POLITECNICO
MILANO 1863

2022

Design and Test Plan Document

Authors:

- **Mohanad Diab**
- **Majed Alotaibi**
- **Shengshen Li**

Deliverable: DTPD

Title: Design and test plan document

Version: 1.0

Date: 25th of May, 2022

Download page: **github.com/MohanadDiab/SE4GI_Project**

Revision history

Version	Date	Change
version 1.0	25th of May, 2022	first submitted version
version ...	...	...

Contents

Introduction

The real estate industry is rapidly developing, and the influencing factors are so varied that it is difficult to evaluate the property quality and so on. Recently, real state agent aim to approach clint via online market (website) which bring several advantage; such as, clearer channels of communication, these are just some of the many benefits that real estate agents can get from having a website of their own. If an agent looking for more clients or a better marketing strategy, putting up their own website is the next step that they should do.

The main objectives of our project are by combining WebGIS technology and algorithm analysis, and to complete the prediction model analysis of the real estate quality and visualize the data through front and back-end interaction and GIS spatial data analysis by combining various influencing factors.

The goals of this system are to process and analyze the Real-estate data, by combining WebGIS technology and algorithm analysis, and to complete the prediction model analysis of the real estate quality and visualize the data through front and back-end interaction and GIS spatial data analysis by combining various influencing factors.

Acronyms and Definitions

Name	Definition
Epicollect5	Epicollect5 is a mobile and web application for free and easy data collection. It provides both the web and mobile application for the generation of forms and freely hosted project websites for data collection.
WSGI	Web server gateway interface (WSGI) is a simple calling convention for web servers to forward requests to web applications or frameworks written in the Python programming language.
DBMS	Database Management Service (DBMS) is a software that interacts with end users, applications and the database itself to capture and analyze stored data.
Web Application	Web application, or web app, is a client–server computer program that the client (including the user interface and client-side logic) runs in a web browser.
API	Application programming interface (API) is a computing interface to a software component or a system, that defines how other components or systems can use it.
UTM	Universal Transverse Mercator (UTM) is a conformal map projection system for assigning coordinates to locations on the surface of the Earth
PostgreSQL	Free and open-source relational database management system with SQL compliance. It will be used in the development of this project.
Machine Learning	Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.

Software Design Document

The Design Document provides a specification on the architecture of our website system. In fact, this document is corresponding to the Requirements Analysis and Specification Document which is previously delivered, also it provides further description on its components, their interactions and the implementation, integration and testing plan.

This document will describe the following characteristics of the project;

- Project Database: data is the most important aspect of our project also data. In fact, data could be so complicated; therefore, preprocessing and cleaning up the data it does result sufficient analysis. Upon these mentioned fact Database characteristics will be explained and described

- Software Structure: as it will be determined in this segment, the software is divided into a layers architecture which will make the best possible interaction between the client and the server taking in account the static and dynamic elements

- User cases application: As it can be seen in further detail in the RASD, it can be seen how use cases or requirements map on the components of the software.

- Test Plan: A test case is a singular set of actions or instructions to perform and validates a specific aspect of a product or application functionality. A detailed document that describes the test hypothesis, input from all users strategy, and steps

1.Project Database

The project data, were retrieved from EpiCollect5, pre-processed and copied to a PostgreSQL database. The web app it does interact with DBMS and perform CRUD operations on the PostgreSQL database. **The** advantages for storing the data in a PostgreSQL database as opposed to fetch them directly from Epicollect5 include:

- to enable verification, pre-processing and storing of consistent data;
- to ensure availability of data, decoupling our web application from EpiCollect5;
- to reduce the risk of data loss;
- to improve performance;
- to leverage DBMS capabilities and, in particular, the interface between Python and PostgreSQL.

1.2 Epicollect5 Dataset

The dataset of "Analysing the housing quality in Crafers Stirling and Aldgate". that will be used in this project comes from Epicollect5 and contains data collected in;

Australia.
The dataset consists of 72 georeferenced measurement points of ERP geolocalized in living out of Crafers, Stirling and Aldgate

Dataset: https://five.epicollect.net/project/housing-quality-index-crafers-aldgate-stirling

We retrieved data from an existing project whose main features are shown below table and we modified the attributes of the dataset according to the main purpose of our application.

