****

**CITI TECHNOLOGY HACKATHON 2018**

**Financial Planning of Real Estate**

**Prepared by:**

**Andy Lam**

**Kenneth Loh**

**Yong Han Jun**

**Executive Summary**

**Overview**

CitiFinder is a web-based application that allows prospective home buyers in Singapore to search for houses that they are eligible to purchase, based on their current financial situation.

**Problem**

Prospective home buyers unsure of eligibility and various factors that would affect their affordability. There are many rules and regulations that apply when deciding whether to purchase a house in Singapore. These regulations are further complicated depending on the buyer’s financial profile and whether he/she currently owns a property.

To cut through all the technical jargon and ease the process of searching for a house, we propose a technical solution to help prospective home buyers analyse financial situation and eligibility to allow them to identify and compare suitable properties.

**Solution**

Typically, one would first calculate their limit on the mortgage loan that they can borrow. Consequently, he/she would have to source for the house, and depending on their preference, factor in the choice of engaging the services of a property agent to smoothen the process. Furthermore, either a HDB concessionary loan or bank loan is needed to be approved (Au-Yong, 2017). Hence, we propose the following solution.

A simple and straightforward web-based application that works in three steps:

1. To gather the basic financial information required to assess affordability
2. Based on the current affordability, user can input the preferred location and house specifications
3. The top three houses will be displayed, and the rest of the houses accordingly

**Scope:** The challenge will be limited to houses located in Singapore. The factors that affect the choice of the users are limited to those that we have decided to analyse.

Assumptions:

1. Bank’s valuation of property price is the same as the listed price
2. No refinancing is considered

**Technology Stack**:

Language: Python, HTML, CSS

Database Management System: MySQL (using sqlalchemy)

Frameworks: flask, sqlalchemy, pandas, xlrd, xlsxwriter, flask-track-usage

**Sources:**

#### AU-YONG, R. (2017). *Do your homework before you buy that first home*. *The Straits Times*. Retrieved 9 September 2018, from <https://www.straitstimes.com/singapore/housing/do-your-homework-before-you-buy-that-first-home>

#### *Creating a Web App From Scratch Using Python Flask and MySQL*. (2015). *Code Envato Tuts+*. Retrieved 9 September 2018, from <https://code.tutsplus.com/tutorials/creating-a-web-app-from-scratch-using-python-flask-and-mysql--cms-22972>

#### *Guide to Using CPF for Private Residential Properties* . (2018). *Edgeprop.sg*. Retrieved 9 September 2018, from https://www.edgeprop.sg/property-news/guide-using-cpf-private-residential-properties

#### *Login authentication with Flask – Python Tutorial*. (2018). *Pythonspot.com*. Retrieved 9 September 2018, from <https://pythonspot.com/login-authentication-with-flask/>

#### 

Flask-track-usage documentation: https://flask-track-usage.readthedocs.io/en/latest/

**Setting Up Web Application:**

1. Install all the necessary Frameworks - pip install
   * flask
   * sqlalchemy
   * xlsxwriter
   * xlrd
   * pandas
   * flask-track-usage
2. Open Command Prompt - cd to folder prototype
3. Create the random generated data - run rbg.py
4. Create .db file - run tableDefine.py
5. Force the data from xl to SQL - run xlToSQL.py
6. Open web application - run webApp.py
7. Open [localhost:5000](http://localhost:5000) / http://0.0.0.0:5000/ (Best Viewed with Google Chrome)
8. For tracking the record, http://0.0.0.0:5000/tracking

**Web Application:**

* Proposed architecture diagram aren’t necessary the same as the web application as some of the factor is not considered in.
* With the proposed architecture diagram, a web application is being built with the most simplified version.
* Database aren’t the same as the proposed architecture of the database as we were just mock a simplified version of data.
* There will be mainly three page for the web application.
* First page consideration that doesn’t apply according to architecture diagram:
  + Married status
  + Information tab
* Second page consideration that doesn’t apply according to architecture diagram:
  + Affordability report
  + No of bedroom / no of toilet
  + Location is in coordinate wise
  + No checking of wrong location format (For now)
* Third page consideration that doesn’t apply according to architecture diagram:
  + Featured pick with photo included
  + Option to select and view the desired property
  + Filter button according to the attributes

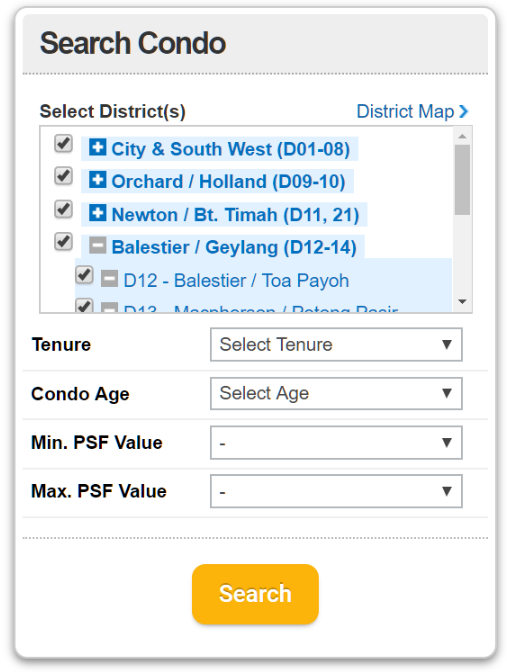
**Possible improvements to the program:**

* Link social media accounts to share about their dream house
* Utilise machine learning to predict what is the top choices
* Seamless bank loan approval from Citibank
* Improvement on GUI

**Proposed Solution**:

Upon entering the web application, users have to firstly fill up their

Input:

* Enter Personal Information
  + Estimated Personal Income (Create a range)
  + Estimated Personal Savings (including CPF)
  + Existing House
    - Estimated House Price of Existing House
  + Estimated Living Expenses
    - Credit Card Loan
    - Child Support
    - Alimony
* Existing Loan
  + Insurance
  + Mortgage Loan
* Location for Property
  + Location list similar to the image below
  + 

Formula to determine the housing loan is applicable:

* Debt to Income ratio (DTI less than 43% is a good number)
* Personal saving around 4-5% for Initial Deposit
* Existing house pricing value for Initial Deposit

Output:

List of housing information that match the personal financial information

* Price (Down-payment of the property)
* Price Per Square Foot
* Tenure
* Location
* Size of property
* Type of property
* Amenities

After view the list of housing information, user can compare different properties in the list based on

* Price
* Size
* Property Type
* Amenities

Feature to develop for the list:

* Filter function for each attribute.
* Compare Property function (Output the featured property TOP 3 preferences)

Feature 1

Feature 2

Feature 3

img

Feature 1

Feature 2

Feature 3

img

Feature 1

Feature 2

Feature 3

img

List 4

img

Feature 1

Feature 2

List 5

img

Feature 1

Feature 2

List 6

img

Feature 1

Feature 2