**Software Design Document**

**for**

**Mapping for Change**

**Graph Generator**

**Rev. 1.2**

**Prepared by Reece Walsh**

**and Daniel Herman**

**UBC Okanagan**

**March 23 2018**

**TABLE CONTENTS**

[**Introduction**](#_uurthqxqwb4k) **4**

[Purpose](#_8v75u7ldqdzc) 4

[Scope](#_w84d3dtbvn4) 4

[**Design Overview**](#_sbmyh5afmxu7) **4**

[Background Information](#_4d34og8) 4

[Alternatives](#_u8j0ia66jm75) 5

[**User Characteristics**](#_ipoadojdz12a) **5**

[**Requirements and Constraints**](#_95u5syavo07r) **5**

[Performance Requirements](#_26in1rg) 5

[Security Requirements](#_fn3ufne5izpt) 5

[**System Architecture**](#_4301eye4gyge) **6**

[**Detailed Design**](#_3dqsh6s4r7v0) **7**

[Description for Component “Report”](#_oz3dzl20qkrm) 10

[Description for Component “Database”](#_2kwqnvx813n) 10

[Description for Component “Graph”](#_vemtnmewac5c) 10

[Description for Component “Data”](#_oxx2id3p6stt) 10

[Description for Component “DataGenerator”](#_ixoog4akfa5j) 10

[**Data Architecture**](#_1y810tw) **10**

[Data Analysis](#_oe5zr84kwxiv) 11

[Output Specifications](#_wfr5zmo9o0pb) 11

[Logical Database Model](#_xpum18ygx60m) 11

[Data Conversion](#_sjjpq2rgn2yi) 13

[**Interface Requirements**](#_6hra6fbggppf) **14**

[Required Interfaces](#_qsh70q) 14

[External System Dependencies](#_3as4poj) 14

[**User Interface**](#_7nsfj55yjboo) **14**

[UI Design Documents](#_o4cxx7i521zq) 15

[Interface Design](#_z8tbcgrgdjx7) 15

[Functionality](#_wmrsj96yeg7u) 15

[**Non-Functional Requirements**](#_o7vu0k9jalhf) **16**

**Revision Sheet**

|  |  |  |
| --- | --- | --- |
| Revision Number | Date | Brief summary of changes |
| 1.0 | 02.05.18 | Baseline draft document. |
|  |  |  |
| 1.1 | 07.03.18 | Updated the sequence, activity, use case and architecture diagrams for overall program design. |
| 1.2 | 08.03.18 | Updated UI section with completion expectations and filled out applicable areas of the Data section. |
| 1.3 | 23.03.18 | Completed sections 7,8,9,10 |
| 1.4 | 28.03.18 | Added logical database table explanations |

# **Introduction**

The Mapping for Change Report Generator is designed to supply social support workers and city planners with report generation capabilities that are contained in the scope of a website hosted on a server. The generator will create these reports from Mapping for Change data provided by a database. The Report Generator will also account for changes and updates within the dataset.

In the scope of this project, “report” will be defined as a set of diagrams or graphs generated (possibly prefaced by text) from Mapping for Change data that are displayed on a web page. The landing page of the Report Generator will feature a preset “report” which will display a series of insightful and handpicked graphs or diagrams. Custom or tailored reports will also be supported on the appropriate pages.

## **Purpose**

The Mapping for Change Report Generator is designed to facilitate insight into issues surrounding homelessness and to increase support for adopting a cross-organization real-time database. We wish for visitors to take advantage of these graphs so that they may draw meaningful conclusions from available data that they might not have made on their own and to show what might be possible from further support.

## **Scope**

A report generation and delivery system will be produced for the Mapping for Change project. This system will generate illustrations (charts and graphs) to facilitate discussion of community services to the homeless and vulnerable.

The report generation utility will be hosted on a server and interface with the end user’s browser.

If further features are requested or added, this document will update to reflect their contributions and design.

# **Design Overview**

## **Background Information**

The Mapping for Change project was founded around the idea of creating an online, spatial database to catalogue and identify resources for people in need. Services such as shelters, medical care, meals, etc are recorded and displayed through a geo-coordinate system. This info is stored in an accessible and updatable database. A snapshot of the Mapping for Change database will serve as the primary data source for the Report Generator.

## **Alternatives**

In the event of web or server resource unavailability, a local, cross-platform program could replace the need for such resources.

# **User Characteristics**

Potential Report Generator users include city planners, government workers, shelter volunteers or shelter managers. Basic web navigation skills will be required to use the report generation system. These users will be assumed to be able to understand a legend and have the basic knowledge of how to read a graph.

Due to potential corporate policy in a city planner or government employee’s workplace, special development resources will be allocated towards the accommodation of Internet Explorer.

# **Requirements and Constraints**

## **Performance Requirements**

The access and creation of a report should not cause performance burden on any servers, nor should the client have to wait more than a second to receive their requested report.

