

**Interactive Statistic Mapping Application**

**Software Design Description (SDD)**

CMSC447\_Team3\_CodePods\_SDD

Version 1.1

May 12, 2017

|  |  |
| --- | --- |
| **Course** | CMSC 447 |
| **Team** | Team 3 - Code Pods |
| **Members** | Benjamin Hazlett  Darrell Laffoon  David Pan  Desiree Mercuree  Ian Moskunas  Kevin Miller |
| **Sponsor (Customer)** | Shawn Squire |

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Description** |
| 03/26/2018 | 1.0 | First Draft Started |
| 5/12/2018 | 1.1 | Finalized |

**Table of Contents**

[**Introduction**](#_9f4gojb3olqe) **4**

[Purpose](#_9knlm25ugs0) 4

[Scope](#_f6scwfr78r6y) 4

[**References and Definitions**](#_hiy022kjwail) **4**

[Table 1 - Document Artifacts](#_e2ittkge897t) 4

[Table 2 - Glossary of Terms](#_l3qvlc6uf41v) 5

[**System Overview**](#_aj1pdb38zxvo) **6**

[Diagram 1 - System Overview](#_rbv2sdz15po) 6

[Table 3 - Application Components](#_nwsbzxy9zdi2) 7

[Table 4 - Application Actors](#_7g80tbsrr37n) 7

[**Requirements Traceability**](#_pscmptpldbrt) **7**

[Table 5 - Requirement Section References](#_hmwyj8yb7x0t) 8

[**Design Overview**](#_7otud28i40h6) **9**

[System Architecture](#_fht696qendb3) 9

[Diagram 2 -System Architecture](#_k4s6hxwxqhoj) 9

[System Operation](#_pkw0xenapdaa) 10

[Diagram 3 -System Operation](#_j093vv6vgrkz) 10

[Constraints and Assumptions](#_kc7chgp84zvz) 10

[**System Components**](#_ke8j0c20tc7) **10**

[Application Web Client](#_20ltn5bpmo7j) 10

[Diagram 3 Web Client](#_2ogcfrigixen) 11

[Statistics Server](#_43sztg8yx385) 11

[Diagram 4 -Server API Integration](#_tflvnf6fbabo) 12

[**Appendix**](#_to02ke4nky9u) **12**

[Table 9 - Potential Statistical Data Sources](#_nqouvtvutjuq) 12

[Table 10 - Google Maps API](#_ywt85srp4jgs) 13

# Introduction

## Purpose

The purpose of this Software Design Description(SDD) is to define the design description used to develop the Interactive Statistic Mapping Application.

## Scope

This SDD describes the architectural and design decisions the plan that will be used by the team to design the software and build the Application.

The SDD contains the following information:

* Terms and Definitions
* Systems Overview
* Requirements Traceability
* Design Overview
* Description of the System Components
* Data Model

# References and Definitions

### *Table 1 - Document Artifacts*

|  |  |  |  |
| --- | --- | --- | --- |
| **Document** | **Description** | **Version** | **Date** |
| CMSC447\_Team3\_CodePods\_SRS | Software Requirements Specification | 1.2 | 03/26/2018 |
| CMSC447\_Team3\_CodePods\_SDP | Software Development Plan | 1.1 | 03/26/2018 |
| CMSC447\_Team3\_CodePods\_SDD | Software Design Description | 1.1 | 05/12/2018 |
| CMSC447\_Team3\_CodePods\_STD | Software Test Description | 1.1 | 05/14/2018 |
| CMSC447\_Team3\_CodePods\_STR | Software Test Report | 1.3 | 05/15/2018 |
| CMSC447\_Team3\_CodePods\_SUM | Software Users Manual | 1.2 | 05/14/2018 |

