and the number of bedrooms and bathrooms are drop down menus ranging from one to ten. The city and description fields are blank input fields that accept text inputs from the user.

The contacting model page is not to be interacted with by the user and is simply a placeholder or loading screen for when the model is running in the back-end and communicating those results back to the UI. A screen capture of the contacting model page can be seen in *Figure 2* of the Appendix which shows the dynamic visualization that portrays to the user that the model is running as well as some short phrases to keep the user engaged with the UI. The dynamic visualization is imported from a library of react loading spinners.

The model results page, shown in *Figure 3* of the Appendix*,* is where the top three comparable properties according to the user inputs on the home page are visualized for the user to interact with. The model results page consists of three separate forms that each list an individual property and the corresponding property details. The model results page populates an image of the property, the location of the property including country and city, the price of the property in three different currencies including dollars, pounds, and euros, a summarized description of the property, and a similarity score and cost multiplier. The similarity score and cost multiplier will be explained further in the analysis section of this paper.

The model results page has two unique features that are incorporated into the JavaScript code. When viewing the results, the user can click on the image of the property, and they will be re-directed to the actual property listing on HomesGoFast.com in a new window. This allows the user to explore the property at greater lengths if they desire. If the user wants to read more than the property summary provides, hovering over the full description tooltip will present the complete property description.

The second aspect of the project is an NLP model that utilizes sentence embedding. An NLP model is an artificial intelligence technology that enables machines to read, decipher, understand, and make sense of human languages. Sentence embedding is one of many different techniques for an NLP model to utilize and involves mapping sentences to vectors and numbers so that computers can make sense of the sentence. For this project specifically, the property descriptions are the main variable of interest, and the descriptions were mapped over 768 vectors.

When the user inputs the specifications of a property of interest, the description, with the help of the AWS services previously mentioned, is processed through the same NLP model that was used to create the database of global properties from HomesGoFast.com. The user description is mapped to the same 768 vectors so the computer can read the description and then these vectors are compared to every property that is stored in the database. The algorithm iterates through each property and calculates the cosine similarity score between the user description vector and the vector of each database. Details of this scoring metric are included in the Analysis section.

The final aspect of the project is the AWS architecture, visualized in *Figure 4* of the Appendix. This project utilizes three major AWS services including Amazon S3, Amazon API Gateway, and AWS SageMaker. Documentation and the code for the project itself is referenced in *ZIP File 1* in the Appendix. A more detailed explanation of how to implement these services can be found in the ZIP file in the "Integrating AWS Services" folder.

S3 stores all data for the project in the cloud. This includes the NLP model deployed from SageMaker, the inference endpoint data, and the React app. S3 can host the React app as a static website allowing anyone to access it making a great solution for an international organization like the World Bank.

API Gateway is a service that allows an organization to create an API (Application Programming Interface) to allow an external internet-connected application to communicate with an internal service. This project utilizes this service by connecting the deployed model's endpoint to the React app discussed in *ZIP File 1* in the Appendix. The React app sends a get request containing property information. This data gets sent to API Gateway and formatted in a way the model can interpret it. In the case of this project, it uses the JSON (JavaScript Object Notation) data format. Once the model receives this data, it makes a prediction and sends the data back out through the API Gateway in JSON. The React app then formats this in a way that is visually appealing to the user.

SageMaker is the cornerstone of the project's operations. This service allows users to prototype, develop, and deploy machine learning models in the cloud. This service supports both R and Python. The model was developed in a SageMaker notebook or instance (specifically a ml.t2.large instance) using Python 3.6. Using the SageMaker notebook, the team was able to deploy a model to an endpoint that can be pinged either within the notebook itself or connected to the API Gateway to communicate with the front-end application discussed in *ZIP File 1* in the Appendix. The benefit of using SageMaker is that there are many pre-built models that you can customize and build upon. Models can be found in libraries like TensorFlow, PyTorch, and HuggingFace.

