Functional Requirements Document
Care Compare SK

Version	Description of Change	Author	Date
1.0	Start of functional requirements doc	Kegan & Daris	Oct. 31st, 2019
1.1	Added material to Methodology heading	Kegan & Daris	Jan. 7th, 2020
1.2	Added diagrams and requirements information	Kegan & Daris	Mar. 3rd, 2020
1.3	Removed unnecessary information, formatting, & worked on “Other Requirements” section	Kegan & Daris	Mar. 24th, 2020
1.4	Completed ‘Functional Requirements’ section and did some formatting	Kegan & Daris	Mar. 25th, 2020
1.5	Added reference link	Kegan & Daris	Mar. 30th, 2020
1.6	Updated Application Data Diagram, finished “Reliability” & “Recoverability” sections	Kegan & Daris	April 4th, 2020

CONTENTS

1 INTRODUCTION

Project Vision: We want to make finding the best insurance smooth and easy for all types of people. We want to create an application that people of all different backgrounds and experiences actually want to use because it makes their life easier.

1.1 Purpose

The purpose of this FRD is to organize and update the status of our capstone project over the next year. We created this project because we saw a need within society to provide a fast and efficient solution for finding great health care and this FRD helps bring the project to life and track its progress.

1.2 Scope

This document will record every part of our capstone starting with requirements gathering, prototyping, diagrams and modeling, code, algorithms, testing, and anything else needed as the project proceeds. We are keeping the scope of the Capstone as small as we can simply because it is such a big topic and can become out of control easily. We are trying to only include a few insurance providers from Saskatchewan and a few key personas instead of focusing on everyone as a whole. This document will be updated regularly to make sure goals are being completed and that the project and document serve the purpose they set out to serve.

1.3 Background

Our Capstone group consists of two people Kegan Lavoy & Daris Lychuk. We are working collaboratively on all portions of the project and are therefore responsible for the success of the project as one unit. We will both be adding to this document to keep us on track and organized, provide documentation for when tasks were completed, and give us a visual look at our Capstone at a high level.

1.4 References

https://www.thepolyglotdeveloper.com/2018/04/simple-user-login-vuejs-web-application/

https://www.policygenius.com/?fbclid=IwAR3u-OWY6MjbBXZi3Kf3_UNCH_z62TICDoBSemJ3GrBFbekl69dJpJxy1CA

https://www.canva.com/design/DADxkqME88I/dYV3DsKiU9dKoWcOuoFgpQ/edit?category=tACZChe47fQ#

https://www.webdesigndev.com/clean-modern-forum-templates/

https://www.webdesigndev.com/wp-content/uploads/2014/03/001507-Forum-Plus-Responsive-Drupal-Forum-Theme-RTL-Preview-ThemeForest.jpg

http://whatismyscreenresolution.net/

1.5 Assumptions and Constraints

1.5.1 Assumptions

- We are assuming we can get in touch with at least 2 insurance providers so that we will have real data to apply to our project.
- We are assuming that our users will try to access our site on a desktop/laptop instead of mobile device, therefore, will be designing Care Compare SK for optimal use on these devices.

1.5.2 Constraints

- Time to produce the product
- Tech stack; need front end and back end that work together, database connection.
- No current API's for Canadian insurance companies
- Insurance companies are private entities, not forthcoming with giving up their data easily.

1.6 Document Overview

Our document introduces the scope of the project and document discussed. It will then dive into the methodologies used to create the application. After the methodologies are discussed, it will report on all functional requirements the system has. Lastly, it will discuss all non-functional requirements the system must adhere to.

2 METHODOLOGY

Agile Architecture

We used an Agile architecture as a baseline methodology for our project. We did our work in 1 or 2 week sprints depending on the tasks being completed at that time. We chose this architecture because it allows for constant change and adjustments throughout a project which we find important as change will always be present.

User story mapping

We created epics and scenarios of the different tasks and functionalities our users would like to have when looking for insurance. This allowed us to visualize and keep track of what we need to accomplish as well as gather requirements for our application.

Personas

We created different categories of people to try and narrow down tasks that certain groups are more likely to do than other groups. This is also a type of requirements gathering as we are trying to get into the heads of different types of people to see what they are looking for within our application.

Kanban board

We used a Kanban board to organize and visualize our tasks that need to be done, are currently being worked on, as well as our completed tasks. This scheduling style works well within the Agile architecture because it can be constantly modified within the flow of the project.