### 

### *Table 2 - Glossary of Terms*

|  |  |
| --- | --- |
| **Term** | **Meaning** |
| **The Application** | The targeted software solution - the Interactive Statistic Mapping Application |
| **The System** | The System that encompasses the application. The System and The Application could be used interchangeably in most cases |
| **The Customer** | The Customer that sponsored the project/software Application - Shawn Squire |
| **The Team** | The team of students who will build the Application |
| **SDLC** | Software Development Life Cycle - describes the steps and phases used to design, build and test the application |
| **Agile** | A set of principles that are used to define a iterative and incremental SDLC that is used to build the Application in iterations and allows for Customer feedback to guide the development of the Application |
| **Actor** | A person, external system, or other ‘actor’ who interacts with The Application or System |
| **Component** | A sub-system or part of the Application or System |
| **Map** | The selected, displayed Map, including boundaries. For example, it could be the State of Maryland, or Baltimore, or a neighborhood |
| **Map Data** | Detailed data about the selected Map |
| **Statistics Option** | One of the statistic choices such as crime, school ranking, etc. |
| **Proof of Concept** | A version of the Application that is used to test and prove aspects of the design |
| **Use Case** | Details behavioral requirements |
| **System Model** | Diagram that depicts system components and communication context |
| **Class Model** | Documents the Data entities of the System |
| **Sequence Model** | Documents the sequence of events between actors and components for particular use cases and application events |
| **State Model** | Documents state transitions for the system during particular uses casee and application events. |
| **Test Driven Development (TDD)** | A development methodology where unit tests are created before components are built. |
| **Software Repository** | A distributed data store that holds and tracks versions of the Application’s source code, documentation and other artifacts. |
| **Unit Test** | A test done on low level components as they are built |
| **Integration Test** | A test that tests the ability of system components to work together properly |
| **Regression Test** | A Test that checks that previously implemented features are still working after changes are deployed |

# 

# System Overview

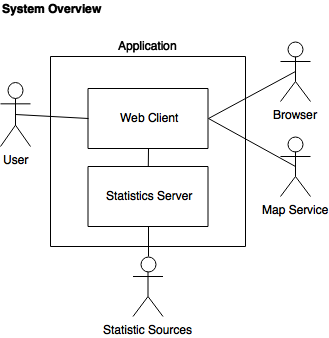
This section provides a high level System Overview for the Application. The Application’s purpose is to help a user answer a universal question:

"Where do I want to live, work, or retire?”

The Application will allow a user to select a Map and visually overlay important Statistics such as crime, income, school ranking, average commute, etc. The overlay will help the user identify areas on the Map that are more or less desirable. An expected, common use case would be a parent, who is looking to move to a safe and desirable place to raise her child, would use the Application to see and compare the crime rate and school rankings in the areas of interest.

*Diagram 1* depicts the key actors and components of the Application.

## *Diagram 1 - System Overview*

[](https://www.draw.io/#G1nqSkvUCma-uFmSX96F2KdXn3Pjgeq0vn)

The Application will consist of two components:

## *Table 3 - Application Components*

|  |  |
| --- | --- |
| **Component** | **Description** |
| **Web Client** | Frontend web application that allow a user to interactively identify a Map, and desired Statistics. It will dynamically update the Map with a visualization of the selected Map and selected Statistics Options |
| **Statistics Sources** | Backend server application that provides aggregated Statistical Data (to the Web Client), such as crime, income, commute, etc., from various trusted Statistic sources. This server encapsulates all data sources and converts them into a common, normalized format |

The Application will have of four primary actors :

## *Table 4 - Application Actors*

|  |  |
| --- | --- |
| **Actor** | **Description** |
| **User** | Will interact with the Web Client to select the Map and Statistics Option she wishes to see visualized |
| **Browser** | Will host the Application and provide input and output. Specifically, the browser will display the Map and a representation of the Statistical Data, etc. |
| **Map Service** | An external map service (Google Maps) that will provide the mapping capabilities and Map Data to the Web Client |
| **Statistic Services** | Publicly accessible Statistical Data sources that will return data for the user selected Map. *See Appendix for list of sources under consideration* |

The User will interact with the Web Client, which will present a Map and Statistics Options. As the User makes changes to the Map and/or chooses a Statistics Option, the Web Client will communicate with the Statistics Server to get updated Statistical Data for the selected Map. The Web Client will then redraw the Map to include Statistical Data based on the User’s selections. The Map Service will provide Map Data. The Statistics Sources will provide Statistics Data.

# 

# Requirements Traceability

For a detailed look at the user requirements of the Application and the corresponding functional and non-functional system requirements can be found in the Software Requirements Specification (SRS) document (CMSC447\_Team3\_CodePods\_SRS).

The SRS details five User Requirements that the Application must meet, and are presented as Use Cases (UC-1 through UC-5).