All three of these AWS services do cost money to use. The current setup of the project uses the following pricing tiers:

AWS Service	**Price**
S3 Standard	$0.023 per GB
API Gateway REST API	$3.50 per million requests
SageMaker Notebook Instance (ml.t2.large)	$0.10 per hour
SageMaker Real-Time Inference Endpoint (ml.m5.large)	$0.115 per hour

Figure 5: AWS Pricing

Analysis:

The objective of the project is to simplify the process of determining the cost of housing in different countries and cities. The measure of success for the output of this project is how accurate the comparable properties are when compared to the initial user inputs. Calculating the extent to how comparable the properties truly are, lies in the "Similarity Score" metric which can be found near the bottom of each property form on the model results page of the UI. The similarity score is calculated by the model using a cosine similarity function. Specifically, the cosine similarity function takes the dot product of the 768-dimension vector generated from the user input and the 768-dimension vector of each property in the data base. Then, this result is divided by the product of both vector's length. This returns a score between 0 and 1, and for the score outputted by the model, this number was multiplied by 100 to show the score as a number between 0 and 100. To get the final output of similar properties, all properties in the database are sorted in descending order according to their cosine similarity scores, and then returned to the user on the front-end results page. A similarity score of zero would indicate that the property is the least comparable to the type of property that was inputted by the user, and a score of one hundred would mean that the property is basically an identical match to the user inputs.

The other metric that is included with each comparable property is the cost multiplier. From the top three most comparable properties, the cost multiplier aims to show what sort of price range properties with very similar attributes can have. For example, there will always be a property with a cost multiplier of 1.0x and this property will have a price that is the median of the three properties. Of the other two properties, one will have a cost multiplier that is less than one and another one will have a cost multiplier that is greater than one. Providing this range of prices will hopefully give World Bank the opportunity to accurately determine the cost of housing in each country.

Discussion of Results

Recommendations:

For this project, web scraping was required to obtain the data needed to produce a dataset of global property listings. This project is currently limited to a dataset of only 4,500 records. The model is a pre-trained NLP model, so the relatively small number of records in the dataset does not directly affect its performance. However, the model is currently being restricted in its ability to accurately filter down comparable properties. With more data, the model will be able to accurately filter down on specific geographic locations and provide the World Bank with even more accuracy in determining the housing portion of the PPP.

Additionally, the advantage of using AWS cloud services allows the project to be improved upon by leveraging other services on the platform. Some recommended services include Rekognition, Translate, and RDS. Rekognition uses image recognition technology and could be applied to the project to match properties based on pictures alone. Translate is a service that translates text. Given the World Bank is an international organization, it may be beneficial to cater the application to people of many different languages which is where Translate could come in handy. RDS is AWS's Relational Database Service and would allow the World Bank to host its database on this platform. This may speed up how the model operates as it currently loads in a pre-processed CSV file containing the property information. There are many more services out there, but the services mentioned above are the most relevant given the current state of the model and front-end application.

Value to World Bank:

This project brings value to World Bank in a variety of ways with each aspect impacting World Bank differently. This project is cost-efficient, versatile, and user friendly because it is hosted on the cloud and the model is pre-trained.

AWS employs a shared responsibility model that is attractive for users in many ways. AWS allows World Bank the ability to not have to dedicate servers and hardware to the PPP initiative while also providing great security benefits. The ability to segment different product groups into multiple AWS organizations is also a huge benefit to World Bank. If the project is reproduced to fit the needs of different product groups, these models and notebook instances can all be separated to ensure no crossover between product groups. Lastly, hosting the UI and the model on AWS cloud services is an extremely cost-effective solution for World Bank with pay as you use pricing.

Although the current focus of the project is on housing and global real estate prices, the project's architecture is set up in a way that a batch of inputs can be processed to return different types of output for the many product groups that are included in the PPP. Given the nature of the model being an NLP with sentence embedding techniques, the model is just slightly tailored to housing with a few modifications. These modifications can be changed in the future to meet the needs of a new product group while maintaining generally the same model.