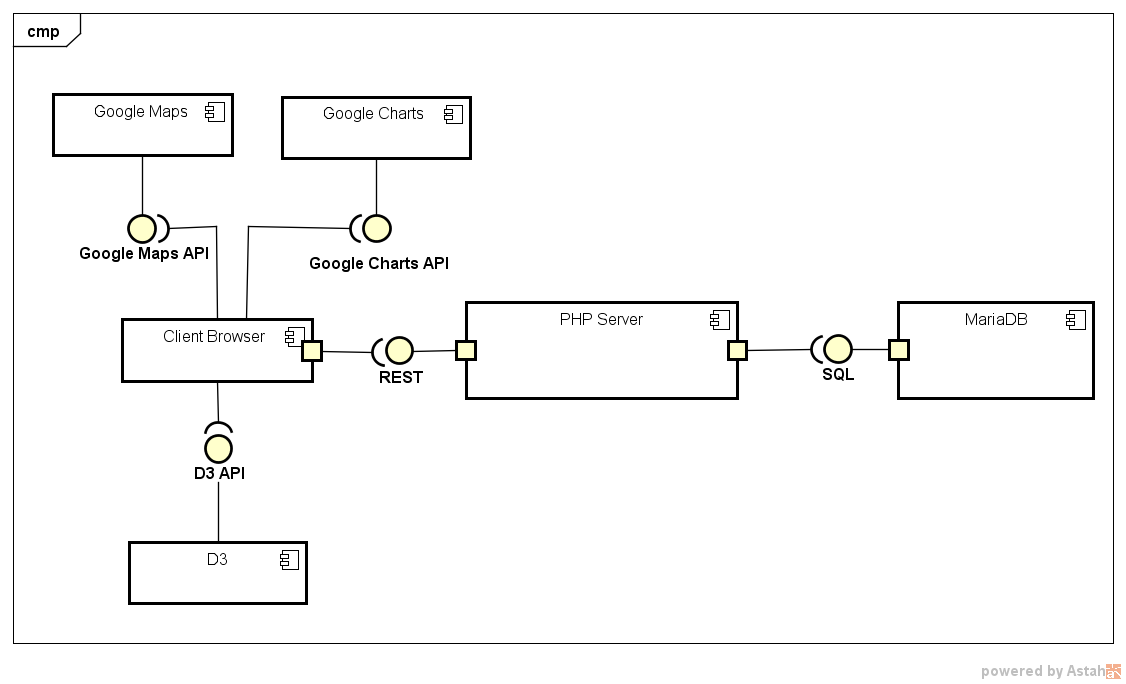
## **Security Requirements**

When displaying graphs to the user, the system sanitizes POST request variables and checks to ensure (after sanitization) that the variable is legitimate.

When querying the database for graph data, the system prevents SQL injection attacks through usage of prepared statements and input sanitization.

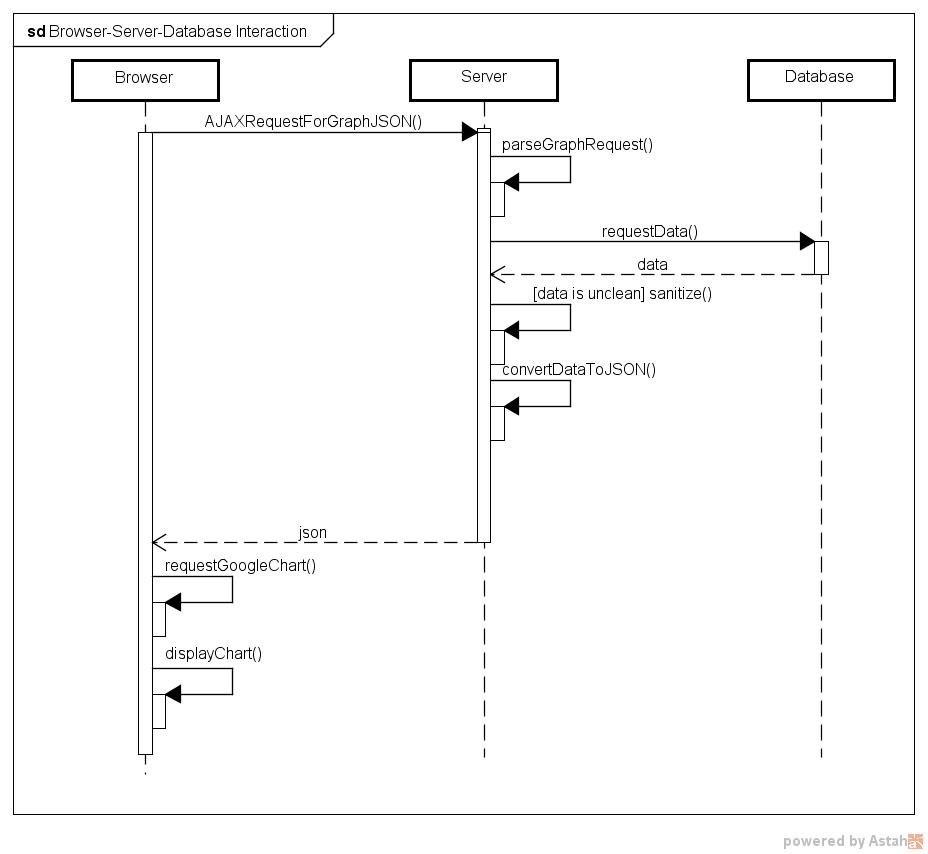
# **System Architecture**

The system architecture will consist of a front end portion (accessible and displayed within the end-user’s browser) and a back end portion (hosted on a server). The server will utilize PHP to access the database, process the data, and send it to the client, upon request.

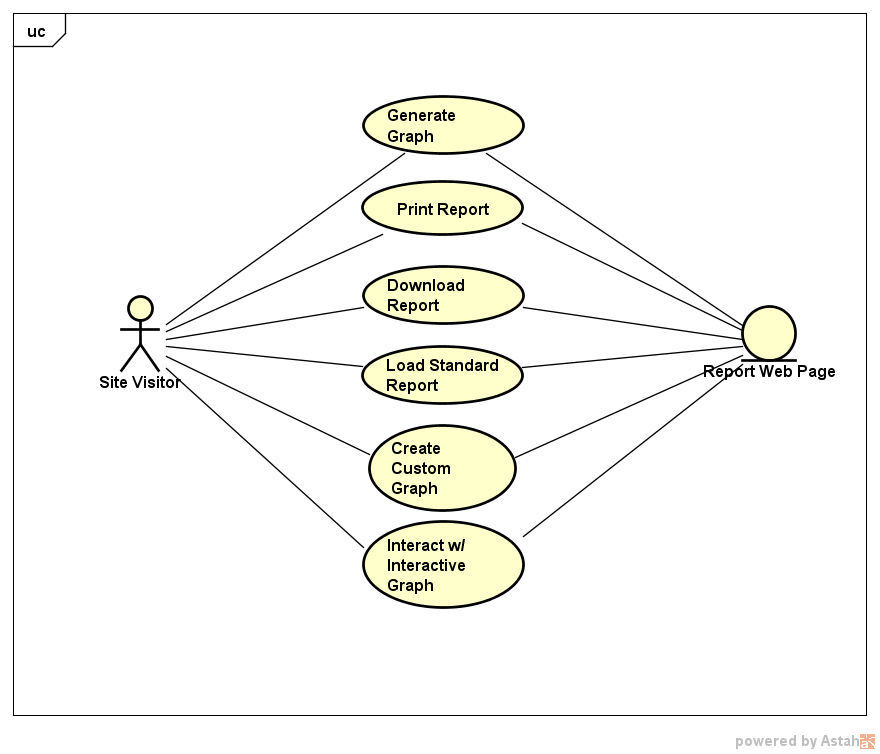


A conceptual illustration of the system architecture is shown above.

