CSC 7062 Final Report – Kyle Carson,40180325,kcarson09@qub.ac.uk

Project Outline

The dataset used for this project was 'AirBnB listings in major US cities' and can be found at AirBnB listings in major US cities | Kaggle. The project entailed building a relational SQL database around this dataset, coding a REST API that queried this database and then subsequently building a functional website utilising the data stored within the database which is obtained by calling the API. The theme of the website is an AirBnB type site whereby users can search properties in a range of US cities, favourite properties that they are interested in and also make reservations for them. More details on the full functionality will be detailed later in the report. A video demonstration of the final project was also created and can be found at https://web.microsoftstream.com/video/d639af28-3fab-4d70-a5a3-9cb04c30c371.

Processing of Dataset

The first stage of the project was designing an initial database and subsequently inserting the data from the csv file into those tables by using an array of PHP run once scripts (these can be found in the source code submission under the folder 'FinalProjectPreProcessing'). The run once scripts read the csv file, pre-processed the data, removed any unknown characters and deleted any whitespace within the data. Once the pre-processing was complete, a connection was established with the database and a SQL insert query ran, which inserted the data into the relevant tables. Some additional fields were added to the dataset to help it better fit the theme of the website . For example, a review score for the property and for each host was added, as well as adding a 'response rate' column for the host. These values were randomly generated using the PHP random function. Within the database, each property listing had a thumbnail URL. It was found for the main property page that this image was too small and was very pixelated. However, it was found by manipulating the last word of the image url from 'small' to 'large' combatted this problem and as a result an additional column called 'PropPageImage' was created with this larger image held. As well as this, a 'Number of Likes' column was created, initially setting the value to 0 for each listing. This value is incremented every time a user favourites a property. As well as this, any column that contained empty data was filled with a relevant placeholder. For example, a number of the listings within the initial dataset lacked a thumbnail image and a price. This was combated by inserting a placeholder image into the database and also creating a randomised price using the random PHP function.