Attribute	Description
Created At	expressed in m/dd/yy
Location	expressed by utilizing UTM Coordinates
Take photo	Photo of the property
Dwelling type	The Property house type
Number of trees	Trees Number within the property
Distance to major junctions	Distance in Meter from the closest junction
Decibel reading	
Age of Property	The age of the property
Quality of housing	This represent the quilty of housing considering different aspect such as (Age of house , Type of the house and distance from junction)

Furthermore, we created a new project called “Reality Housing Solions” Web Application whose main features are summarized in below below;

Attribute	Description
Created At	expressed in m/dd/yy
Location	expressed by utilizing UTM Coordinates
Take photo	Photo of the property
Dwelling type	The Property house type
Number of trees	Trees Number within the property
Distance to major junctions	Distance in Meter from the closest junction
Decibel reading	
Age of Property	The age of the property
Quality of housing	This represent the quilty of housing considering different aspect such as (Age of house , Type of the house and distance from junction)

1.3 PostgreSQL database

The web application running on the WSGI server will interact with a Database Management System for data storing and management. In fact, our decision came up to use PostgreSQL since it is a free and open-source relational database management system emphasizing extensibility and SQL compliance, beside that to leverage database adapters for Python programming language,

Furthermore, PostgreSQL provides useful extensions depending on the specific nature of data.
Given that data entries are georeferenced we have decided to exploit PostgreSQL's extension PostGIS, an open source software program that adds support for geographic objects to the PostgreSQL object-relational database. PostGIS follows the Simple Features for SQL specification from the Open Geospatial Consortium (OGC). Key features of PostGIS include, among others:

● Geometry types for Points, LineStrings, Polygons, MultiPoints, MultiLineStrings, MultiPolygons and GeometryCollections;

● Spatial predicates for determining the interactions of geometries;

● Spatial operators for determining geospatial measurements like area, distance, length and perimeter;

● Spatial operators for determining geospatial set operations, like union, difference, symmetric difference and buffer.

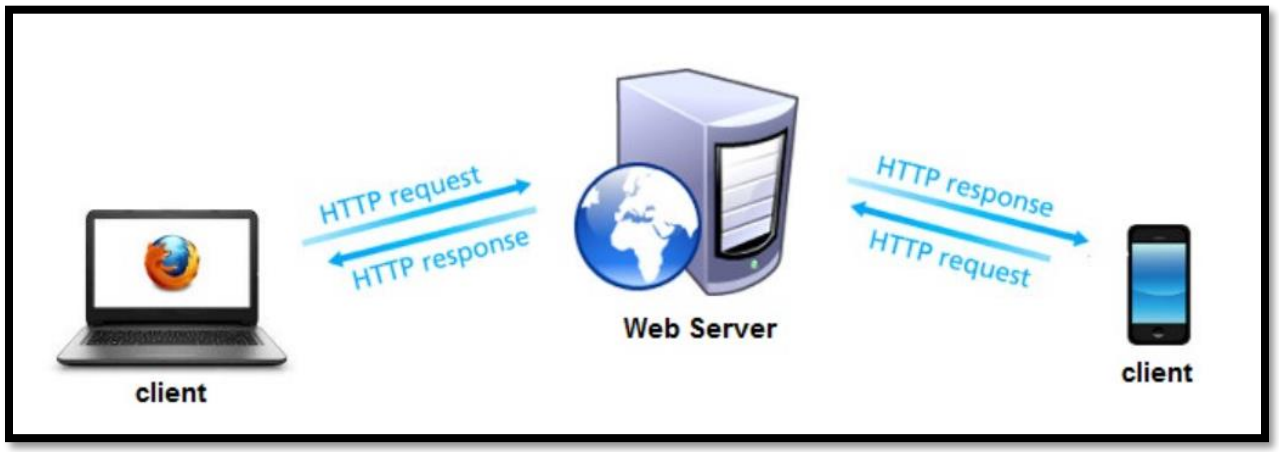
2 Software Structure

The software's architecture will consist on a 3-layer system composed of a Web Server (HTTP Server), Logical Server (Application Server) and Database Server.

