Medical Glossary for Aboriginal Interpreters

By

Jose Arturo Santoyo Guilarte

supervisor:

Professor Yvan Labiche

A report submitted in partial fulfillment of the requirements of SYSC-4907 Engineering Project

Department of Systems and Computer Engineering

Faculty of Engineering

Carleton University

2016-04-07

Abstract

This report describes the results in the design, development and implementation of a Medical Glossary for Aboriginals Interpreters application. The application's objective is to provide First Nations Canadians, and dwellers of Canada who work in the medical sector, a conversational tool, and to facilitate their communication in the case of any language barriers concerning medical terminologies. The solution to this problem was implemented taking into consideration the various environments in which users can navigate through the App. The results were successful. The application has a correct functioning on the web and it has been launched on Android environment as well.

Contents

List of Figures

1 Acknowledgments

I would like to thank my supervisor Yvan Labiche for giving me the opportunity to work in the development and implementation of the Medical Glossary for Aboriginal Interpreters web and mobile application, for his guidance and advice. I would also like to thank the technical director of the client's contact, Delasie Torkornoo for providing me the necessary information to pursue the design, development and implementation of the application, for being available at my request regarding questions about the software and the database model used.

2 Introduction

2.1 Background

According to 2011 census, aboriginal peoples in Canada totaled 140,0685 people, or 4.3 percent of the national population, spread over 600 recognized First Nations governments or brands with distinctive cultures, languages, art, and music [2] This project is motivated by the idea that in Canada there is a vast First Nations ethnic group that could make use of language tools to better communicate within the medical domain. Other majorities would also benefit from having the possibility of improved coping with language barriers, as well as expanding their knowledge about aboriginal languages. Effective communication is an asset from which society has always benefited. The development of this project will provide an alternative tool of communication to aboriginal groups in North America whose first language is Cree; as well as to people who need to communicate with these minorities, when they have poor or no knowledge of Cree.

An example of this language barrier problem is seen in the communication between doctors or nurses and their patients. Doctors usually use specific and scientific terms that patient are not necessarily familiar with. The communication and necessary explanation of these terms is further complicated when is done from one language to another. Language barriers can lead to misunderstandings and frustration, both of which can have harmful effects in the realm of medical care. Misunderstandings could lead to patients following medical advice partially or incorrectly, which can lead to medical complications. Furthermore, the cultural differences in opinions about health that might arise would be navigated in a more instructive manner, if medical professionals and patients could better communicate their

perspectives, identify the source of difference in opinion, and problem-solve around it. For these reasons it would be helpful to have a source of information that patients and the physicians or nurses can use in order to find the necessary terminology and to understand each other effectively. It is believed that this facilitated communication would be specifically helpful for Cree speaking aboriginals and their medical professionals as well. Given that Cree is one of the most spoken aboriginal languages by First Nations in Canada[3]; the solution was narrowed to the design, development and implementation of an application that targets the Cree, English and French Languages.

2.1.1 Statement

First Nations people living off-reserve exceed those who live on-reserve by 20 percent. Their presence in urban centers will continue to grow [4]. The problem arises when rural or urban medical professionals without knowledge of Cree need to communicate with First Nation peoples who do not have strong English/French language skills. In this case, the aboriginal patient would benefit from an interaction with someone who has some knowledge of Cree terminology and both parties interacting would benefit from having a tool to facilitate communication across the two languages

2.2 Proposed Solution

The proposed solution to these problems is to create a web application that will allow the user to navigate through a medical database, and search for terms, conversational phrases, audio sources, images, and finally the translation of all these searchable contents within the medical context. It is also taken into account that we are living in the mobile era hence the need to deliver this solution across mobile platforms that are currently on the market, i.e. Android and iOS.

A full description of the stages of the requirements elicitation, analysis, design, implementation and development of the medical glossary for aboriginal interpreters application is presented in detail in the following sections. The App has been successfully deployed on the web, and across Android and iOS platforms.

3 Technical Section

3.1 Requirement Elicitation

The gathering of requirements from the client was mostly done by the client's technical contact. The majority of the requirements were given prior to the analysis, design and development of the App. However during the multiple iterations performed some other non-functional requirements were elicited, e.g. specifications in the design of the view of the application, like chosen layouts, icons, behaviors of different components.

3.2 Analysis

The realization of the Medical glossary for Aboriginal Interpreters consists of a web and mobile application. The user can navigate through the application on different Operating Systems(OS) environments without noticing any difference in the way the User Interface(UI) is displayed or interacted with from one another. Basically the same look and feel is transported from the web environment to the various mobiles platforms in which the system is deployed. The idea behind a multi-platform application is to increase its accessibility, and usability. The way this is implemented is by gathering the necessary information about the main components that are used to build the frame for the web App (e.g. Tabs, List, Settings). The frame is built and some details are introduced which characterized these main components(i.e Tabs are populated with some content). Lastly these functionalities are tested. This is then repeated such that each iteration is a more detailed implementation of the last one. A version control(VC) system is used to keep track of the changes made during the development and implementation of the App. A relational database was designed favoring the structure of the application. Medical terms and translations have an unique identification and also a parent/child relationship that is conveniently used during the development of the App.