Database Design

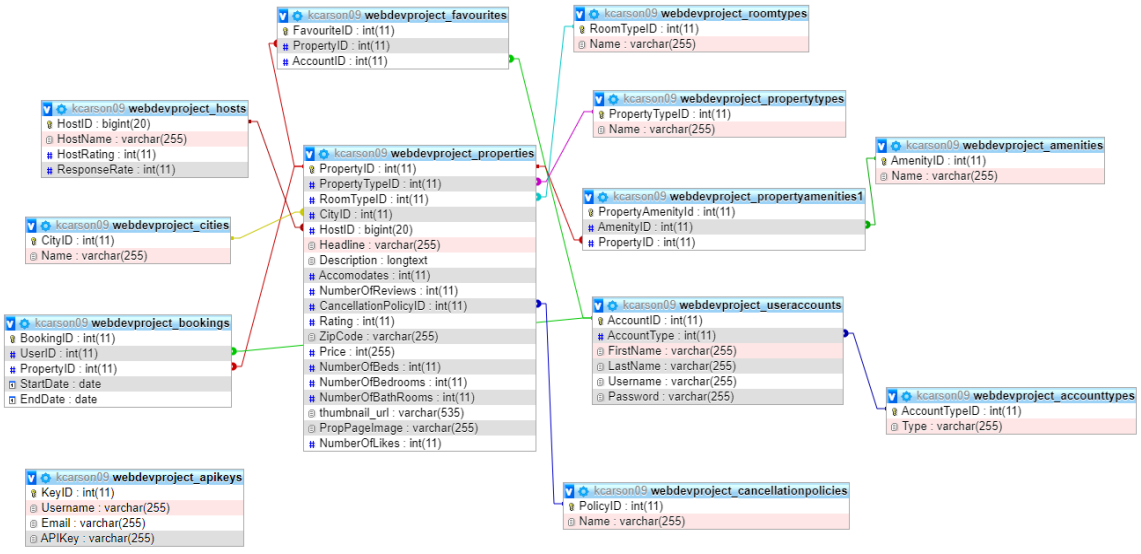

Figure 1 – Final ER Diagram of Implemented Database

Figure 1 shows the final ER diagram for the database system created.The main table within the database is the properties table. This contains all the information relating to each property listing. The PropertyID is the primary key and is unique to each property enabling a user to easily query the database for a particular property. The PropertyID is also used as a foreign key within some other key tables such as the bookings and favourites tables. This ensures each booking and favourite entry created has a property listing associated with it and also enables the system to be easily queried to obtain all the property details for a particular booking/favourite. Some normalisation was developed within the system. For example the different cities that the properties are located, the different room and property types as well as the hosts are normalised out into their own separate tables. The primary keys of these tables are all used as foreign keys within the properties table linking the tables in question together. The advantage of such normalisation is that it prevents the redundancy of data within the system and enables easier updating of data, insertion of data and deletion of data. For example if a Hosts information such as their response rate or rating where to be updated (which is quite common in a site like this), only one row within the hosts table will need to be updated rather than updating all the rows in the properties table that had that host if there was no normalisation. Another good example of normalisation is the amenities and property amenities tables. In the initial dataset, there was a single column for amenities which contained a list of all the amenities that the property had. During the pre processing stage, this list was processed, removing and brackets/whitespace and subsequently each amenity added to an array. The array_unique PHP function was then ran to remove any duplicate amenities. The final array was then added to the amenities table where each amenity has a unique primary key named AmenityID. The AmenityID is used as a foreign key within the property amenities table, as is the PropertyID. This database design prevents the unnecssary repitition of data (for example multiple properties will use the same amenity) and also ensures that each property can have numerous amenities linked to it as intended from the initial dataset.

REST API

The next stage of the project was developing a REST API that was capable of obtaining all the relevant information within the above database system as well as updating/deleting the data. The design decision was taken to divide the API file into different folders – each relating to the different functionalities of the site. For example the API contains a Property folder which contains API requests that obtains information about the property and an Accounts folder which contains requests that obtain data relating to the user accounts. The API can be used by any developer, however each request for the API requires an API key. This enables an additional layer of security for the dataset. If the requester inputs an invalid key or no key at all, they will not be able to access the data. The API keys are randomly generated and are unique and are stored in the apikeys table. The key used for this website is '82afda-89b15f-721c0c-843fb6-debbf7'. This is already included in the API URLs detailed below. However, any developer will be capable of obtaining their own API key by visiting http://kcarson09.lampt.eeecs.qub.ac.uk/mybnb2/generateapi.php and logging their credentials. The system only allows for one API key per user. The whole website developed obtains its data by calling the API and allows the system to create, read, update and delete the data within the database. Some examples of the main API queries are listed below, the full API code can be found in the folder 'mybnbapi' which will be uploaded in the project source code section however. Each of API calls below return a JSON response with the first entry being the total number of results where appropriate, and the subsequent entries providing all the details of the results obtained from the query. Each of the search calls return a maximum of 20 properties – this is how many are shown per page on the website. The page parameter in the API URL dictates what results are returned i.e page one returns first 20, page 2 returns next 20 after that etc. More detail on this will be included in the video demonstration.

Searches API URLS :

All Property (Displays all properties within system)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/searches/allproperty.php?page=1&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Basic Search (Los Angeles, accommodates >=1)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/searches/searchapi.php?location=2&accomodates=1&page=1&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Sort Search (Sorts above search by price)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/searches/sortsearch.php?location=2&accomodates=1&page=1&sortby=Price&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Filter Search (Filters above search by flexible cancellation policy, apartment and contains a TV)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/filtering/filters.php?location=2&accomodates=1&page=1&key='82afda-89b15f-721c0c-843fb6-debbf7'&proptype[]=1&cancelpolicy[]=1&amenities[]=1

Property API URLS :

Property Details (Returns all details of specified property)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/property/property.php?propertyID=109&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Property Amenities (Returns all amenities of specified property)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/property/propertyammenities.php?propertyID=109&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Trending Properties (Returns top 5 most liked properties. By adding cityID parameter you can refine this call to a specified location eg cityID = 2 for LA)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/property/trendingproperties.php?key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Names API URLS :

Cities (Returns all cities within database)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/names/cityNames.php?key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Amenities (Returns all amenities within database)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/names/amenities.php?amenities&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Accounts API URLS :

Favourites (Returns all favourites of user with ID = 1)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/accounts/favourites.php?userID=1&page=1&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Favourites (Inserts specified property into specified users favourites table)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/accounts/deletefavourite.php?userID=1&propID=27759&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Favourites (Deletes above favourite for specified user)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/accounts/deletefavourite.php?userID=1&propID=27759&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Bookings (Returns all bookings of a user with ID = 1)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/bookings/viewbookings.php?userID=1&page=1&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Bookings (Creates a booking for specified user)
http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/bookings/createbooking.php?propID=2539&startdate=%272021-04-29%27&enddate=%272021-04-30%27&userID=1&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Bookings (Deletes the booking listing above)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/bookings/deletebooking.php?userID=1&propID=2539&startdate=2021-04-29&enddate=2021-04-30&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Admin (Please see source code for more examples such as updating and creating listings) :