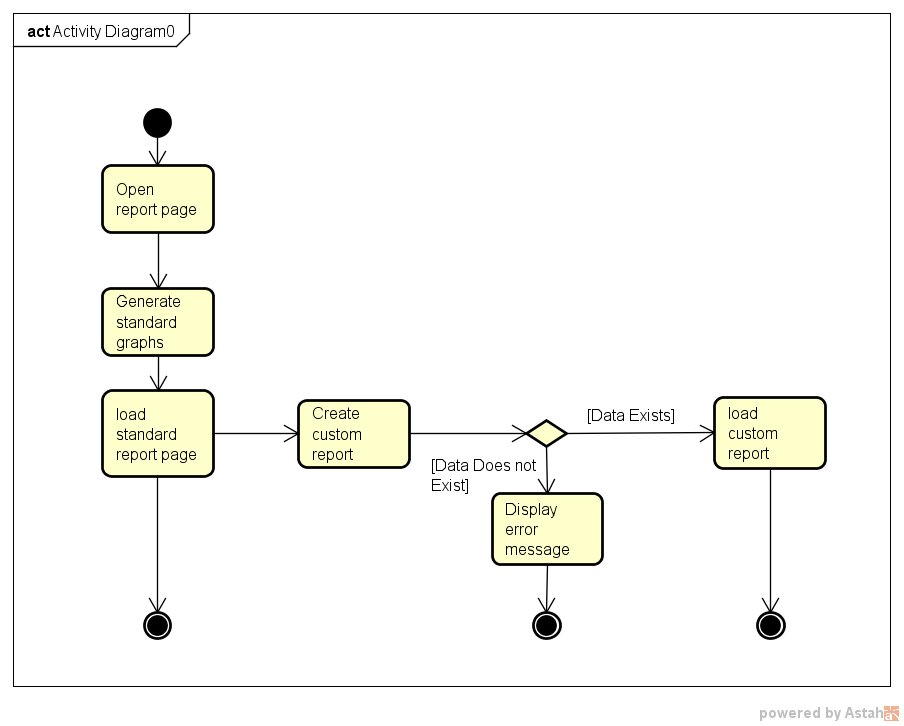
# **Detailed Design**



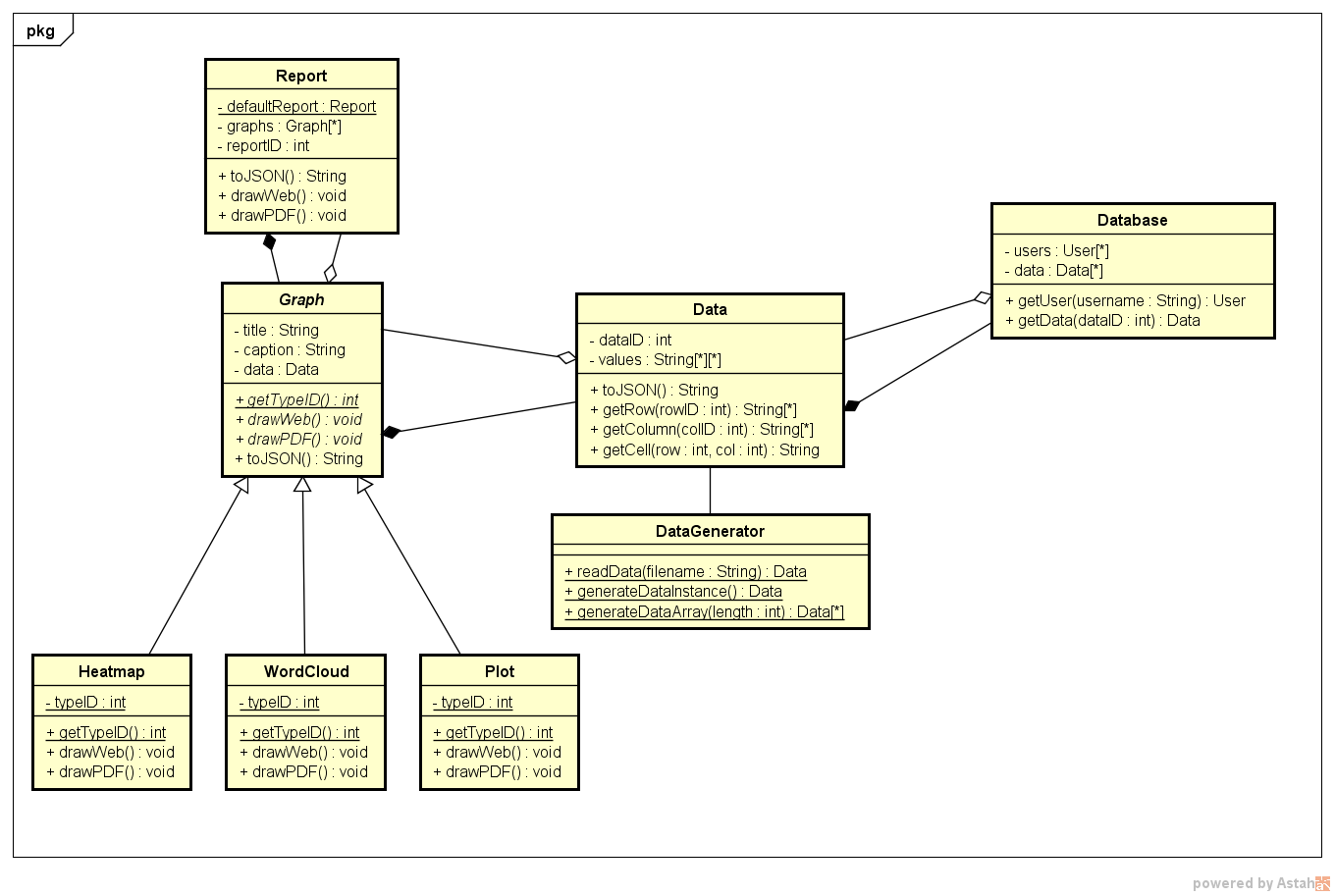
A step by step visual representation of the steps our software takes to generate one graph, when the user requests that graph.