- **HTTP Server**

The client is able to communicate with HTTP Server through client requests (URL, clicks, formularies, etc). This server will provide a response to the client server (user browser). The response is composed by HTTP and CSS codes that are run by the client server and the completion status to confirm whether the operation is successful or not.

The HTTP and CSS codes obtained as output from the Web Server are generated by means of requesting to the Application Server all the needed information for producing the required web page. Thus, the Application Server provides the logical outcome as a response to the HTTP Server's request.



Application Server

This server will provide all the logical operations that will be needed in order to achieve the software requirements. Therefore, Python coding will be used for this project, this is a WSGI Server, thus Python applications can run and serve the Web Server in a consistent way

EpiCollect5 Data preprocessing

When we import the data form EpiCollect5 we had to run data quality check for each attribute of the data and modify some column ;as example of that, we added colum for Latitudes and longitude for object point in order deal with Geopapanda , Also we had ensure about some Colum forma t such as the dates and must be in correct DD/MM/YYYY format.

Data import into PostgreSQL

The required data from EpiCollect5 we have been preprocessed and filtered based on our analysis need. As result of that the data were ready to be imported to PostgreSQL database

- **Mapping Tool**

  The WSGI Server will provide a user-interactive map for showing a points of each property , basemaps and geospatial data visualization. The Mapping function will receive a list of elements to be plotted as input as well as specifications of how each element has to be plotted. This function will mainly use the library which is aimed to develop interactive visualization for web applications.

- **Template Engine**

  Several Engine Template utilize for our project in order to achieve our project goals, could classified our Template Engine based on the following criteria;

- **Front-End Development**
  The front end development is programming which focuses on the visual elements of a website or app that a user will interact with (the client side). In our project the following (**Front-End**)Engine template used.

- **JINJA**
  JINJA template engine which dynamically builds HTML pages using familiar Python concepts such as variables, loops, lists, and etc. based on logic and context. Unlike static HTML, templating systems like JINJA empowers us to do things like share snippets between pages, or render pages conditionally based on context.

- **Leaflet**
  Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. it has all the mapping features that developers need.
  Leaflet is designed with simplicity, performance and usability in mind. It works efficiently across all major desktop and mobile platforms, can be extended with lots of plugins, has a beautiful, easy to use and well-documented API and a simple, readable source code that is a joy to contribute to. We utilize tools to show our map out put and Visualization.

- **CSS codes**

  CSS code is the code that styles web content, it does manage the style and the display of the template web pages like border style, color of the components, text align and etc. and it has been utilize to customize our webpage

- **The back-End Development**

  The back end development focuses on the side of a website users can't see (the server side). The following development Engine used for utilized for our project;

- **Flask**

  Flask is a friendly Python web framework that provides useful tools and features for creating web applications in the Python Language. It gives developers flexibility and is an accessible framework for new developers because you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

- **HTML**

  The HTML has been used to structure a web page and its content; such as, the content that could be structured within a set of paragraphs, a list of bulleted points, and using images as well as data tables..

- **Linux server**

  A Linux server is a server built on the Linux open-source operating system. It offers businesses a low-cost option for delivering content, apps and services to their clients. Because Linux is open-source, on our project we used it set up a PostgreSQL Database on Linux

