Software Design Specification

for

HealthQ

Version <1.0>

Prepared by Benny Villegas, Eddie Matos,
Stuart Tresslar, Zikomo Bullock, Brian Deguzis, Austin Harris

Team 2

SWE 4724 Software Engineering Project
Spring Semester, 2018
Kennesaw State University
Department of Software Engineering and Game Design

04/29/2018

TABLE OF CONTENTS

Revision History

Name	Date	Reason For Changes	Version
Brian Deguzis	2/27/2018	Document Created	0.1
Austin Harris	3/11/2018	Revision to Sections 1 and 2	0.2
Brian Deguzis	3/14/2018	Section 6: Human Interface Design	0.3
Stuart Tresslar and Zikomo Bullock	4/05/2018	Finished Requirements matrix and formatting	0.4
Benny Villegas and Austin Harris	4/20/2018	Components and terms	0.5

1. INTRODUCTION

1.1 Purpose

This SDS document is designed to track the necessary information required to effectively define the system design and architecture to give the development team a reference and guide for the architecture of the ongoing system development. The intended audience is the project owner and the project/development team. There are areas of this document that may be shared with potential clients, user, or other stakeholders whose input/approval is needed.

1.2 Scope

The purpose of this product is to be able to create software that is able to analyze data and draw conclusions based on collected data. The software will be able to integrate with devices that use Google Assistant such as Google Home to be able to be used by the general public.

1.3 Overview

This document is the software design document for HealthQ. This document is divided into seven main sections. The introduction, system overview, system architecture, data design, component design, human interface design, and requirements matrix.

1.4 Reference Material

"Actions on Google," Google. [Online]. Available: https://developers.google.com/actions/. [Accessed: 09-Feb-2018].
U. S. C. Bureau, "Developers," Census.gov. [Online]. Available: https://www.census.gov/developers/. [Accessed: 09-Feb-2018].
U. S. C. Bureau, "Data Tools and Apps," Tools and Apps. [Online]. Available: https://www.census.gov/data/data-tools.html. [Accessed: 09-Feb-2018].
J. Dillard, "5 Most Important Methods For Statistical Data Analysis," Operations Consulting, Decision Analysis And Process Improvement. [Online]. Available: https://www.bigskyassociates.com/blog/bid/356764/5-Most-Important-Methods-For-Statistical-Data-Analysis. [Accessed: 09-Feb-2018].
"Google Genomics - Store, process, explore and share | Google Cloud Platform," Google. [Online]. Available: https://cloud.google.com/genomics/. [Accessed: 09-Feb-2018].

2. System Overview

HealthQ is an application that is run through Google Assistant. It utilizes that voice activation capabilities of Google devices to allow the user to communicate it. The application connects to a database using a SODA API, and from there it can retrieve information and perform basic calculations based on the retrieved data.

3. System Architecture

3.1 Architectural Design

Dialog Flow gives HealthQ a pleasant user interface for building conversation flows (in some cases no code was needed), and also incorporates some AI aspects to help figure out the user's intent. Actions SDK gives the product "bare-bone" access to user input, a backend is provided separately which will parse the received input and generate the appropriate responses. The product uses DialogFlow, as it can handle some of the flows (e.g. asking the user a yes or no question, and understanding the user response), and would also allow for a quickly developed prototype. DialogFlow's built-in capabilities proved very useful. For instance, if a user didn't know how to translate the sentence they were given, they can say, "I don't know" to get the answer and skip to the next one. In terms of the SDK components, Google offers two bridges: the Google Assistant Library, and the Google Assistant Service. The product uses Google Assistant Service to carry out its methods. The Google Assistant Service is the best option for flexibility and broad platform support (hardware device, browser, phone, etc). It exposes a low level API which directly manipulates the audio bytes of an Assistant request and response. Bindings for this API can be generated for languages like Node.js and other JavaScript libraries.