3.3 Solution approach

An Agile development process is a suitable methodology for this project. It provides the guidelines to peruse iterative access to the project's life-cycle throughout the implementation and development process [5]. Multiple iterations were performed until the completion of the alpha version. A combination of Feature-Driven Development(FDD) and Extreme Development(XP) agile methodologies best describe

the flow of these iterations and the overall dynamic followed to completion of the App. Requirements elicitation, design development, and implementation were part of these iterations, and are explained in detail in section 3.4. These iterations contributed to the incremental overall progress in the product's result. The opportunity of getting feedback at early stages and, having continuous revisions was thought to reduce the development cost. Agile methodology, in particular FDD and XP, re-evaluates the direction of the project after each iteration [5].

3.4 Design and implementation

The App was built with apache Cordova software, AngularJS, and ionic framework. This decision was made after meeting with Delasie (Client's technical contact), and collaboratively eliciting the requirements. It was concluded that using these softwares was one of the best approaches in order to deliver regular builds of the system, to demonstrate features at very early stages of the development, and to introduce milestones at which new requirements could be added to the systems, or modified from the ones that were previously elicited [6]. The requirements were:

- Having a product that could be deployed in Android, iOS devices and on the web.
- The Mobile App should have the same look and behavior regardless of which native platform is used.

As well the functional requirements are:

- The app will display medical terms in English or French and provide the correct translation to Cree North or Cree South at Users' request.
- It will provide media (video and images) corresponding to the terminology queried.

Fig. 1 is a use case digram representation of the overview of design of the user interface. The application provides the user with tab Glossary, tab Diagrams, tab More, and tab Bookmarks. In tab Glossary the user will have the option of selecting or searching for medical terms under the corresponding category in English or French, and see the translations in Cree.In tab Diagrams, diagrams can be selected for display, and the audio of the terms displayed on the figures are also provided at users request(on-click). In Tab More the user can set the languages preferences (English or French), can search for conversational phrases, which also displays their textual translations in Cree, and plays voice translations, and finally

the user can view the acknowledgments. Tab bookmarks is not fully implemented in the released version of the application, however it is still part of the UI.

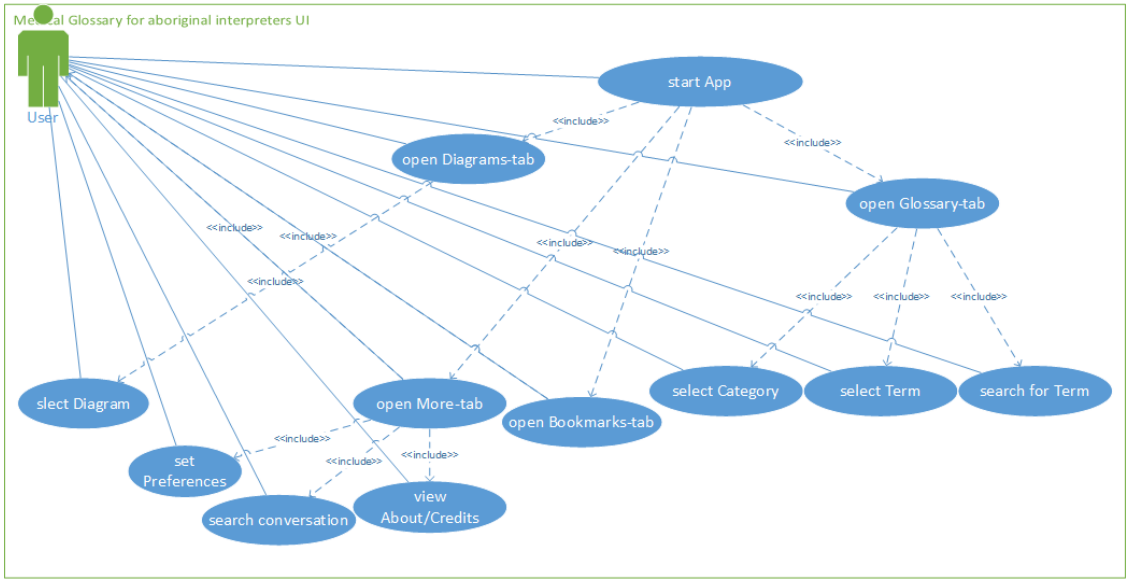

Figure 1: Use Case Diagram model

3.4.1 Graphical User Interface design

The design of the Graphical User Interface (GUI) was done collaboratively with the client's technical contact. Ionic tools were utilized to layout the different components of the GUI. Ionic is a Cascading Style Sheet (CSS) framework and also a javaScript UI library [7]. It provides template elements that can easily be added to the App (e.g tabs, buttons, lists, list-items, icons, search boxes, etc.). The GUI was desired to have tabs that allow the user a rapid access to the content of the application and easy navigation throughout its main components. Fig. 5 shows how tabs Glossary, Diagrams, Bookmarks, and More were laid out at the bottom. It was agreed that the default view would be the one from tab Glossary, given that it displays and provides access to the main features of the application, terms and term translations.

3.4.2 MVC and AngularJS

The implementation of the App follows the guidelines of the Model-View-Controller(MVC) architecture pattern. The MVC implementation ensures a high level of modularity within the application components [8]. This is in fact very useful when working in an agile environment. It makes it easier to understand

and design a particular unit without affecting others. It establishes a dynamic flow on each iteration during the development process of the application.

In Fig. 2 the MVC components of the system are presented. Medical Glossary Controller manipulates the model(Medical Glossary Model) and updates the scope of the view of the system. The model corresponds to a relational database that was provided by the client's technical contact and its design was developed favoring the easy access to the categories, terms, and terms translations, as shown in Fig.3.