A visual representation of all the features a user has access to when interacting with the webpage.



An activity diagram that shows the potential paths a user may take from first connecting to our webpage till they receive the info they desire.



This object oriented visual representation of our site allows the reader to see all the objects that they may interact with and the variables and methods that each object has associated with it. It also shows how each object interacts with the other objects.

## Description for Component “Report”

A report is comprised of a set of graphs. The report object has the ability to perform bulk actions on all of the graphs contained within.

## Description for Component “Database”

Represents the database that the Report Generation utility will be querying for data. Outputs data on a query from the server.

## Description for Component “Graph”

A graph object. Represents an abstract class that serves as a base for specific graphs. A few examples of extended graph are listed below. A graph object will be created by the user’s browser once they have received data from the server. Most graphs are being constructed with Google’s Google Map API or Google Charts API.

## Description for Component “Data”

Represents the output from the server after making a REST request. The data can be processed by the “DataGenerator” and output as JSON data to be received by the front end. JSON is a data container which allows for easy sending/receiving and processing with a browser. It is also the data structure that Google Charts takes as input.

## Description for Component “DataGenerator”

Represents the server-component which will serve to process and facilitate sending data to the front end. This component will be written as PHP code which will execute on the server when a user requests a report. Upon a report request, the “DataGenerator” will query the “Database” for data and, once received, will process/deliver this to the client’s browser.

# **Data Architecture**

## **Data Analysis**

Accessible non-technical data analysis is the primary focus of this project, and to that end all visible and reported analysis is graphical in nature. Our general data analysis architecture is as follows:

1. A REST request is made to the php server to deliver data to the front end.
2. The server then either generates the data or formulates a request to the backend.
3. The server then packages the data into a defined JSON format.
4. This JSON is returned to the front end, where it is submitted to one of our used APIs to generate a graph.

The Major APIs used in data analysis are:

Google Charts

Google Maps

It should be noted that catering to the Google Charts API had a major influence on our project design.

## **Output Specifications**

REST API and JSON:

Supports the system requirement of having easy to access, well formatted data to be transmitted to the front end for analysis.

* The use of a REST API for the request of JSON formatted data to be transmitted to the client’s browser.
* The PHP class data\_gen/GoogleJsonFormatter.php is intended to aid in easy conversion from either server generated data or database request data to Google Charts API JSON format requirement.

## **Logical Database Model**

The database is never updated with real time data due to the fact that our database is only a snapshot of the Mapping for Change Database.

**Tables in the Database:**

Organization: A table representing Service providers in the Kelowna Region

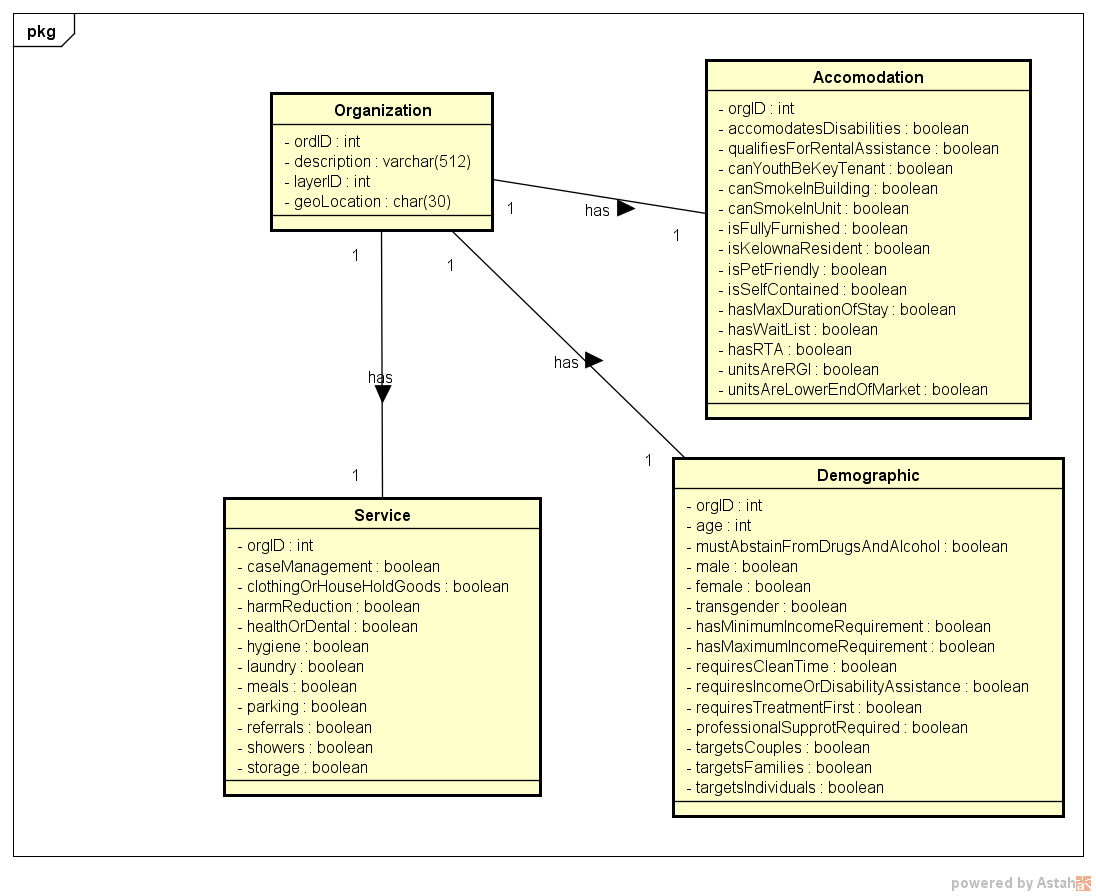
Accommodation: The policies and allowances that of an organization within their buildings.

Service: The services that an organization provides.

Demographic: General demographic information about the demographics an organization servers.

The Organization table has a 1 to 1 relationship with each of the other tables. This is because the other tables are full lists representing the attributes of their category. Since the lists is comprehensive, and each organization is given the same list, each organization only needs one of each entity.

**ER diagram:**



A visual representation of our database showing the tables, the attributes for each table, and how the tables interact with each other.