- **Code function**

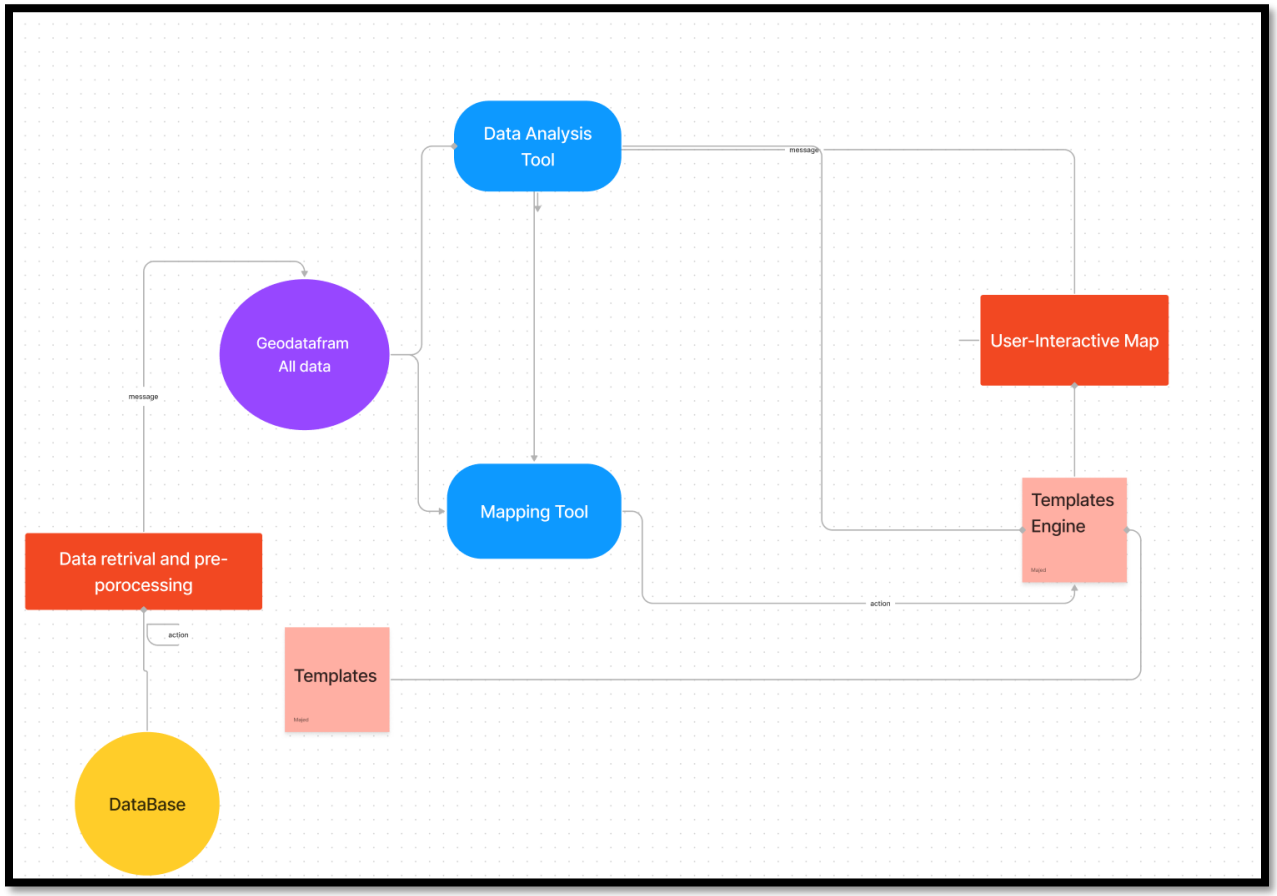
  Several code functions implemented in this project to run our webpage application , the below table classified the function and its objective

Function	Objective
Setup DB Connection	generic connection path to the server setup
Welcome page	Function that navigate the user to he website Home Page
Sign in page	Function that allow the website user to sing in by using User Name and Password
sign up page	Function that allowed new user to create user name and Password
Sing out	Function that allowed existing webpage user to exit the session
Map Functions	Function that displays and visualize the map by utilizing java script

- **Interaction between application server components**

A different components within the application server were utilize and interact with each other . First of all, the Epicollect5 specific data were retrieved and preprocessed. Furthermore, the accurate data were placed into Geodataframe, after that the data were distributed either to the Mapping Tool or to the Data Analysis Tool.