MVC architecture was implemented using AngularJS built-in services and dependency injection. These features allow to dynamically update the model of the system(i.e inject the data resource in a factory function) when an event happens which in turns updates the view of the application [9]. It was possible to customize the service to provide access to the phone data on the fly. Controllers are used to modify the view of the application upon events triggered by the user. Services and Controllers can be overwritten in order to obtain the desired event response from the system giving a higher level of abstraction to programmers.

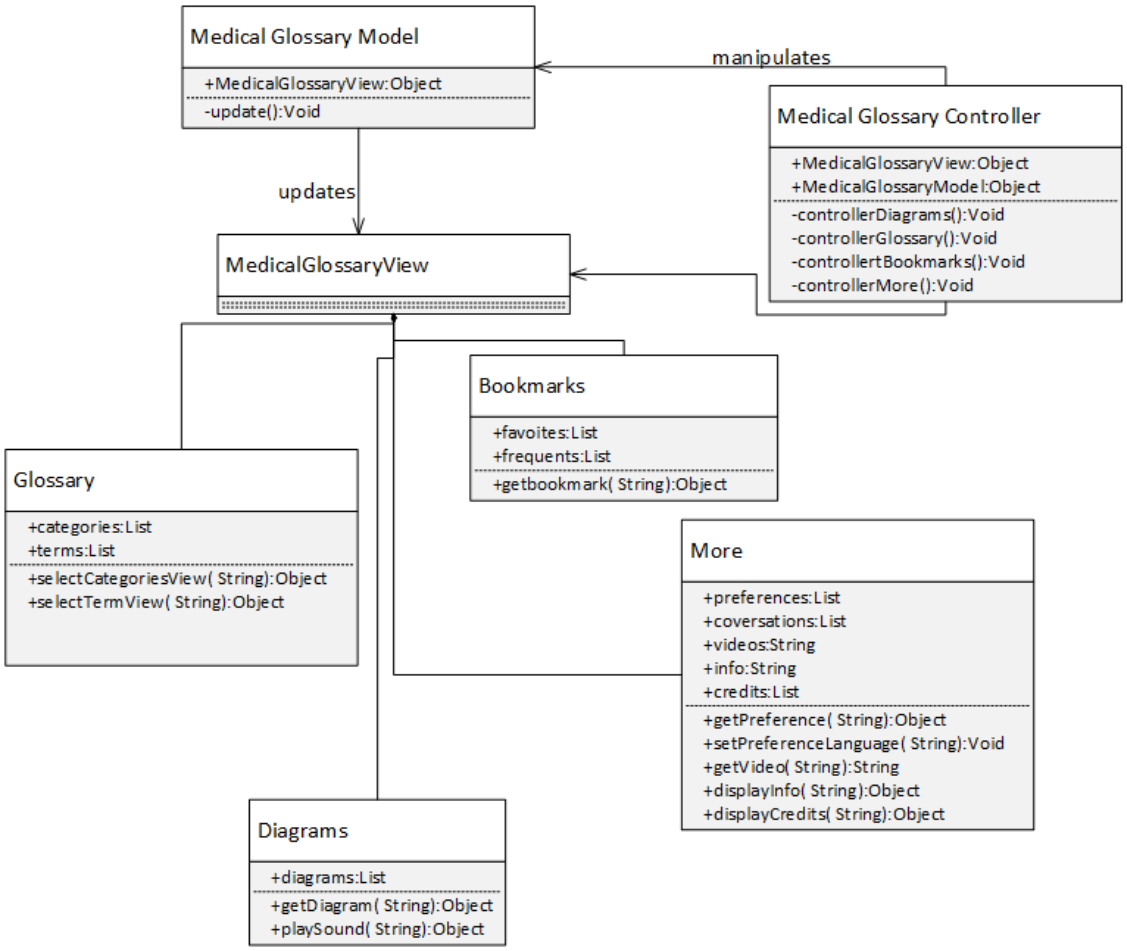

Figure 2: MVC Medical Glossary for aboriginal interpreters

Fig.3 is a Relational database diagram, where Category object is described by name, unique identifier number(uuid), the location in which the object is found(locale), this location is related to the language of the object, the number of terms this object is parent of (term_count), and the actual list of term objects(term[]),lastly the parental description of the object(parent). Term object is described by its unique term id number(termUUID), its corresponding translation or translations (termTranslation), its category dependency(termCategory) and the language in which it is presented(termPublicNote). Translation Object is described by its term dependency(term), its actual translation(definiton), and its description of location relating to the language of the translation(locale).