The project also consists of two extremely user-friendly ways to interact with the model. As mentioned throughout the report, the front-end UI is the main source of contact for users to get certain results they are searching for. However, for more technically advanced users, there is a back-end UI, which is shown in *Figure 6* of the Appendix, in which they can interact directly with python scripts that connect to the model. From this back-end UI, users can download CSV files containing all the property information for the comparable properties they have found.

Appendix

Figures:

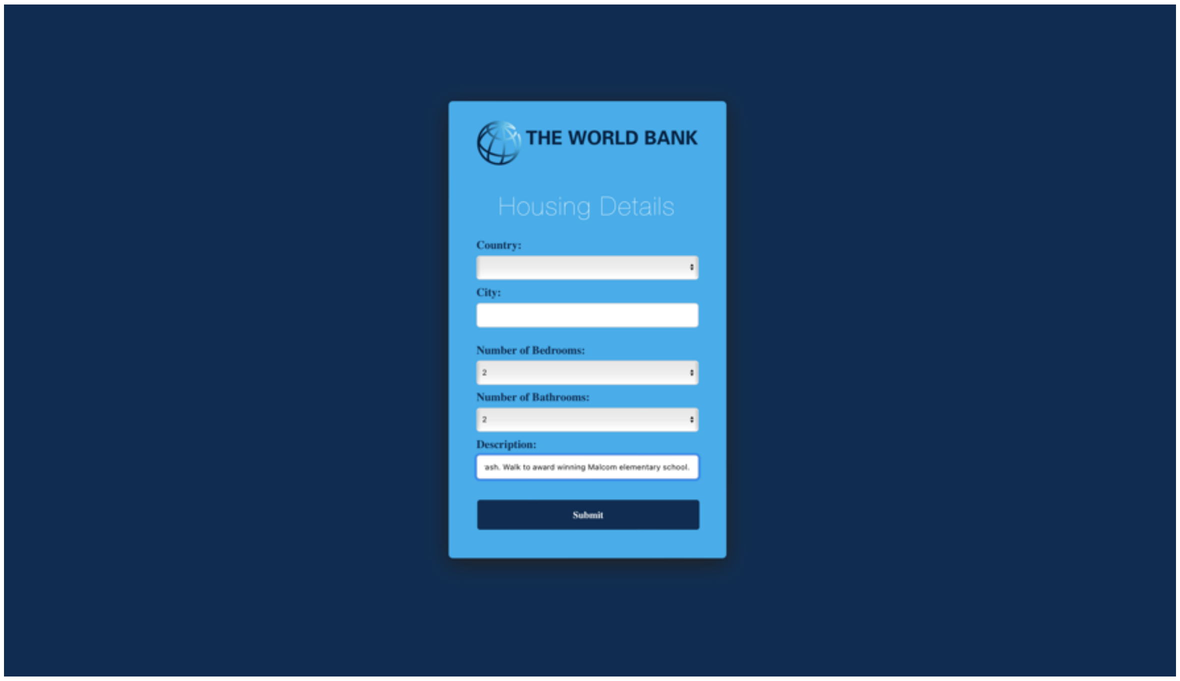


Figure 1: Home Page of the User Interface

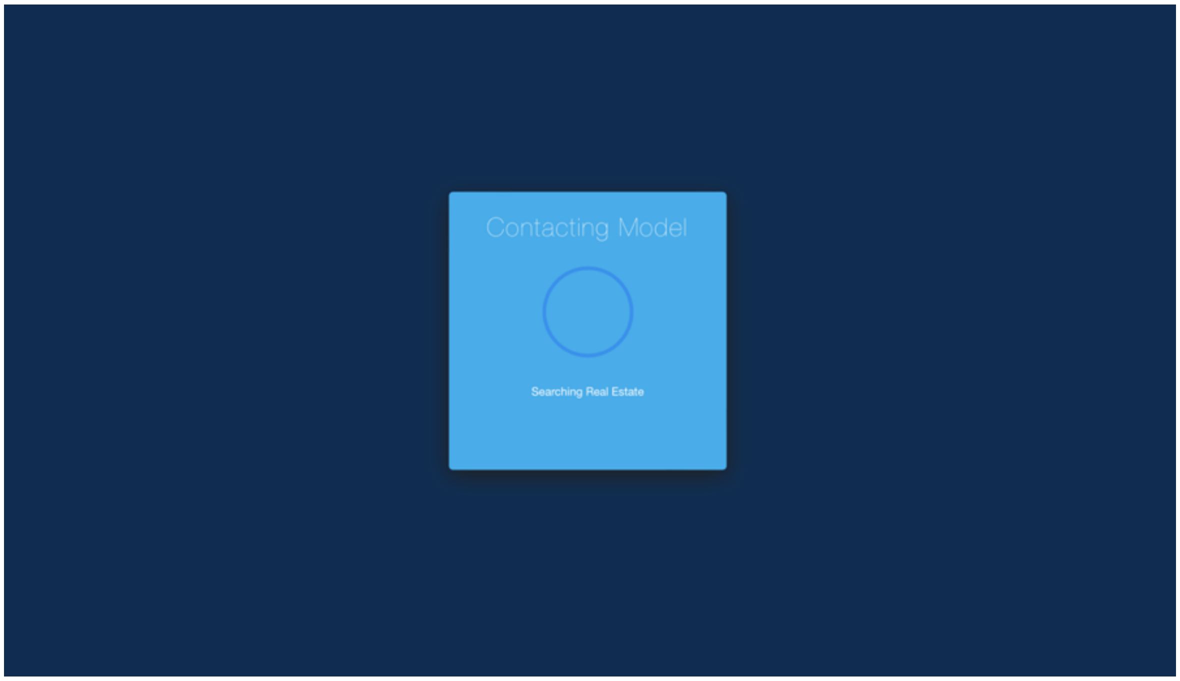
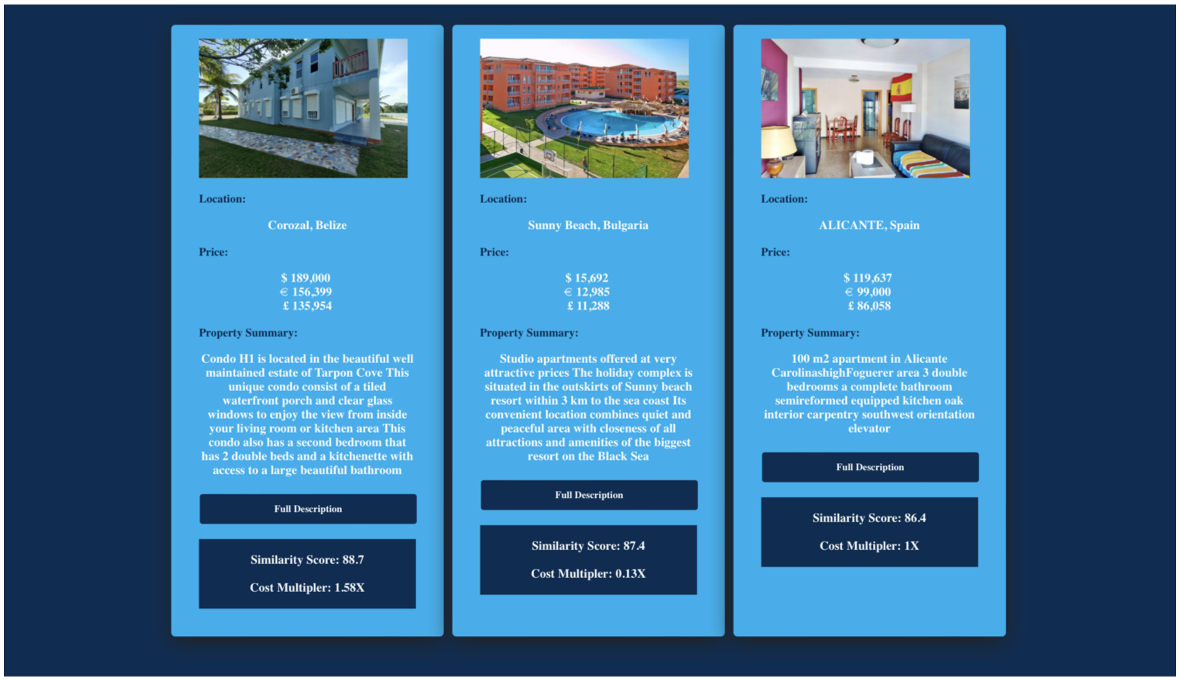