The Template Engine renders the final client-side web page. Each time this function run it, the engine will have as input some built-in base templates designed to provide a user-friendly experience in the web application.



User Cases & Scenarios

In order to explain the software functionalities, this section is going to address an explanation about the actions taken by the software and the user in a list of cases that are useful to explain the internal processes of the application. In this section we describe what is going on from server-side and client-side on when the user cases happen by specifying the different actions that take place in these situations.
Actors:

1. Visitors: To manage this website in an efficient and orderly manner, we don't allow unregister user to use this website. So, all visitors will be redirected to the register/login page.
2. Registered Users: They get the access to most of the functionalities of this website, including sending request to the server and visualize those responding data by map applications and chart/diagram, add their preferred location and see its score with regard to the prediction model.

3. Administrator (Domain experts): They can modify the relevant APIs, inspect the exception handling feedback and check the usability of each functionality.

   Further details for User Cases have been explain in "RASD_V1.1" document

Test Cases

When our Web-Page application completed, test cases will run overall the webpage. Test case can provide several advantages such as; ensure all developed function are working in the efficient way, also Increasing of the found bugs, the following test cases have been conducted;

- **Login**

Test Case Number	1.1.1
Test Case Name	Login (Successful)
Test Case Description	This test case verifies whether a user can be properly logged in to the System.
Preconditions	N/A
Process Description	
Steps	**Hypothesis**
1	User inputs valid user ID and password.
2	System validates user ID and password.
3	System show welcome page corresponding to the user.

Test Case Number	1.1.2
Test Case Name	Login (Not Successful)
Test Case Description	This test case verifies whether unauthorized users are restricted from accessing the system.
Preconditions	N/A
Process Description	
Steps	**Hypothesis**
1	User inputs invalid user ID and password.
2	System attempts to validate user.
3	System redirects user back to login page.
4	System administrator verifies whether the user is not logged in.

Test Case Number	1.1.3
Test Case Name	Login (Lock)
Test Case Description	This test case verifies whether a user cannot attempt more than three consecutive failed attempts at login with a given password.
Preconditions	N/A
Process Description	
Steps	Hypothesis
1	User inputs a valid user ID with an invalid password.
2	System attempts to validate user.
3	System redirects user back to login page.
4	User repeats steps 1-3 (inclusive) two more times
5	System locks the user ID and redirects user to the appropriate error message.
6	System administrator verifies whether the information is stored correctly in the database.

1.2 Logout

Test Case Number	1.2
Test Case Name	Logout
Test Case Description	This test case verifies whether the current user can log out of the system and verifies whether he/she can no longer access data
Preconditions	The user is logged in
Process Description	
Steps	Hypothesis
1	User inputs a valid user ID with an invalid password.
2	System attempts to validate user.
3	System redirects user back to login page.
4	User repeats steps 1-3 (inclusive) two more times
5	System locks the user ID and redirects user to the appropriate error message.
6	System administrator verifies whether the information is stored correctly in the database.

1.3 Change the Password

Test Case Number	1.3.1
Test Case Name	Change Password (Proper Values)
Test Case Description	This test case verifies whether the current user can change his/her password
Preconditions	The user is logged in
Process Description	
Steps	**Hypothesis**
1	User selects option to change password.
2	System redirects the current user to the "Change Password" page.
3	User completes fields with valid input.
4	System updates the system
5	System redirects user to the appropriate "Welcome page".

Test Case Number	1.3.2
Test Case Name	Change Password (Incorrect Password)
Test Case Description	This test case verifies whether the current user may change password with an incorrect confirmation.
Preconditions	The user is logged in
Process Description	
Steps	**Hypothesis**
1	User selects option to change password.
2	System redirects the current user to the "Change Password" page.
3	User completes fields with incorrect password for authorization but valid new passwords.
4	User repeats steps 1-3 (inclusive) two more times
5	System notifies user of error and redirects user to the "Change Password" page.
6	User attempts Test Case 1.3.1 to verify that the new password has not been entered in the system.