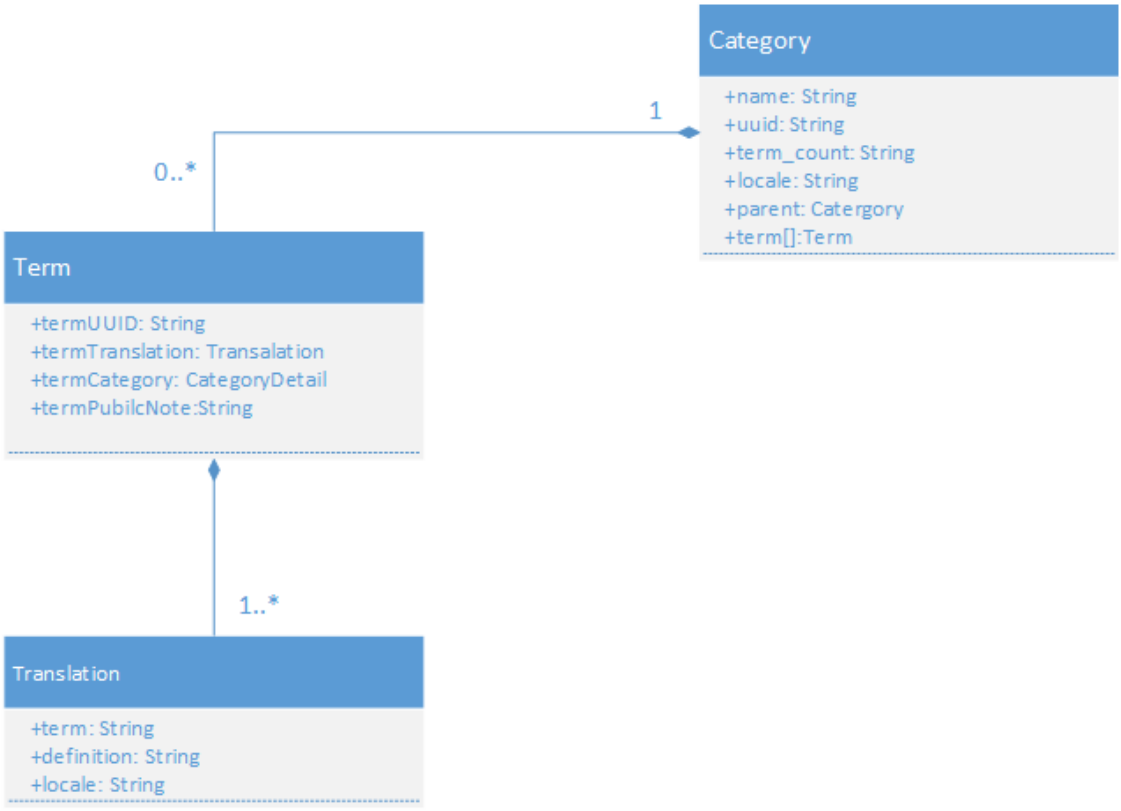

Figure 3: Relational database diagram

The application was implemented in a web environment. It executes in a web view by default and makes reference to HTML, AngularJS, CSS, and the database file resources (see Fig. 4 top left corner). It was deployed on an Android phone using Apache Cordova shown in Fig 4 as the bigger box containing Web App, Cordova Plugins and HTML engine. Apache Cordova is a software that implements the iOS and Android native version of the App. It is a JavaScript Application Programming Interface(API), which serves as a wrapper for native code and is consistent across devices [10]. It uses a web view that can run the HTML app on each different platform. Ionic framework provides the style and behavior like UI native applications using HTML5 CSS and JavaScript [11]. In the same manner Cordova allows to use standard web technologies to develop the App and has pre-developed plug-ins that allow to execute targeting different platforms (eg. Android, iOS, Windows) [1] represented in Fig.4 as the rectangular box at the bottom which contains services, input, graphics and sensors. This is how having a multi platform application of the Medical Glossary for Aboriginal Interpreters was achieved.

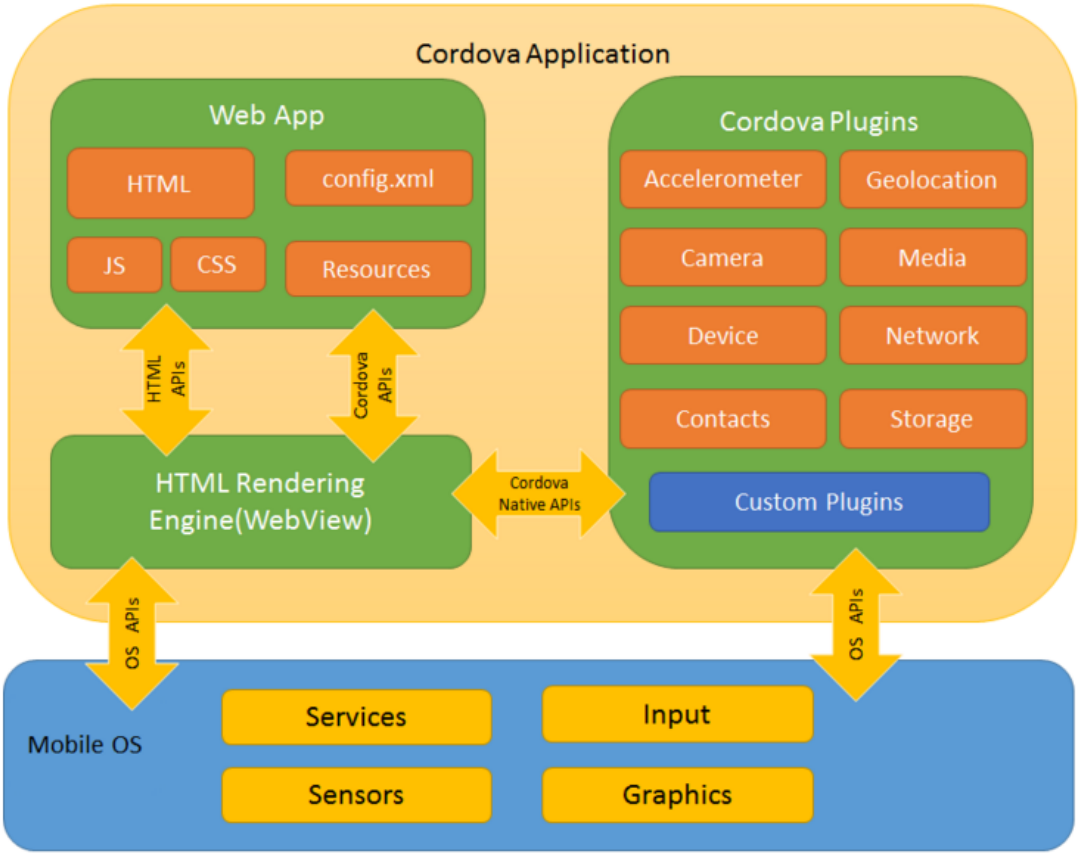

Figure 4: Cordova Application [1]

Figures 5 and 6 capture the App running on the web browser.