**SQL Table Field Descriptions:**

|  |
| --- |
| **Organization: A table representing Service providers in the Kelowna Region**    **orgID INTEGER PRIMARY KEY**: An automatically generated ID to uniquely identify the organization in the database.  **description VARCHAR**(512): The organizations self description  **layerID INTEGER**: A number representing what type of services is offered. The meaning of this value is specified by the Mapping for change initiative, and is not within our control to define it.  **geoLocation CHAR**(30): The longitude and latitude of the service |
| **Accommodation: The policies and allowances that of an organization within their buildings.**  **orgID INTEGER PRIMARY KEY**: Represents the organization; see the organization table.  **accommodatesDisabilities BOOLEAN**: Whether or not the organization has special accomodations for the disabled  **qualifiesForRentalAssitance BOOLEAN**: Whether or not the organization has met the appropriate qualifications for governmental rental assistance for individuals.  **canYouthBeKeyTenant BOOLEAN**: Whether or not the organization allows for a person under the age of 18 years of age to be the primary tenant of the residence.  **canSmokeInBulding BOOLEAN**: Whether or not the organization allows an individual to smoke within the building.  **canSmokeInUnit BOOLEAN**: Whether or not the organization allows an individual to smoke in their personal unit/ such units exist in the building  **isFullyFurnished BOOLEAN**: Whether or not the organization’s units are fully furnished.  **isKelownaResident BOOLEAN**: Whether or not the organization requires the individual requesting service to be a kelowna resident.  **isPetFriendly BOOLEAN**: Whether or not the organization allows pets in the unit/ on the premises.  **isSelfContained BOOLEAN**: Whether or not the units that the organization provides contain a kitchen and bathroom. If this field is not applicable, it is false by default.  **hasMaxDurationOfStay BOOLEAN**: Whether or not the organization has an upper bound on the length of time that an individual can receive.  **hasWaitList BOOLEAN**: Whether or not the organization is already its maximum serving capacity  **hasRTA BOOLEAN**: Whether or not the organization requires members receiving the service to sign a Rental Tenancy Agreement.  **onlyServesAboriginal BOOLEAN**: Whether or not the organization is strictly geared towards providing services to Aboriginal individuals.  **unitsAreRGI BOOLEAN**: Whether or not the units an organization are priced proportionally to the income of the individual receiving the service. RGI means Rent Geared to Income.  **unitsAreLowerEndOfMarket BOOLEAN**: Whether or not the units can be considered very cheap (if the individual is required to pay out of pocket for the service).  **FOREIGN KEY** (**orgID**) **REFERENCES** Organization(**orgID**)  **ON UPDATE CASCADE**  **ON DELETE NO ACTION** |
| **Service: The services that an organization provides.**  **orgID INTEGER PRIMARY KEY**: Represents the organization; see the organization table.  **caseManagement BOOLEAN**: The organization provides case management.  **clothingOrHouseHoldGoods BOOLEAN**: The organization offers clothing and general household goods.  **harmReduction BOOLEAN**: The organization provides mechanisms and supports to decrease the impact of harmful activities on an individual that is receiving the service.  **healthOrDental BOOLEAN**: The organization offers health care or dental services.  **hygiene BOOLEAN**: The organization offers general hygiene amenities or supplies.  **laundry BOOLEAN**: The organization offers laundry services.  **meals BOOLEAN**: The organization offers/provides meals at least once a day, or is somehow involved in giving individuals food.  **parking BOOLEAN**: The organization offers places for an individual to park their vehicle if they have one.  **referrals BOOLEAN**: The organization provides referrals to other organizations so that the individual may better receive help.  **showers BOOLEAN**: The organization offers showers.  **storage BOOLEAN**: The organization offers a place for an individual to store belongings.  **FOREIGN KEY** (**orgID**) **REFERENCES** Organization(**orgID**)  **ON UPDATE CASCADE**  **ON DELETE NO ACTION** |
| **Demographic: General demographic information about the demographics an organization servers.**  **orgID INTEGER PRIMARY KEY**: Represents the organization; see the organization table.  **age INTEGER**: The lowest age that an individual is eligable to receive the service. If this field was a range or did not exist, then it is -2^32.  **mustAbstainFromDrugsAndAlcohol BOOLEAN**: Whether or not an individual must not use an legal or illegal drugs except for medication.  **male BOOLEAN**: The individual can be male.  **female BOOLEAN**: The individual can be female.  **transgender BOOLEAN**: The individual can be transgendered.  **hasMinimumIncomeRequirement BOOLEAN**: The individual must have at least a certain level of income to receive the service.  **hasMaximumIncomeRequirement BOOLEAN**: The individual must have an upper bound on the amount of income they can have per month.  **requiresCleanTime BOOLEAN**: Recovering addicts must have been clean for a certain length of time.  **requiresIncomeOrDisabilityAssitance BOOLEAN**: The individuals must be on some sort of income assistance.  **requiresTreatmentFirst BOOLEAN**: The individuals must have first gone through some sort of detox or addictions counseling.  **professionalSupportRequired BOOLEAN**: The individual must be receiving some form of counselling from either a medical professional or a social worker.  **targetsCouples BOOLEAN**: The organization caters to couples.  **targetsFamilies BOOLEAN**: The organization caters to families.  **targetsIndividuals BOOLEAN**: The organization caters to individuals.  **FOREIGN KEY** (**orgID**) **REFERENCES** Organization(**orgID**)  **ON UPDATE CASCADE**  **ON DELETE NO ACTION** |

## **Data Conversion**

Data Converting Classes:

* data\_gen/GoogleJsonFormatter.php is a class intended to allow for each, valid Google Charts formatted JSON to be created from raw data received from the database or generated on the server.

SQL Database Data to JSON:

* Data conversion occurs on data received from the database to valid Google Charts JSON that is then sent to the front end.

SQL to Google Maps API:

* Data conversion occurs between the database and the server when a request is sent to the database.
* The returned data is then formatted into a JSON object appropriate for use with the Google Maps API

# **Interface Requirements**

## **Required Interfaces**

**Internal Interfaces:**

* A RESTful API is used for communication between the server and the client’s browser. This is particularly true for data requests.