Figure 2: Contacting Model Page of the User Interface



Location:

Corozal, Belize

Price:

\$ 189,000
€ 156,399
£ 135,954

Property Summary:

Condo H1 is located in the beautiful well maintained estate of Tarpon Cove This unique condo consist of a tiled waterfront porch and clear glass windows to enjoy the view from inside your living room or kitchen area This condo also has a second bedroom that has 2 double beds and a kitchenette with access to a large beautiful bathroom

Full Description

Similarity Score: 88.7

Cost Multipler: 1.58X

Location:

Sunny Beach, Bulgaria

Price:

\$ 15,692
€ 12,985
£ 11,288

Property Summary:

Studio apartments offered at very attractive prices The holiday complex is situated in the outskirts of Sunny beach resort within 3 km to the sea coast Its convenient location combines quiet and peaceful area with closeness of all attractions and amenities of the biggest resort on the Black Sea

Full Description

Similarity Score: 87.4

Cost Multipler: 0.13X

Location:

ALICANTE, Spain

Price:

\$ 119,637
€ 99,000
£ 86,058

Property Summary:

100 m2 apartment in Alicante CarolinashighFoguerer area 3 double bedrooms a complete bathroom semireformed equipped kitchen oak interior carpentry southwest orientation elevator

Full Description

Similarity Score: 86.4

Cost Multipler: 1X