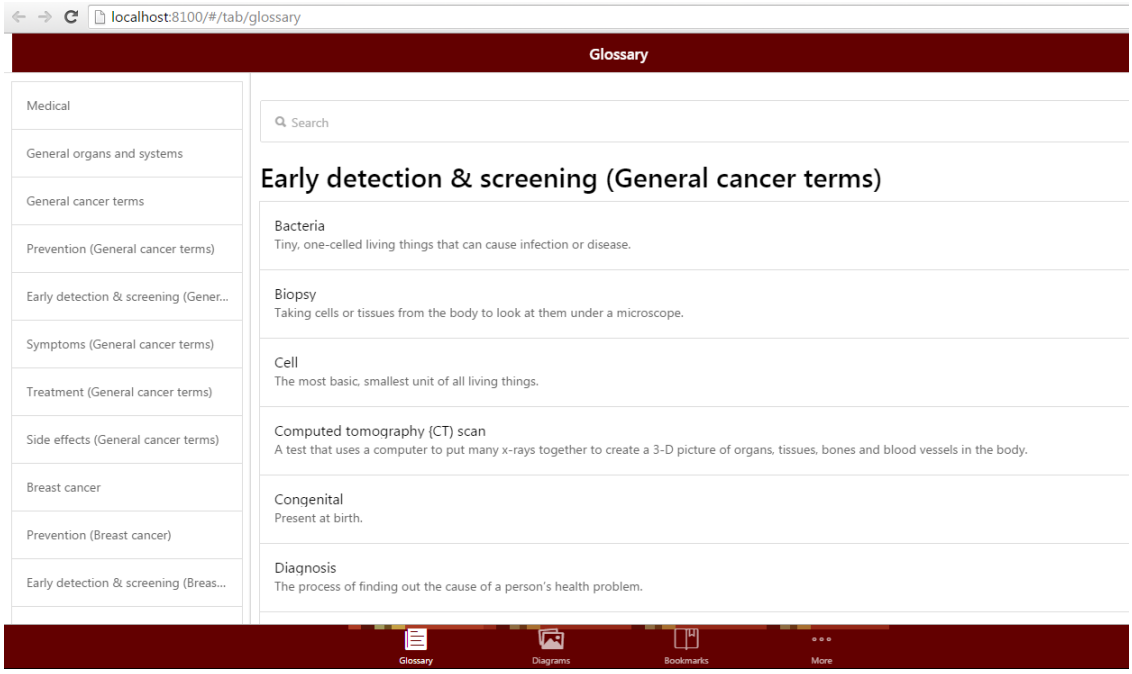

Figure 5: Web view Glossary-tab

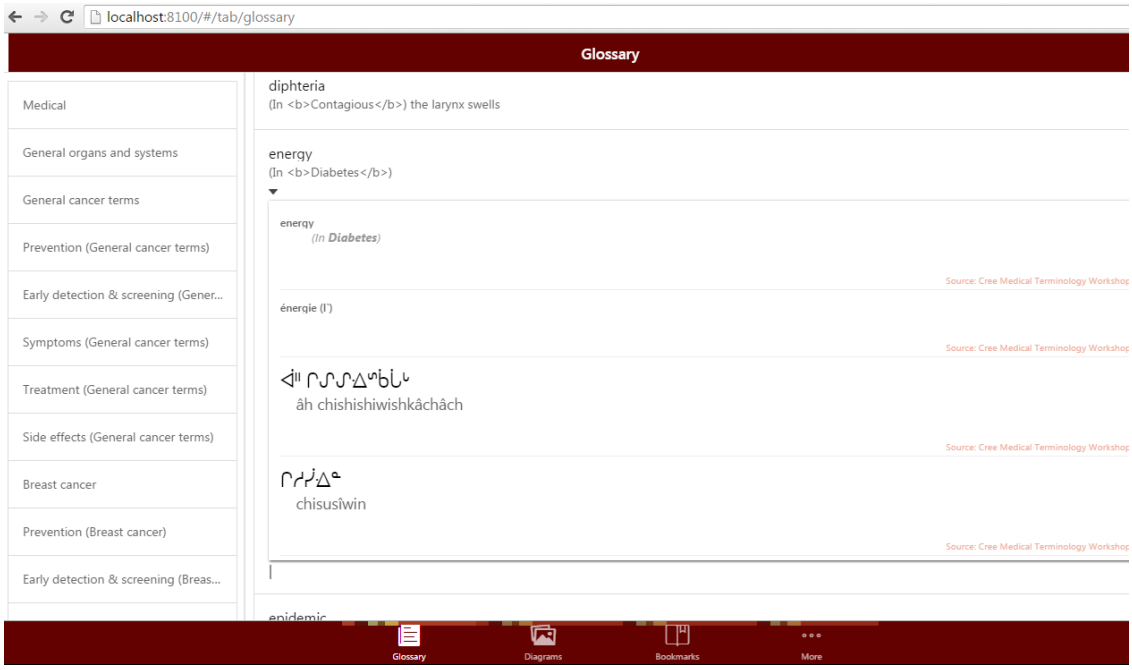

Figure 6: Web view Glossary-tab Cree entry translation

In Fig. 7 a minimized view of the web App is shown. Note that the look and feel of the web version is practically the same as in the mobile versions shown in Figures 8, and 9