Roomtypes (Returns all room types stored within database)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/admin/listingoptions.php?roomtype&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Hosts (Returns details of all hosts within system)

http://kcarson09.lampt.eeecs.qub.ac.uk/mybnbapi/admin/listingoptions.php?hosts&key=%2782afda-89b15f-721c0c-843fb6-debbf7%27

Functionalities of System

The home page of the system developed can be found at http://kcarson09.lampt.eeecs.qub.ac.uk/mybnb2/. Navigation to the other pages utilising all of the sites features within the site is self-explanatory but is detailed further in the video demonstration. Below, the credentials for a basic user account and an admin account is detailed.Both accounts can be logged in by visiting http://kcarson09.lampt.eeecs.qub.ac.uk/mybnb2/login.php . Alternatively a new user account can be created at http://kcarson09.lampt.eeecs.qub.ac.uk/mybnb2/createaccount.php . When a user submits the login form, an API call will be ran to verify the credentials and if they are not valid, the user will not be logged in and the user notified to attempt again.

Basic User Account :	**Admin Account :**
Username : rashy97	**Username :** kyle97
Password : password123	**Password :** kylers

Below is a listed summary of the proposed functionalities declared in the mid term assessment followed by the actual functionalities implemented in the final project which can be explored further by visiting the site.

Proposed Functionalities In Mid Term assessment

- Users can browse an array of properties in different locations
- Users can filter their search results to meet their needs (eg by property type,amenities etc)
- Users can make a reservation for a property and have their booking details store in the database

- An accounts based system implemented, where each user can register their details and create an account enabling them to view all of their bookings once logged in
- A user with an account could also have the ability to favourite properties

Final Implemented Functionalities

- All public users can search for properties by filling out a search bar form with two fields – number of guests and location. The location dropdrown is dynamically coded whereby the values are obtained from the database and so if a new entry is added, it will automatically update in the dropdown. The search results are divided up into pages, where each page has a maximum of 20 properties. Users can subsequently select a property and open a dedicated page to it , giving more information on the listing
- All public users can 'sort' their search by three main fields – price (showing lowest first), rating (highest first) and most popular (most likes)
- All public users can refine their searches by using a series of filters inlcuding property type, amenities and cancellation policy
- Users can create an account which stores all their details and gives them additional functionalities. The users passwords are protected using MD5 encryption within the database
- Users logged in with an account can favourite properties, make bookings and view these bookings/favourites anytime upon login. These pages are blocked from non logged in users – they will just be redirected to the home page if they try and manipulate the URL. This enables security of each users data
- The home page displays trending property listings – these are properties with the most number of user likes and updates dynamically as users continously use the site and favourite properties
- An Admin role has been added – this is a user account but with additional functionalities. An admin logged in to the system can create, edit and delete property listings

Future Developments

Given more time with this project there are additional functionalities that I would of liked to have implemented enhancing the overall user experience. In particular, I would of liked to make more use out of the hosts table stored within the database. For example I would of added the ability to create a host account. This would allow each host to create,delete and update property listings and also view statistics on all of their current listings such as number of bookings, number of unique views and number of favourites. To enhance the user experience further, I also would of liked to make use of the lattitude and longtitude columns within the initial dataset. These could be used by generating a google map using the Google Map API and pinpointing the location of each property on the map. This would enable an alternative way for the user to browse properties closer to their desired location within a city – for example someone may want to stay within walking distance of a football ground. Another additional feauture to make the site more life like, would be to implement a payments based system for each booking and also send a confirmation email to the user upon booking of a property. The security involved in payments however was beyond the scope of the course. One other final suggestion would be to add the ability for a user to review a partiular property listing and for other users to view these reviews. Currently the system only displays the rating values that were provided in the initial dataset.

References

1 https://getbootstrap.com/docs/4.0/layout/grid/

2 https://getbootstrap.com/docs/4.0/components/navbar/

3 https://getbootstrap.com/docs/4.0/components/jumbotron/

4 https://getbootstrap.com/docs/4.0/components/card/

5 https://getbootstrap.com/docs/4.0/components/modal/

6 https://unsplash.com/

7 John Busch 7062 Week 11B Lecture (for generating API keys)