**External Interfaces:**

* SQL requests are sent to our MariaDB using PHP Data Objects in conjunction with Prepared Statements
* Google Charts API:
  + <https://developers.google.com/chart/interactive/docs/reference>
* Google Maps API:
  + <https://developers.google.com/maps/documentation/javascript/tutorial>
* D3 API :
  + <https://d3js.org/>
  + Another chart/data displaying library

## **External System Dependencies**

The Google Charts API and the Google Maps API require that Google’s content delivery network (which are serving these components back to the client’s browser) are functioning nominally.

A Javascript compatible browser is required to view our website’s graphs.

The D3 API does not depend on any companies servers, but we do require that is keeps being free for use.

# **User Interface**

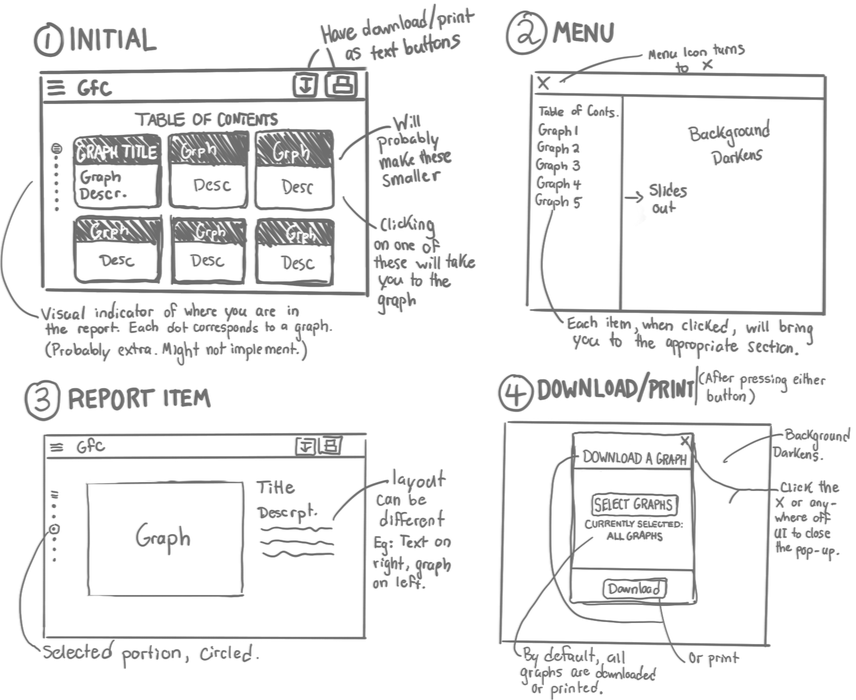
The user should access our website by their preferred web-browser. Currently our website is one page of charts, diagrams and graph, but the client has now requested this changed. These elements are classified by the sort of information they are displaying, and can be jumped to by selecting them in the left sidebar at the top of the page. A user navigates the web page by scrolling or selecting a desired section from the menu.

Google Maps Elements:

* These elements utilize mouse drag events within their container to change where the map is centered.
* Scrolls performed with the cursor (while holding the control or command key within the map element) causes the user to zoom in (scroll down) or zoom out (scroll up).
* A zoom in or out action can also be performed by interacting with the map navigation elements located at the bottom right of the map element. The user may click the ‘+’ button to zoom in or the ‘-’ button to zoom out.

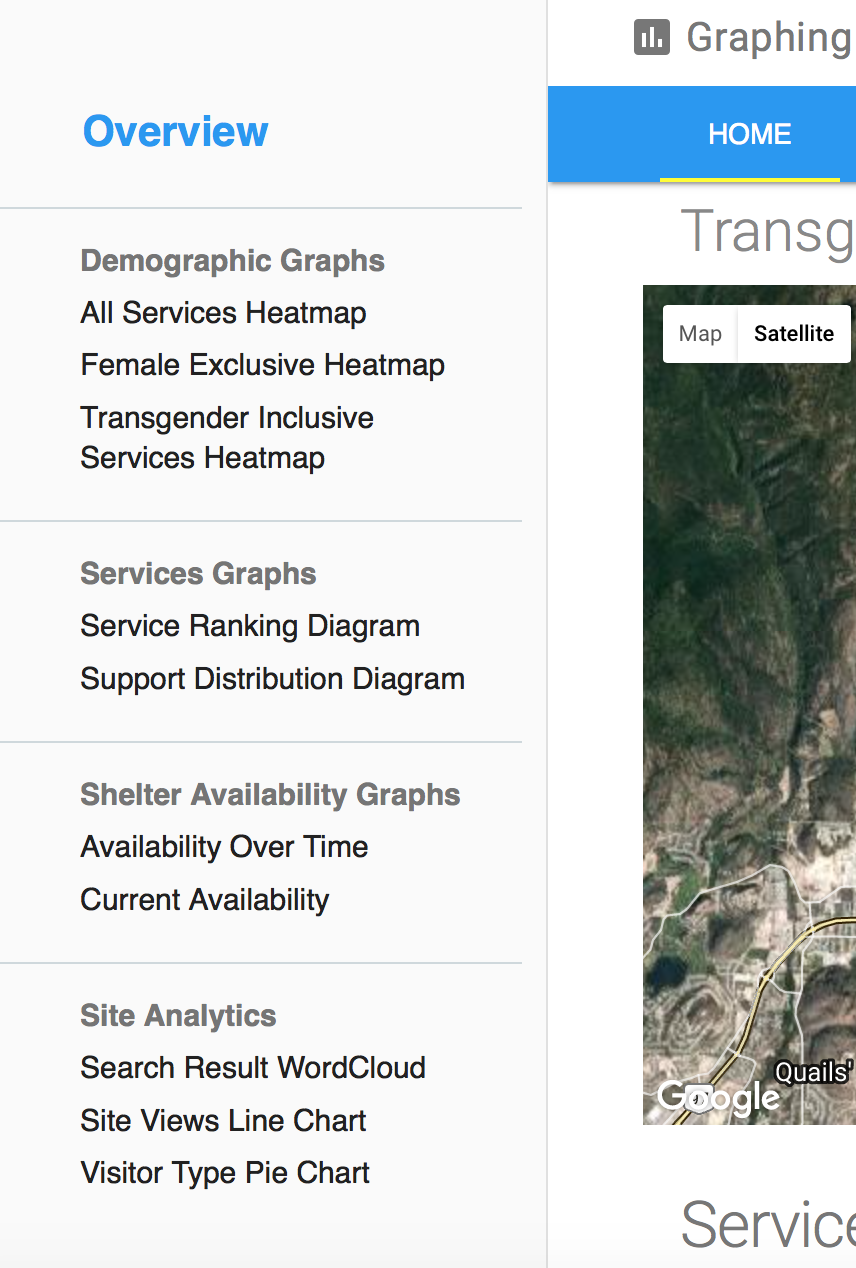
The ‘Download’, ‘Print’, and ‘About’ represent potential selectable element on the top bar of our page. These have been deemed out of scope for the current development stage of the project by the client and serve no content currently.