### *Table 5 - Requirement Section References*

|  |  |  |
| --- | --- | --- |
| **Requirement (Use Case)** | **Description** | **Design Section Reference(s)** |
| UC-1 | Select Map | 5.2, 6.1 |
| UC-2 | Select Statistics Option | 5.2, 6.2 |
| UC-3 | Provide Map Data | 5.1, 5.2, 6.2 |
| UC-4 | Provide Statistical Data | 5.1, 5.2, 6.2 |
| UC-5 | Redraw Map | 5.1, 5.2 |

# 

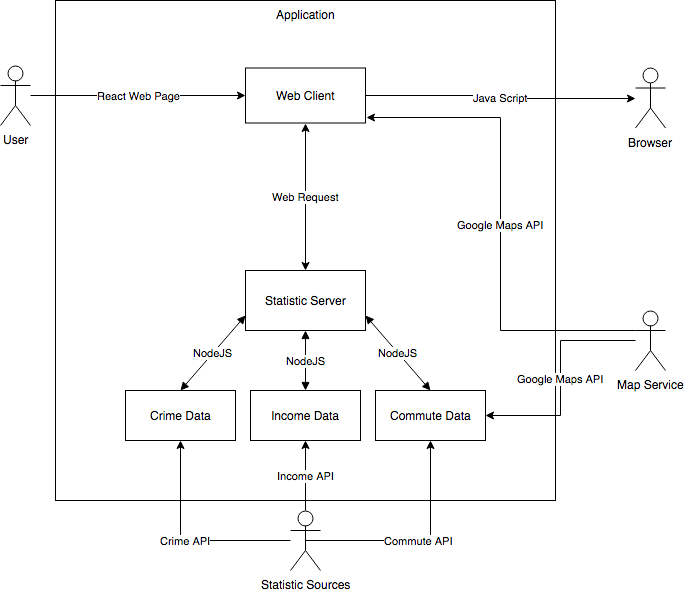
# Design Overview

## System Architecture

*Diagram 2* describes the high-level system architecture. The system is made of the following distinct components.

* **Web Client -** The frontend interface that displays the map and statistical information
* **Statistic Server -** The backend server which will make http requests of the APIs to get **i**nformation that will be sent to the Web Client
* **Crime Data -** Information pulled from the crime API
* **Income Data -** Information pulled from the school API
* **Commute Data -** Information about average commute times from Google Maps API

### *Diagram 2 -System Architecture*

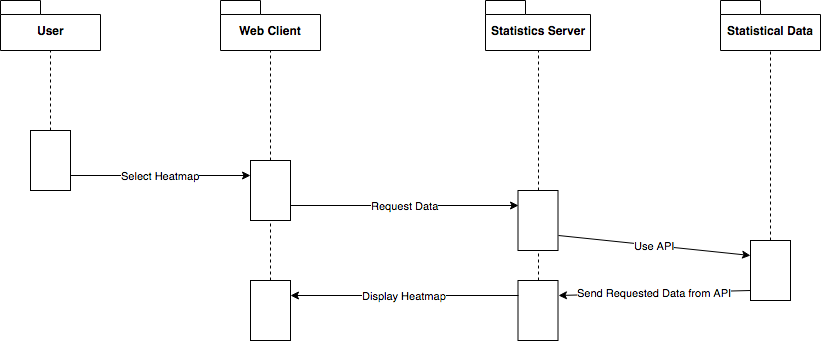
[](https://www.draw.io/#G1vRbpC5atAIIxsa6vK6TMQEf1EfTYXOis)

## System Operation

*Diagram 3* details the basic flow of data and interaction between the User and the system components. As the User makes selects the map area and statistics (select Heatmap), the Web Client requests updated statistics data from the backend Statistics Server, which pulls the data from the various external data suppliers and the data is passed back to the Web Client and an updated map is displayed with a

heatmap visualization of the data.

### *Diagram 3 -System Operation*

[](https://www.draw.io/#G1Xrc-eYpwlBPtoq0ZRCZs18Y1WKMWFFuH)

## Constraints and Assumptions

The Application will access two sets of external services:

1. Google Map Service
2. External Statistical Data Sources

Google Map Service provides a robust set of APIs and will be used to get Map Data for a selected Map and will also be used to update the Map visually with the statistical heatmap. See Appendix for list of Statistical Sources.

The Application will be concerned with only Maryland data, meaning all statistical information will be pertaining to Maryland. The Application might make data available outside of the Maryland area if it is available.