3.2 Decomposition Description

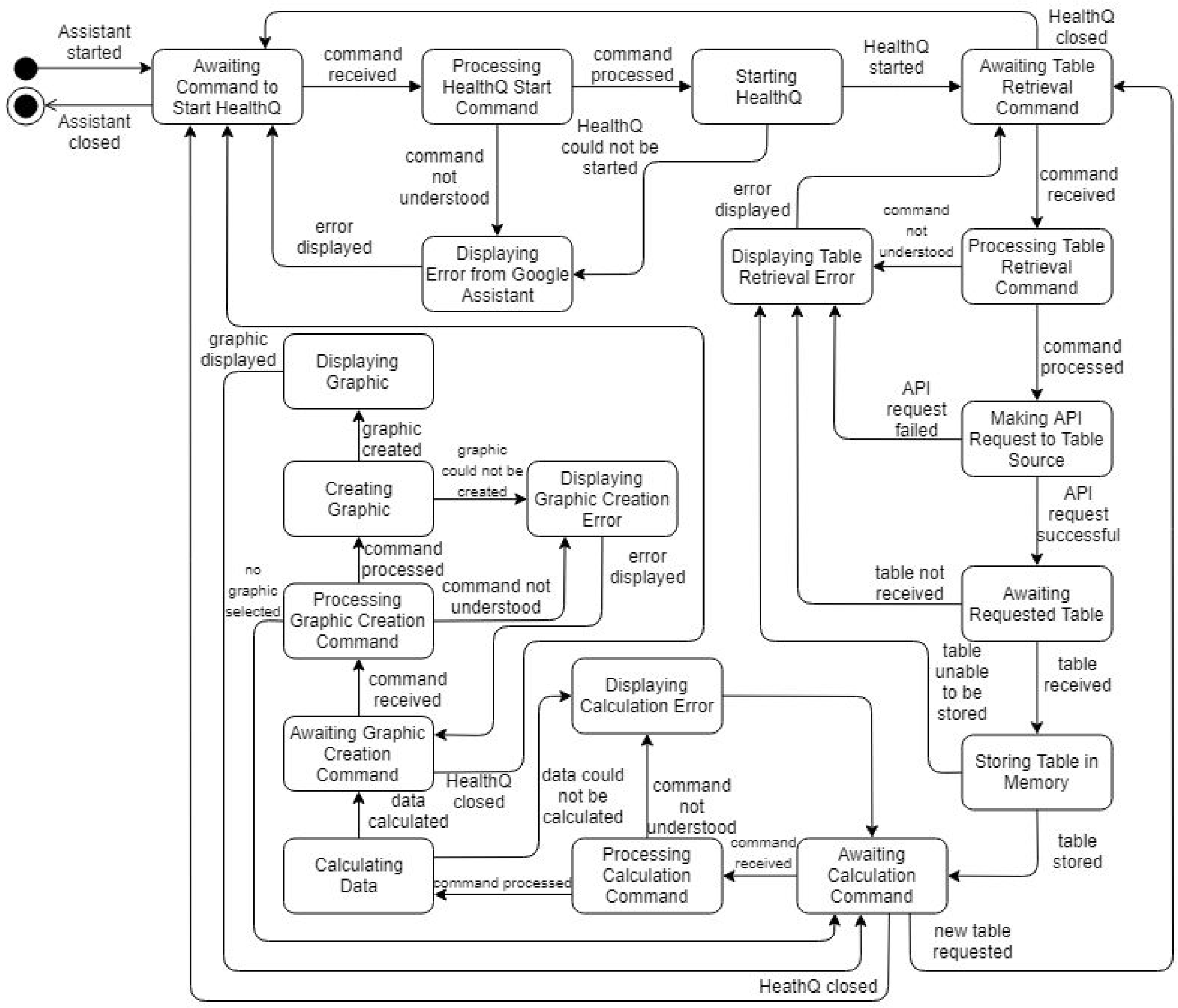
HealthQ uses JavaScript programming in correspondence with the skills of the development team. For the backend, the product uses Node.js and Google's FireBase, a decision that was also backed by the fact there is an official Node SDK for developing Actions on Google using DialogFlow (the actions-on-google package). Google Cloud Platform is a bundle of tools for mobile apps. Among all the other supplement tools, HealthQ relies heavily on Google's FireBase for data storing and parsing parameters. We used Google scripts to define methods when taking in data from the CDC's API. We used Google's Scripts hosting to make sure it reads data in correctly into Google Mini for audio input and output.

3.3 Design Rationale

Google provides design documentation for Voice UI and simulators which allows developers to do a lot of research and experimentation before building. Google offers the most flexibility, however, with great flexibility comes great documentation. Google's tool set documentation surrounding its voice assistant is still limited. We chose Google's data methods because it has a strong AI engine that works in correspondence with Google's knowledge graph, 20 years of search and user data, and a great front-end platform with DialogFlow, which it recently acquired and integrated.