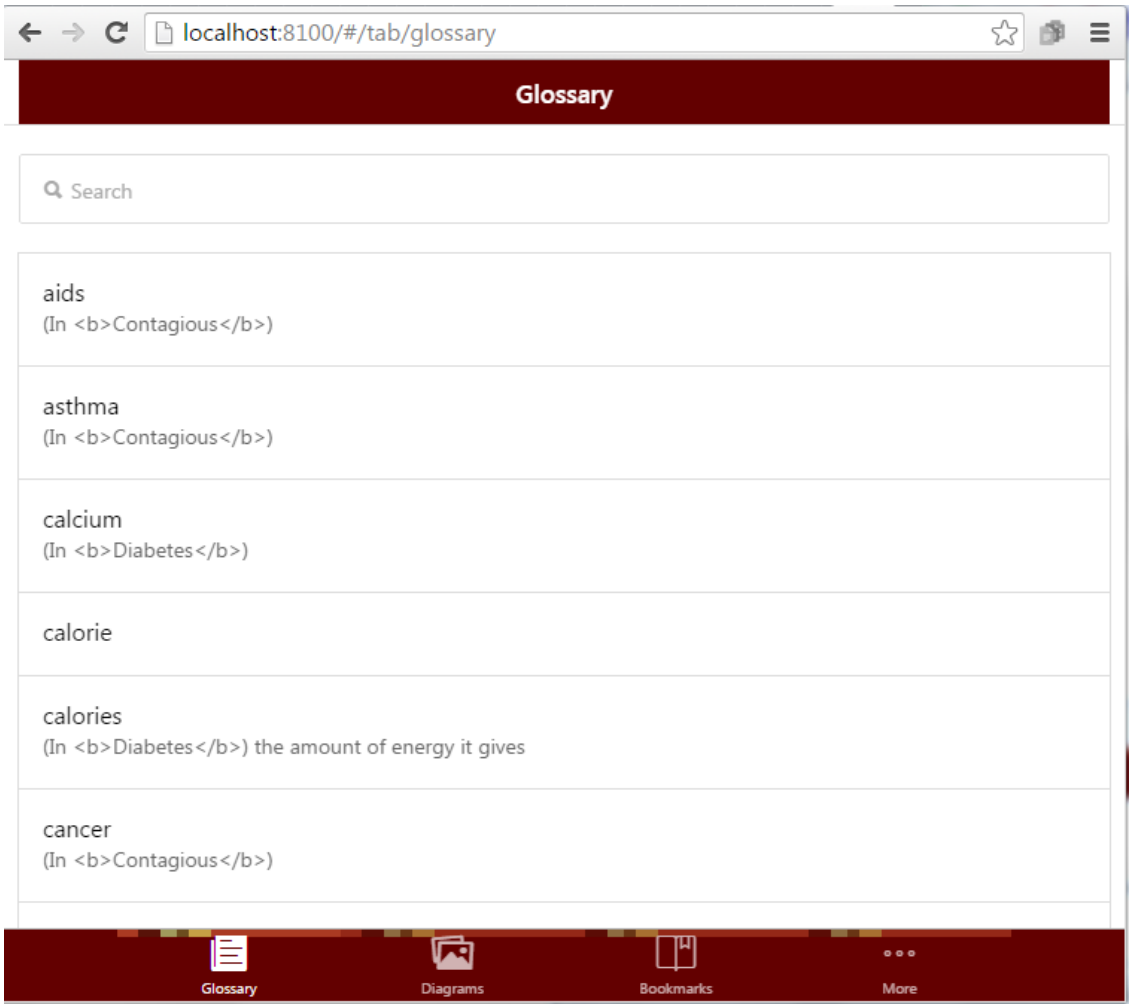

Figure 7: Web view Glossary-tab minimized

Fig. 8 captures the state of an entry translation in the android environment, as well as the preference settings in the iOS virtual platform.

