**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 |

CONTENTS

1 INTRODUCTION 4

1.1 Purpose 4

1.2 Scope 4

1.3 Background 4

1.4 References 4

1.5 Assumptions and Constraints 4

1.6 Document Overview 5

2 METHODOLOGY 5

3 FUNCTIONAL REQUIREMENTS 5

3.1 Context 5

3.2 User Requirements 5

3.3 Data Flow Diagrams 6

3.4 Logical Data Model 6

3.5 Functional Requirements 6

4 OTHER REQUIREMENTS 6

4.1 Interface Requirements 6

4.2 Data Conversion Requirements 7

4.3 Hardware/Software Requirements 7

4.4 Operational Requirements 7

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.

## 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.

## 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.

## 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.

## 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.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>

## Assumptions and Constraints

### 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.

### 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.

## Document Overview

TBD

[Provide a description of the document organization.]

1. **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**

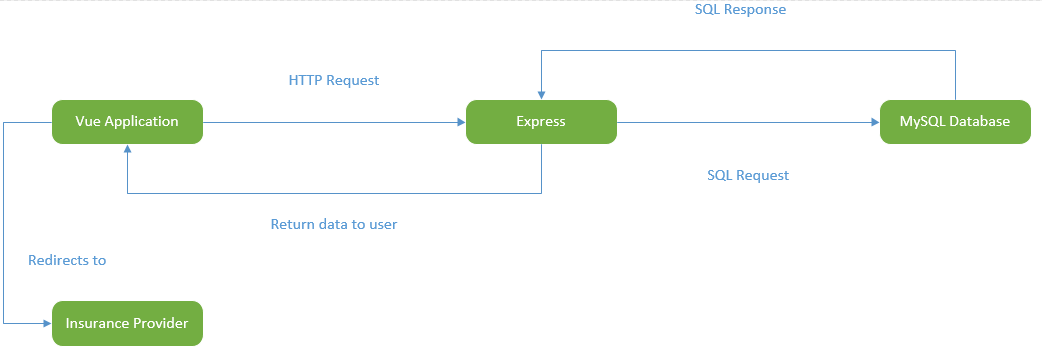
We used Adobe XD software to create low and high fidelity prototypes of our application. Adobe XD projects can be quickly and easily modified and changed which fits well with the Agile methodology.

1. **FUNCTIONAL REQUIREMENTS**

## Context/Communication

Our Vue Application (user interface) sends an HTTP (Hyper-Text Transfer Protocol) 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.

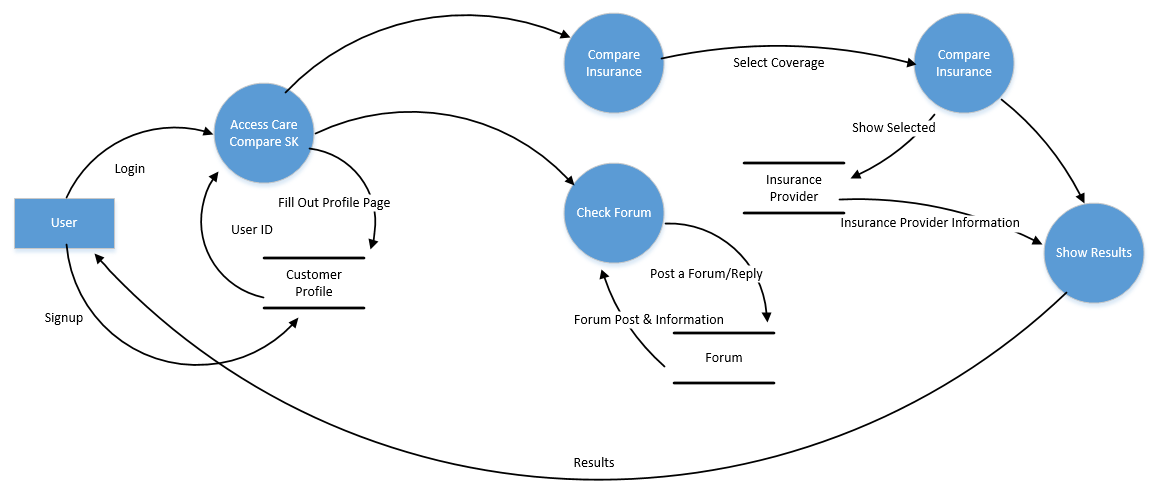
Exhibit 1 - Generic Context/Communication Diagram