## UI Design Documents



An illustration of what we imagined our site looking like at the start of our first sprint to give our client a perspective on what we envisioned this software to look like.

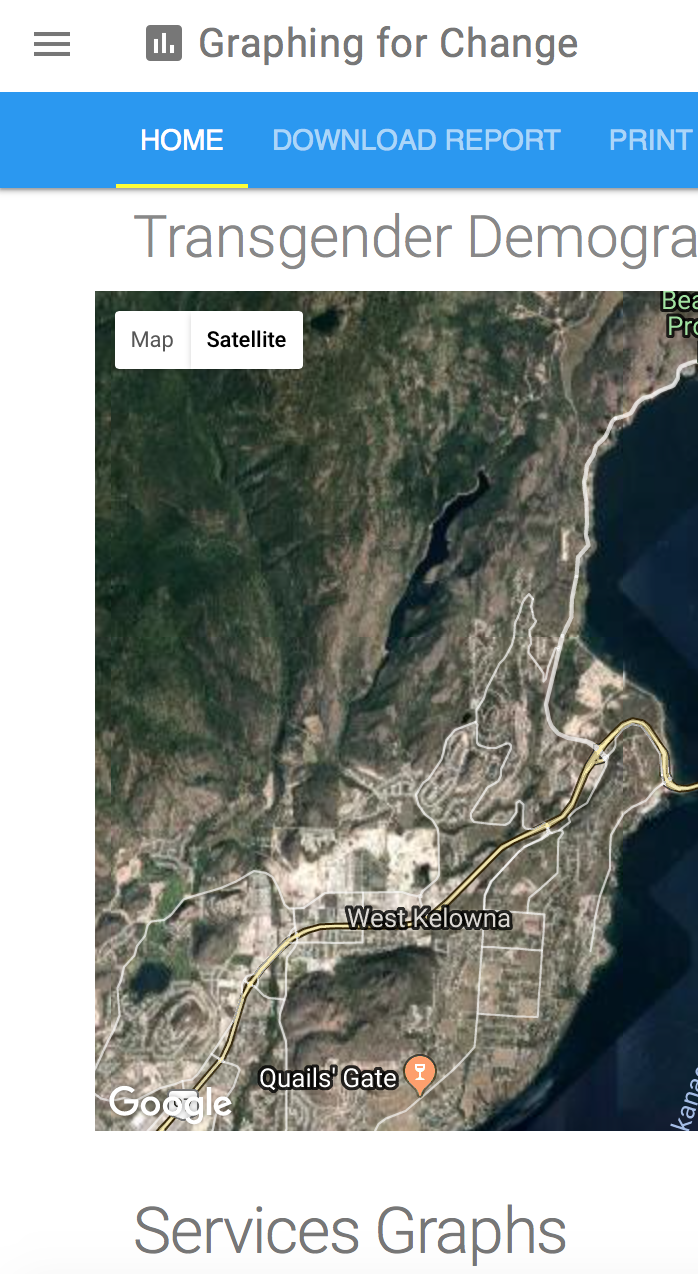
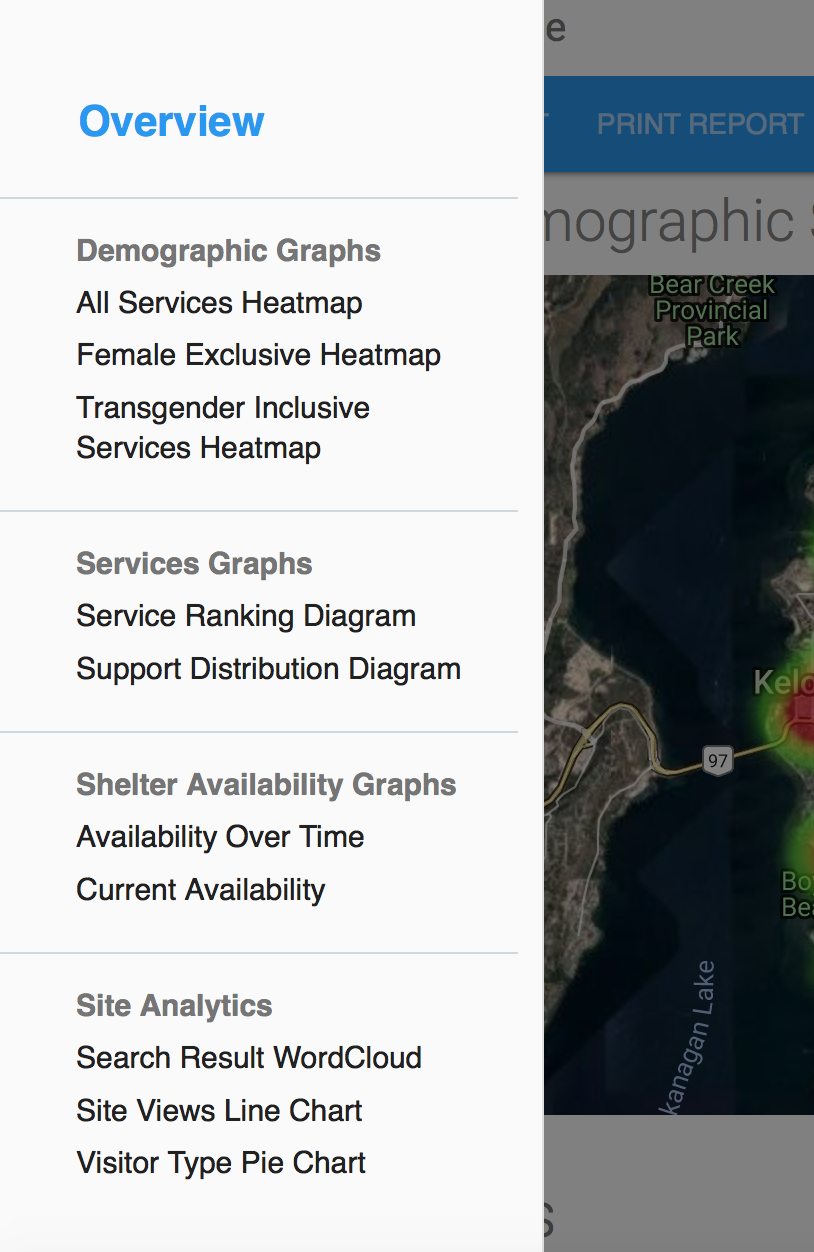
## **Interface Design**