Figure 3: Model Results Page of the User Interface

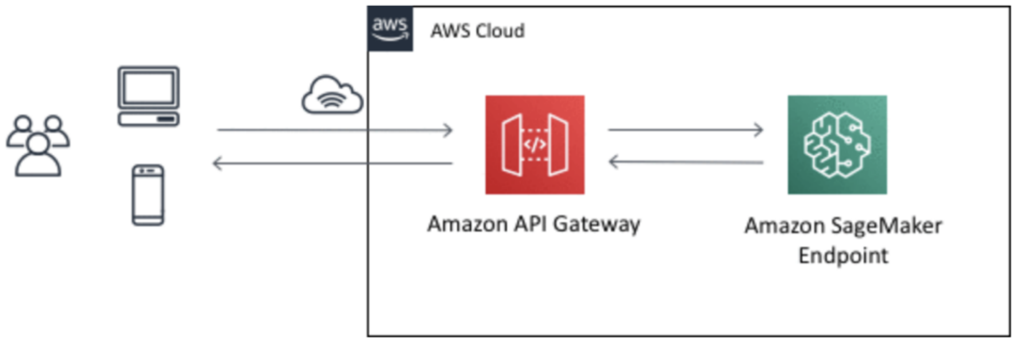
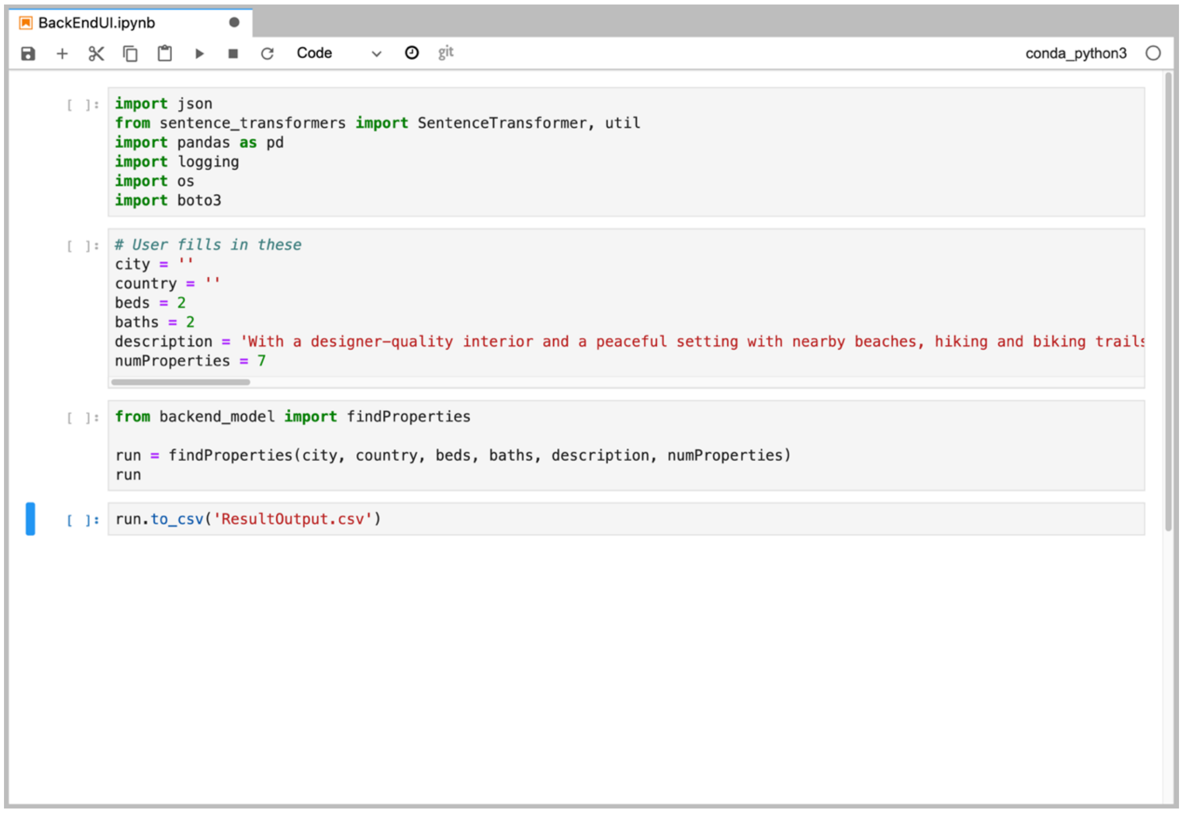


Figure 4: AWS Architecture

Service	Price
S3 Standard	$0.023 per GB
API Gateway REST API	$3.50 per million requests
SageMaker Notebook Instance (ml.t2.large)	$0.10 per hour
SageMaker Real-Time Inference Endpoint (ml.m5.large)	$0.115 per hour

Figure 5: AWS Services Pricing



```
[ ]: import json
     from sentence_transformers import SentenceTransformer, util
     import pandas as pd
     import logging
     import os
     import boto3
```

```
[ ]: # User fills in these
     city = ''
     country = ''
     beds = 2
     baths = 2
     description = 'With a designer-quality interior and a peaceful setting with nearby beaches, hiking and biking trails
     numProperties = 7
```

```
[ ]: from backend_model import findProperties

     run = findProperties(city, country, beds, baths, description, numProperties)
     run
```

```
[ ]: run.to_csv('ResultOutput.csv')
```

Figure 6: Back-End User Interface

Code/Documentation:

ZIP File 1: “World_Bank_PPP_Documentation” ZIP file submitted with this report. Once unzipped, there are README.md and README.txt files containing instructions that detail the contents of the folders in the project, how to set them up, and how to run the applications.