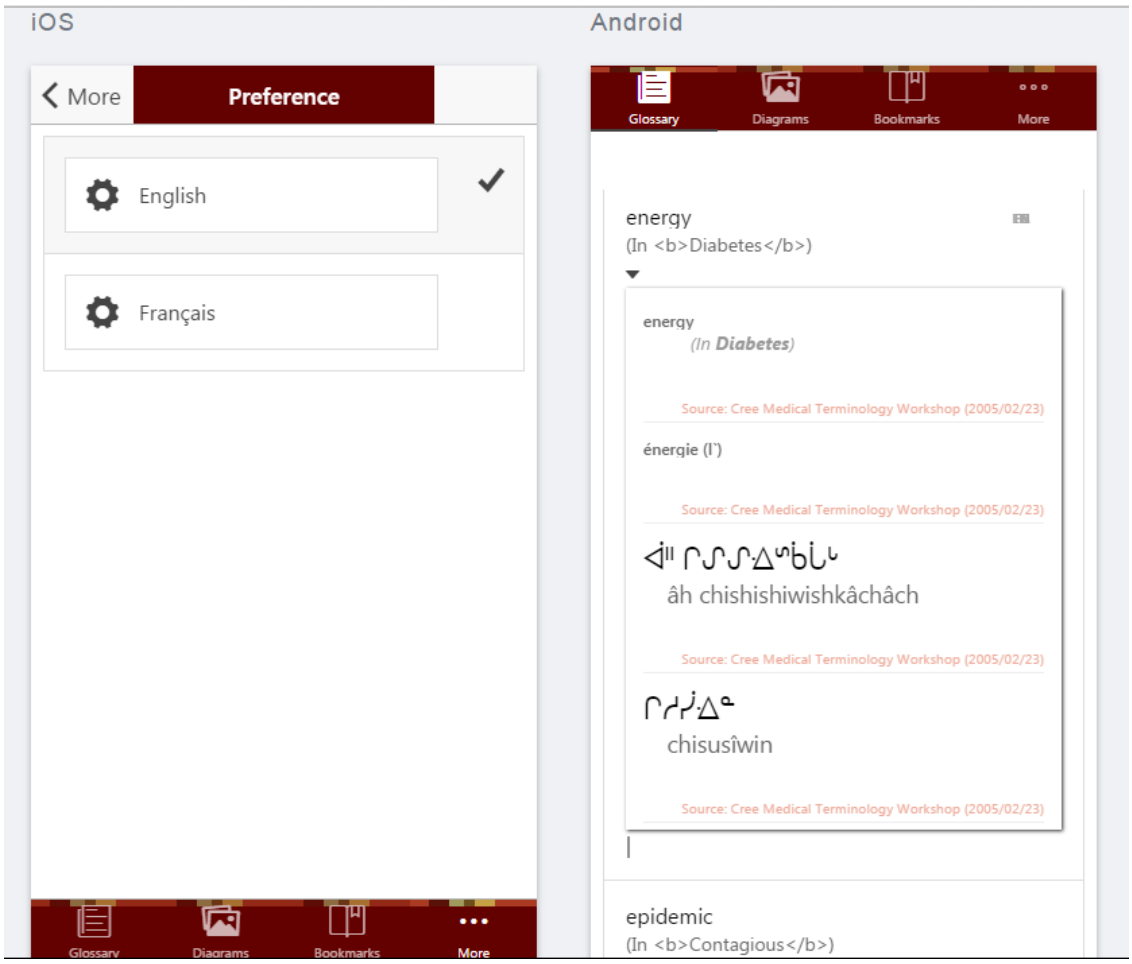

Figure 8: iOS & Android virtual platform view Glossary-tab Cree entry translation

Fig. 9 represents the state of conversational phrases in Android virtual environment, and tab More in iOS.