**

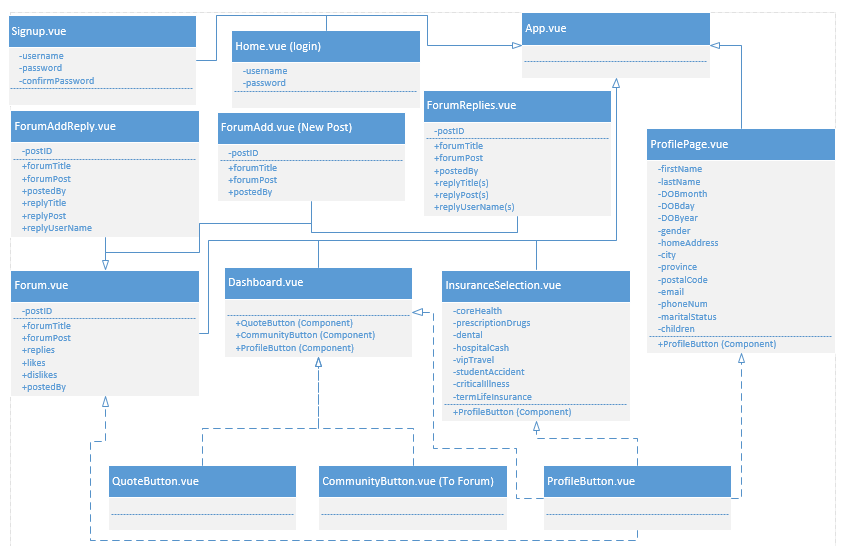
## User Requirements

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



## Logical Data Model



## Functional Requirements

[List the functional requirements of the system.]

### Functional Requirements Group 1

[List the functional requirements for each functional requirements group.]

Exhibit 4 - Sample Requirements Group 1

|  |  |
| --- | --- |
| **Section/ Requirement ID** | **Requirement Definition** |
| FR1.0. | The system shall [parent requirement group 1]. |
| FR1.1 | The system shall [child/parent requirement]. |
| FR1.1.1 | The system shall [child requirement]. |
| FR1.1.2 | The system shall [child requirement]. |

### 

### Functional Requirements Group 2, Etc.

1. **OTHER REQUIREMENTS**

## Interface Requirements

### Hardware Interfaces

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

### Software Interfaces

BlueCross – When selecting external link

Sunlife – When selecting external link

…

### 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.

## 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.

## Operational Requirements

### 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.

1. 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.

### Reliability

A. [State the following in this section:

1. State the damage can result from failure of this system—indicate the criticality of the software, such as:
2. Loss of human life
3. Complete or partial loss of the ability to perform a mission-critical function
4. Loss of revenue
5. Loss of employee productivity
6. What is the minimum acceptable level of reliability?

B. State required reliability:

1. Mean-Time-Between-Failure is the number of time units the system is operable before the first failure occurs.
2. Mean-Time-To-Failure is the number of time units before the system is operable divided by the number of failures during the time period.
3. Mean-Time-To-Repair is the number of time units required to perform system repair divided by the number of repairs during the time period.]

*Reliability is the probability that the system processes work correctly and completely without being aborted.*

### Recoverability

[Answer the following questions in this section:

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?

B. In the event the database is corrupted, to what level of currency must it be restored? For example “The database must be capable of being restored to its condition of no more than 1 hour before the corruption occurred”.

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

*Recoverability is the ability to restore function and data in the event of a failure.*

### System Availability

The system has no specific availability times. *Care Compare SK* is simply an informative site to compare and talk about insurance providers.

### 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

### 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.

### 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**

[Define terms, acronyms, and abbreviations used in the FRD.]