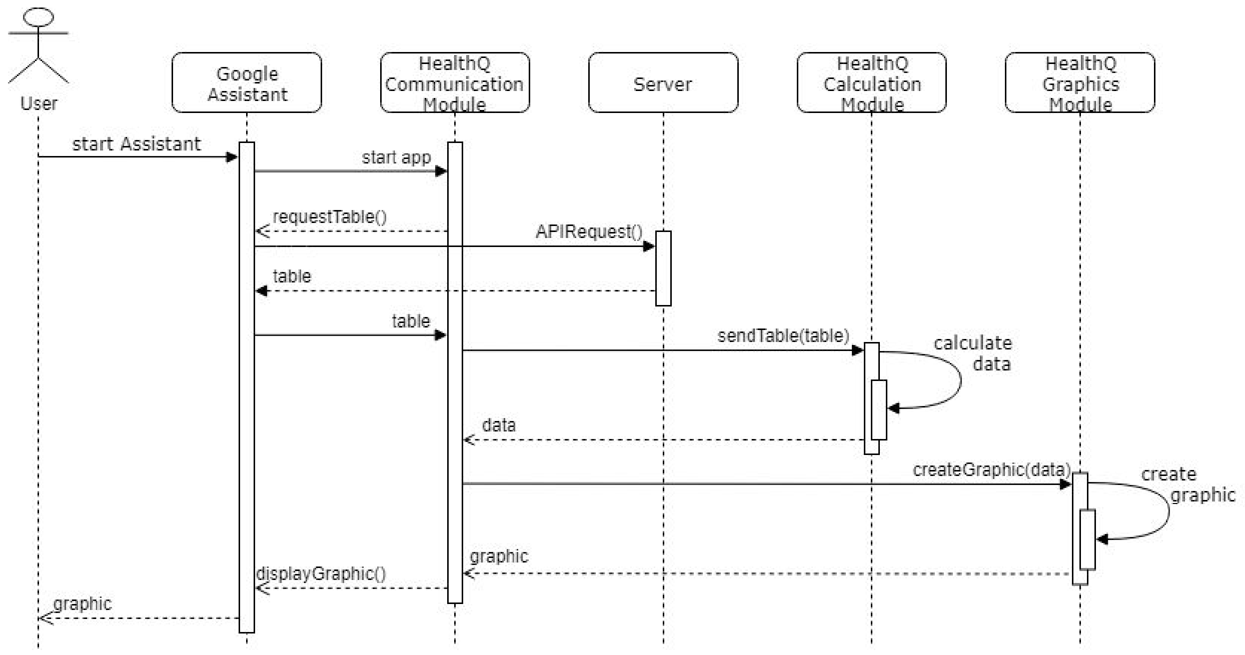
4. Data Design

The data was retrieved using the CDC's native API called SODA, which is an API developed by Socrata. Its endpoints support a number of different response formats that can be specified either via response type extensions or whichever HTTP sources are referenced in the headers. The simplest way to specify the response format is by appending a response type extension to the URL. This allows the developer to set the response format without requiring the ability to set headers in their HTTP client. The HTTP headers allow applications such as HealthQ to automatically negotiate content types with a web service. With SODA, this also means you can request content types using certain HTTP "Accept" headers without needing to provide an response type extension. However, when using Google's SDK, it requires the data to be identified. Google Assistant only accepts JSON, CSV, and XML as data. Below is a state chart to show the "ins and outs" of how the data and requests gets handled and output for the user.



5. Component Design

In this section, we take a closer look at what each component does to give a more clear understanding. Refer to the following document to understand how the components for HealthQ interact with one another.



6. Human Interface Design

Due to the nature of HealthQ, the user will not necessarily have a visual interface. The main method of interaction will be through Google Home devices, such as the Google Mini, where the user will be using their voice as the main method for communication with HealthQ. Other possible interfaces of HealthQ are the Google Assistant applications that run on Android devices, as well as the DialogFlow console.

7. Requirements Matrix

System Component	Relevant Requirements
Process Voice Command	INT.1.1– INT.1.3

	OPR.1.1, OPR.1.2 DAT.1.1 – DAT.1.3
Calculate Statistics	INT.1.2, INT.1.3 OPR.1.5 – OPR.1.7
Retrieve New Data Table	INT.1.2, INT.1.3 OPR.1.3, OPR.1.4