Low fidelity & high fidelity prototyping

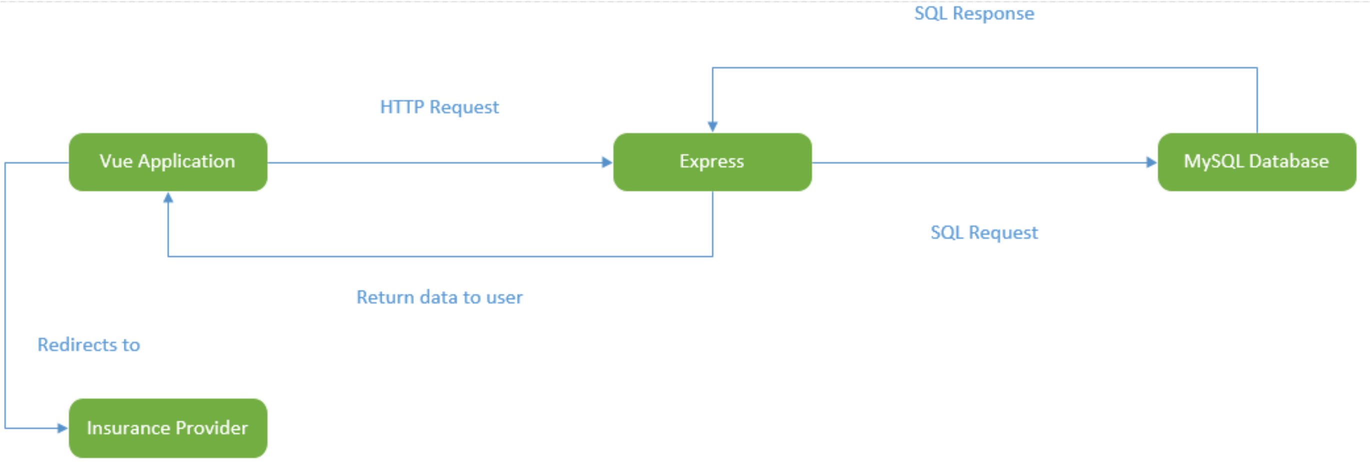
We used Adobe XD software to create low and high fidelity prototypes of our application. Adobe XD projects can be quickly and easily modified and changed which fits well with the Agile methodology.

3 FUNCTIONAL REQUIREMENTS

3.1 Context/Communication

Our Vue Application (user interface) sends an HTTP request to our Express App, which then sends an SQL request to our MySQL database. Our database will then return the appropriate SQL response to our Express App. The Express App then returns the data to the user. Our Vue Application also allows the user to be redirected to the selected insurance provider's site.

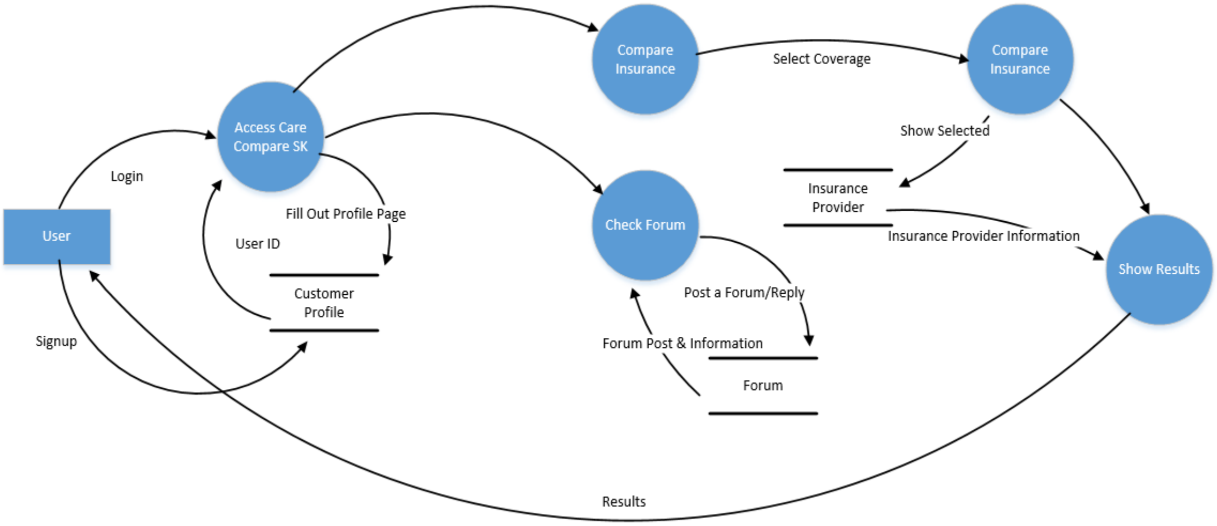
Exhibit 1 - Generic Context/Communication Diagram