**Navigational Sidebar:**

The left side website features a sidebar (mobile menu button) that enables easy navigation throughout the document. Each graph section title is denoted in a bold (dark grey) color. The individual graphs are listed underneath this element. Graph sections are divided by a light grey line. Each graph section is an unordered list made up of list items. The first list item in the unordered list (the section title) is made bold.

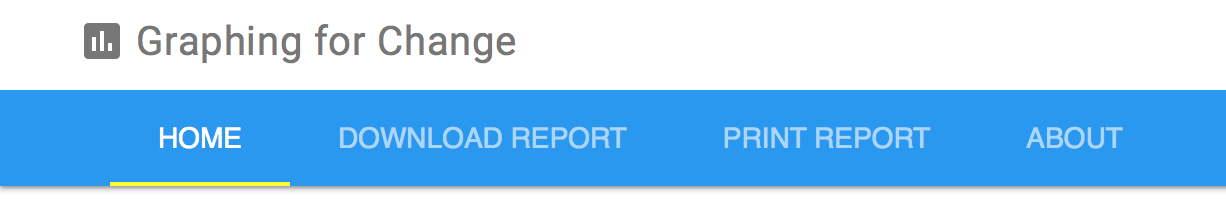
(pictured to the left)

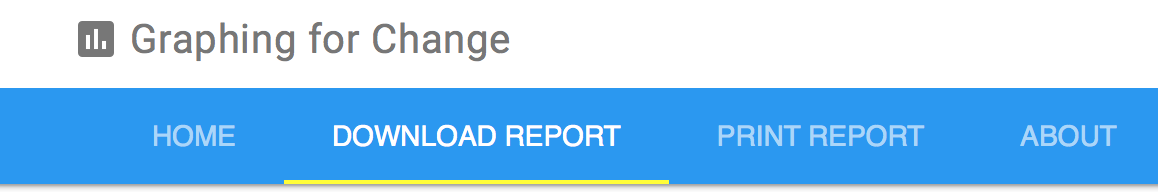


**Mobile Considerations: Navigation Sidebar**

On smaller screens (mobile phones, tablets, etc), the navigational sidebar will take the form of a navigational menu button that will display the sidebar when pressed.

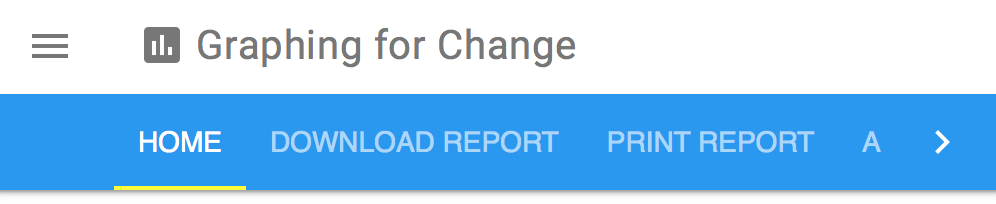
(before and after states are pictured to the left)

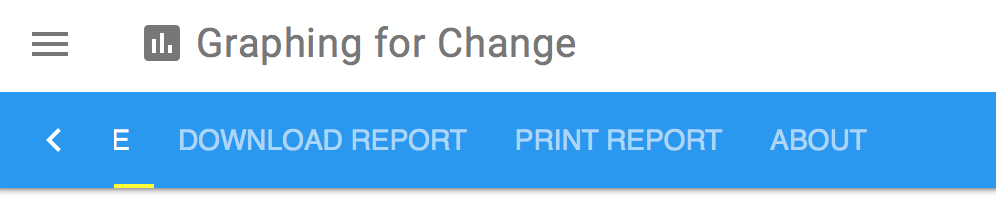




**Navigation Bar:**

The top of the website features a navigation bar which enables quick access to the different pages on the website. The selected page is denoted by a yellow line. An example of differing selection states is pictured above. This action can be triggered with a mouse click or a touch event (Example: a mobile user presses one of the buttons in the navigation bar). An example of the different selection states is picture above. Currently, only the “HOME” button features usable content due to time constraints and client project direction.





**Mobile Considerations: Navigation Bar**

On smaller mobile devices, the navigation bar will scale to provide a menu that users may to scroll to access the various elements. The elements are accessible through a drag motion along the navigation bar (touch devices) or by clicking the arrow icon located on the left or right side (depending on user position in the navigation bar). Denoted above are two different positions, before and after, during a scroll.

## **Functionality**

* All pages should be scrollable when it contains content outside the view of the user.
* Pages should contain a clickable index for navigating down a page.
* All Google Maps displayed should be interactive in the way the google defines.
* The Service Ranking Google Map should expand its cluster icons when sufficiently zoomed in.
* The Service Ranking Google Map should have a control panel that allow a user to filter the icons on the map by the kind of housing it is.

# **Non-Functional Requirements**

* The design of pages should be aesthetically pleasing and minimalist as to not draw attention away from the graphs.
* A web page should not take more than a second to load.
* Graphs should convey the current state of affairs with regards to supportive housing in Kelowna and enable planners to make informed decisions.
* Graphs should be aesthetically pleasing.
* Graphs must display data as it is reflected in the database at time of page load and adapt to changes in the data over time.
* Graphs must display all the relevant data, or display an error if unable to fetch all relevant data.
* All graphs should render within 3 seconds of the page loading.
* The webpage must be easily integratable with the client's current website and function consistently among browsers including Internet Explorer.
* The system must be expandable so more graphs can be easily added.