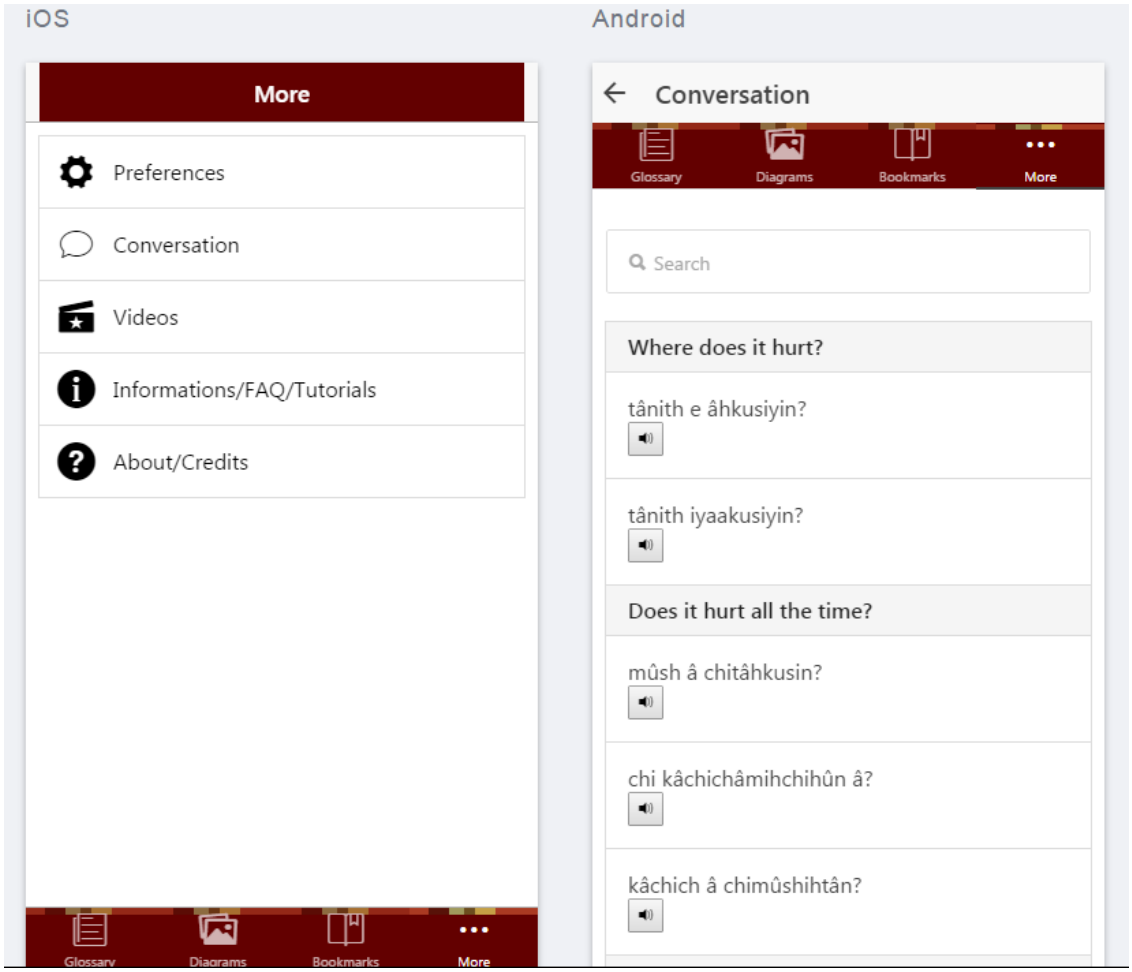

Figure 9: iOS & Android virtual platform view More-tab conversation

3.5 Verification and Validation

As previously mentioned AngularJS supports dependency injection(i.e., passing components dependencies to a function). Essentially testing individual parts of the system becomes relatively easy. Components that have their resources injected can be mocked on a test by test basis without having to affect the rest of the system that is not being under test. This high level modularity is also achievable because the MVC implementation of the application favors high cohesion and low coupling [12]. Any request from the system can be simulated by mocking the dependencies and passing them into the components under test. The DOM is abstracted so the model can be tested without having to do any extra manipulation on it [13]. It is important to mention that the correctness on the behavior of each individual component of the App is verified on the web environment. This means that testing is done in only one environment.

Apache Cordova is used to wrap-up the HTML code and build the application package for Android or iOS(e.g. apk, ipa). Once the packages are built for the native platforms the application is ready to run on them. So even though Medical Glossary for Aboriginal Interpreters is a multi-platform application. Testing only needed to be done in the web environment. By default angular applications run on Google Chrome however there are tools that can be used to test Angular applications against a number of browsers in which the web application is wanted to work on. For example Karma is JavaScript command tool that allows to run the test cases of the web application against different browsers [13]. In that case the App would not only function on Android, iOS, or Google Chrome but also in other web browsers. It is necessary to add that due to time constrains the application was not tested on other browsers.

During each iterations in the development process the functional and non-functional requirements were addressed and validated with the client's technical contact as they were introduced to the system, modified and verified accordingly.

4 Conclusions

Developing a single application for multiple platforms can be time consuming even for experienced developers. The softwares used to develop mobile Apps are updated at a fast pace compared to the softwares used to develop web Apps. It might take the same amount of time for someone to develop an App for a specific platform (e.g., iOS), as for a new update to come through.For instance, the beginning of HTML goes back to the 1990s and up to today only 5 versions have been released [14]. It is true that native application had its advantages over web applications running on natives environments in the past. However this has changed over time. With the developers' investments in HTML5 and the use of AngularJS to manipulate the Document Object Model(DOM) of an application, web Apps run as fast as in native environments. They can efficiently update the browser's DOM improving performance [15]. Therefore the solution to this problem can be found in the implementation of hybrid Apps, Web based Apps that with a platform independent wrapper can be deployed on different mobile environments without affecting the look and feel of the application over cross-platforms.

Medical Glossary for Aboriginal Interpreters application was successfully implemented and deployed over multiple platforms. It met all the requirements and is an example of a web-based application that is functioning across multiple platforms. Users will be familiar with the application's layout independently of the environment, eg.(web, Android,iOs), and will feel comfortable navigating through the App's components.

References

[1] A. Cordova, "Apache cordova," 2012. [Online]. Available: https://cordova.apache.org/docs/en/latest/guide/overview/

[2] "Projections of the aboriginal population and households in canada," 2016. [Online]. Available: http://www.statcan.gc.ca/pub/91-552-x/91-552-x2015001-eng.pdf

[3] "Aboriginal languages in canada," 2016. [Online]. Available: https://www12.statcan.gc.ca/census-recensement/2011/as-sa/98-314-x/98-314-x2011003_3-eng.cfm

[4] "Urban first nations health research discussion paper," 2016. [Online]. Available: http://www.naho.ca/documents/fnc/english/UrbanFirstNationsHealthResearchDiscussionPaper.pdf/

[5] "Agile methodology," 2016. [Online]. Available: http://agilemethodology.org

[6] "What is agile methodology?" 2016. [Online]. Available: https://www.versionone.com/agile-101/agile-methodologies/

[7] "Extend ionic even further with the power of angularjs," 2016. [Online]. Available: http://ionicframework.com/docs/api/

[8] "A description of the model-view-controller user interface paradigm in the smalltalk-80 system," 2016. [Online]. Available: https://web.archive.org/web/20100921030808/http://www.itu.dk/courses/VOP/E2005/VOP2005E/8_mvc_krasner_and_pope.pdf

[9] "Rest and custom services," 2016. [Online]. Available: https://docs.angularjs.org/tutorial/step_11

[10] A. Cordova, "Developing mobile applications with cordova," 2016. [Online]. Available: http://www.toptal.com/mobile/developing-mobile-applications-with-apache-cordova

[11] "Welcome to ionic," 2016. [Online]. Available: http://ionicframework.com/docs/guide/preface.html

[12] "Cohesion vs coupling separate concerns," 2016. [Online]. Available: http://www.agile-code.com/blog/cohesion-vs-coupling-separate-concerns/

[13] "Unit testing," 2016. [Online]. Available: https://docs.angularjs.org/guide/unit-testing

[14] "The history of html," 2016. [Online]. Available: http://www.ironspider.ca/webdesign101/htmlhistory.htm

[15] "React may have just ended the native vs. web debate," 2016. [Online]. Available: http://readwrite.com/2015/09/25/react-native-mobile-apps-web-apps-marc-shilling