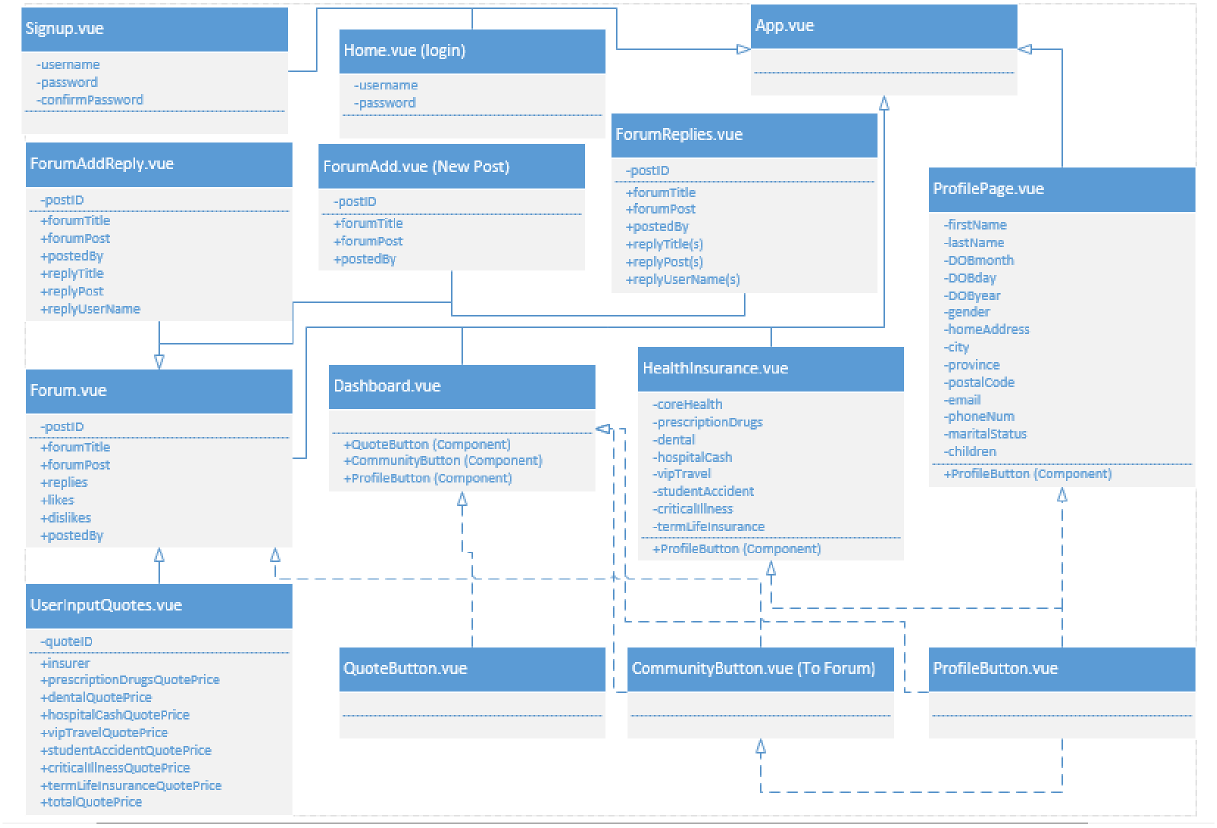
3.2 User Requirements

1. All users are expected to be accessing *Care Compare SK* through either a desktop or laptop and not a smart phone, tablet, or other handheld device.
2. All users must provide a username and password in order to use *Care Compare SK*
3. All users have the option of providing additional information, such as, their full name, age, etc. This is not a requirement to use our website for the forum, however, the additional information is needed to use *Care Compare SK's* comparison tool.
4. All users have the same security level whether full profile information is provided, or partial (username and password only).

3.3 Data Flow Diagrams



3.4 Logical Data Model



3.5 Functional Requirements

3.5.1 Functional Requirements Group 1

Exhibit 2 - Requirements Group 1

Section/ Requirement ID	Requirement Definition
FR1.0	The system shall allow users to access the forum
FR1.1	The system shall allow users to post a new forum topic
FR1.2	The system shall allow users to reply to any other forum post
FR1.3	The system shall allow users to “Like” and “Dislike” posts by users
FR1.4	The system shall allow users to share providers quotes

3.5.2 Functional Requirements Group 2

Exhibit 3 - Requirements Group 2

Section/ Requirement ID	Requirement Definition
FR2.0	The system shall allow users to select which insurance coverage they want to compare
FR2.1	The system shall allow users to see all available insurance provider options
FR2.2	The system shall allow users to see all estimated quotes in our database
FR2.3	The system shall allow users to see average estimated quotes from the community

4 OTHER REQUIREMENTS

4.1 Interface Requirements

4.1.1 Hardware Interfaces

Care Compare SK does not have any hardware interfaces to concern itself with.