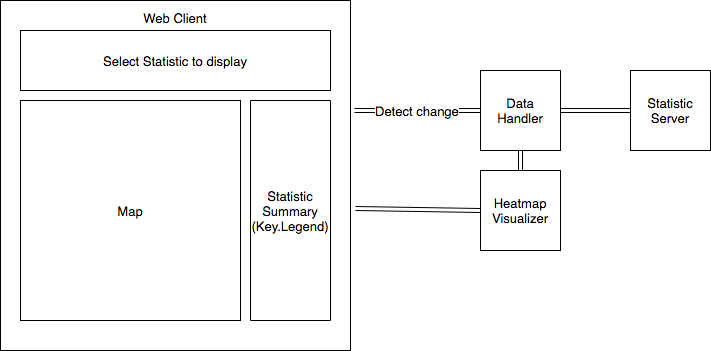
# 

# System Components

## Application Web Client

The Web Client (*Diagram 3*) will be present a map (using Google Maps) and the options for statistics that are available to view. As the user interacts, a data handler will detect the changes and will package up the options selected and pass them to the API Gateway, resultant statistical data will then be passed to a data handler which will pass the data to a visualizer that will render a heatmap layer, representing the data, on top of the current map. The data handler will update the summary area of the client with important summary information for the current statistic and map area.

### *Diagram 3 Web Client*

[](https://www.draw.io/#G1f_MvqvyguVeZwlCct07x7gKqD0SSMxD9)

## Statistics Server

The back end server will integrate with external data sources via API if available. The details of the API integration will be hidden from the front end and a consistent interface will be made available via the servers API Gateway (see Diagram 4).

Each Statistic that is presented follow a consistent pattern. The border area of the map will be presented as the query parameter to pull statistics data from for that area. And the returned result will be a JSON array that contains the statistic data points found for that area.

Here is an example of the:

Query = {

metric=commuteTime,

topRight: { latitude: '39, longitude: '-75' },

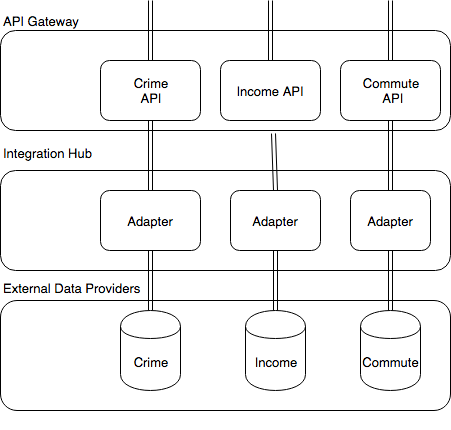
bottomLeft: { latitude: '39, longitude: '-76 } }

Result = {

[{"metricValue":"26.4","loc":{"lat":"39.5117","lng":"-76.2001"}},

{"metricValue":"13.4","loc":{"lat":"39.4366","lng":"-76.1627"}]}

### *Diagram 4 -Server API Integration*

[](https://www.draw.io/#G1ix4t5FF3Ai6MNTJgHwMA38ECg25UaDNP)

# Appendix

### *Table 9 - Potential Statistical Data Sources*

|  |  |
| --- | --- |
| **Census.gov** | <https://www.census.gov/data/developers/data-sets.html>  <https://www.census.gov/data/developers/guidance/api-user-guide.html> |
| **Data.gov** | <http://docs.ckan.org/en/latest/api/index.html>  <https://catalog.data.gov/dataset?q=-aapi+api+OR++res_format%3Aapi#topic=developers_navigation> |
| **Data.gov for Maryland** | <https://catalog.data.gov/dataset?sort=views_recent+desc&q=-aapi+api+OR++res_format%3Aapi&publisher=data.maryland.gov&_publisher_limit=0> |
| **Maryland Transportation** | <https://data.maryland.gov/api/views/ief7-i74z/rows.json?accessType=DOWNLOAD> |
| **USA.gov** | https://www.usa.gov/developer?source=busa#item-211492 |
| **Walkscore** | https://www.walkscore.com/professional/api.php |
| **Zillow.com** | https://www.programmableweb.com/api/zillow |

### *Table 10 - Google Maps API*

|  |  |
| --- | --- |
| **Google Maps API** | <https://developers.google.com/maps/get-started/> |
| **Visualizing data example** | <https://developers.google.com/maps/documentation/javascript/earthquakes> |
| **Heatmaps example** | <https://developers.google.com/maps/documentation/javascript/earthquakes#heatmaps> |