4.1.2 Software Interfaces

Blue Cross – When selecting external link

Sunlife – When selecting external link

CAA – When selecting external link

Sure Health – When selecting external link

4.1.3 Communications Interfaces

Care Compare SK is built on a local machine that is accessed via a port that was opened. The web application is completely local, other than when redirecting users to different insurance providers. This will direct users to the insurance provider's own web application to find additional information about the specific insurance company.

4.2 Hardware/Software Requirements

All desktop/laptops should be able to use ANY web browser (*Google Chrome, Firefox, etc.)* to use *Care Compare SK*. Mobile devices, such as phones and tablets, may not work well with *Care Compare SK* styling. *Care Compare SK* should be able to run on any hardware, due to it being a lightweight, non-performance heavy web application.

4.3 Operational Requirements

4.3.1 Security and Privacy

A. Consequences of the following breaches of security in the subject application:

- Loss or corruption of data may lead to users needing to make a new account. It may also lead to losing valuable information on health insurance providers.
- Disclosure of secrets or sensitive information would be the same as disclosure of privileged/private individual information. Our site does not necessarily contain any other secret or sensitive information.
- Disclosure of privileged/private information about individuals could lead to user information being sold or revealed publicly, such as their address, phone number, and date of birth. This would be extremely bad and potentially ruin *Care Compare SK*, as well as make our users lives more difficult. *Care Compare SK* would potentially be viewed as an untrustworthy web application, and class action lawsuits may begin to form against us.

- Corruption of software or viruses could lead to false information to individuals. The viruses could be misleading and providing wrong information. If the viruses were to infect our users, it could potentially slow their systems down, or do other harmful things to their system. *Care Compare SK* could be known as a web application that does not provide security measures to its users, leading to its downfall.

B. Type(s) of security required:

- No types of physical security are specifically needed currently.
- All users have the same security roles and privileges.
- All users have access to edit their own information. As well as create a forum and/or reply to any forum already posted. Users do not have permission to delete other users' posts.
- There are no system administrator functions or functionality currently implemented. Any administrator issues will be resolved by Kegan or Daris.

4.3.2 Reliability

Care Compare SK is a web application built for and from the community. The system itself should be very reliable allowing clients to access the application. The system shall not allow for database injection or allow for data to be compromised. However, we currently implemented no type of access control for our users due to being granted access if only given Kegan's public IP along with the port. If this were a published application, we would implement access control before publishing.

Some of the data provided by the community itself may not be 100% accurate or reliable and will have to be taken with a fine eye. Users are free to post prices and forum discussions about anything they choose. Being that the only administration looking over *Care Compare SK* currently is Kegan or Daris, if an issue were to come up, it would be solvable as long as the issue is known. Restrictions could be applied in the future to certain input boxes to make sure no harmful data (swears in the forum, biased prices when entering quote prices, etc.) is being entered.

4.3.3 Recoverability

A. In the event the application is unavailable to users (down) because of a system failure, how soon after the failure is detected must function be restored?

- *Care Compare SK* is an informative site for the community. In the event the application is unavailable to users, the failure must be detected within a day. *Care Compare SK* is not a critical web application for users to have, it is considered a place where users can go to help speed up the process, and gain knowledge about insurance providers in Saskatchewan. Due to being a non-critical application, as long as the failure is discovered and resolved within a day, we believe that is a sufficient restoration time.

B. In the event the database is corrupted, to what level of currency must it be restored?

- The database must be able to be restored to no more than one (1) week before the corruption occurred. Due to being a non-critical application, as long as the failure is discovered and resolved within a day, and restored to data used within the past week, our application will succeed.

C. If the processing site (hardware, data, and onsite backup) is destroyed, how soon must the application be able to be restored?

- Our application would be able to be restored as soon as new hardware is acquired. This is because *Care Compare SK* is stored on *Github* as a source control. Due to the unlikeliness of the hardware ever being destroyed, the application would be restored in less than a day. If the hardware were to be destroyed, it would most likely take up to three (3) days to restore the system.

4.3.4 System Availability

The system has no specific availability times. *Care Compare SK* is simply an informative site to compare and talk about insurance providers. The system should be available as long as our server is running.

4.3.5 General Performance

A. Response time for queries and updates

Response time for queries and updates should be near instantaneous

B. Throughput

Due to currently running on Kegan’s personal computer, it should be able to handle a reasonable amount of throughput. It is unsure as to how many users can be on the application at once without any lag, crashing, and degrading of the web application in any way

4.3.6 Data Retention

If being used as a real world application, the system shall retain database information for the lifetime of the web application. Even if the information posted by a user was long ago, it is still useful information to predict future values.

4.3.7 Error Handling

If a user is able to somehow inject false or malicious information, our application should throw an error and terminate our Express App and kill the web application. A restart of the application is manually required once terminated.

APPENDIX A - GLOSSARY

HTTP: Hyper-Text Transfer Protocol

SQL: Structured Query Language