Software Requirements Specification For Team Notato

January 3rd, 2020
Version 1

Prepared by:
Bunnarith Heang

Table of Contents

1 Introduction

1.1 Overview

Notato Calculator Web-App will be a web-based application that enables users to use calculator functions that are available in normal calculators. The application will provide users with access to different modes of a calculator, including simple mode, scientific mode. Moreover, the application will also include the Unit Converter, which allows users to convert to/from different units and for authenticated users to use Randomizer (Team Generator, Random Picker, Decision Maker, etc..).

The purpose of this document is to provide information on the requirements for the application. Project goals, scope and definitions are given in the introduction. Design constraints and application environments are described in the following section. Non-functional requirements are outlined for later verification. Functional requirements are given to show the system features and expected user interaction.

Project constraints will be included in the separate documentation. The Software Project Management Plan (SPMP) will illustrate the project's planning and scheduling.

1.2 Goals and Objectives

The prime objective of this application is to enable users to access multiple mathematical operations, unit conversion, and randomizer in one application. This goal is to ultimately offer these functionalities without accessing multiple platforms.

The expectations from this project is to:

1. Provide an application interface to users to access.
2. Provide mathematical calculators in different modes(simple, scientific) for specific users.
3. Convert the unit of measurement, weight and volume, dimension, digital data and datetime to a different unit.
4. Enable logged in users to use Randomizers, save the data in the history.

1.3 Scope

The application will provide users (Guest, and Authenticated User) the abilities to use the calculator operations and the unit conversion, with authenticated users having extra functionality. Users need to be logged in inorder to use the Randomizer functionality.

1.4 Definitions

Notato Calculator Web-App – the product that is being described here; the software system specified in this document.

Use case – describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

Developer – the person or organization developing the system, also sometimes called the supplier.

Client – the person or organization for which this application is being built.

Simple Calculator – offers simplistic operations of a calculator. It includes algebra operations (+,-,*,/,%), find exponential, and square root.

Scientific Calculator – offers extra operations on top of the simple calculator. Including trigonometry, logarithm, algebra operations.

Converter – convert to/from units of measurements, weight, dimension, volumn, digital data, datetime and reversing the conversion unit.

Randomizer – This provides functionalities such as random picker, decision maker, team generator, custom list. Also users will be able to save the result, and should the users choose to export the data as an XLS file.

Guest – the person or people who will interact with the application with permission to access the two modes of calculators and unit conversion.

User – the person or people who will have the permission to access the normal functions and on top of that the randomizer function.

2 General Design Constraints

2.1 Notato Calculator Application Environment

The Notato Calculator will be based on a website that is designed to work on any browser. The application will interact with the backend server for the randomizer function, while the unit conversion will be interacted with a third-party API, and the two modes of mathematical calculators will be done on the client side.

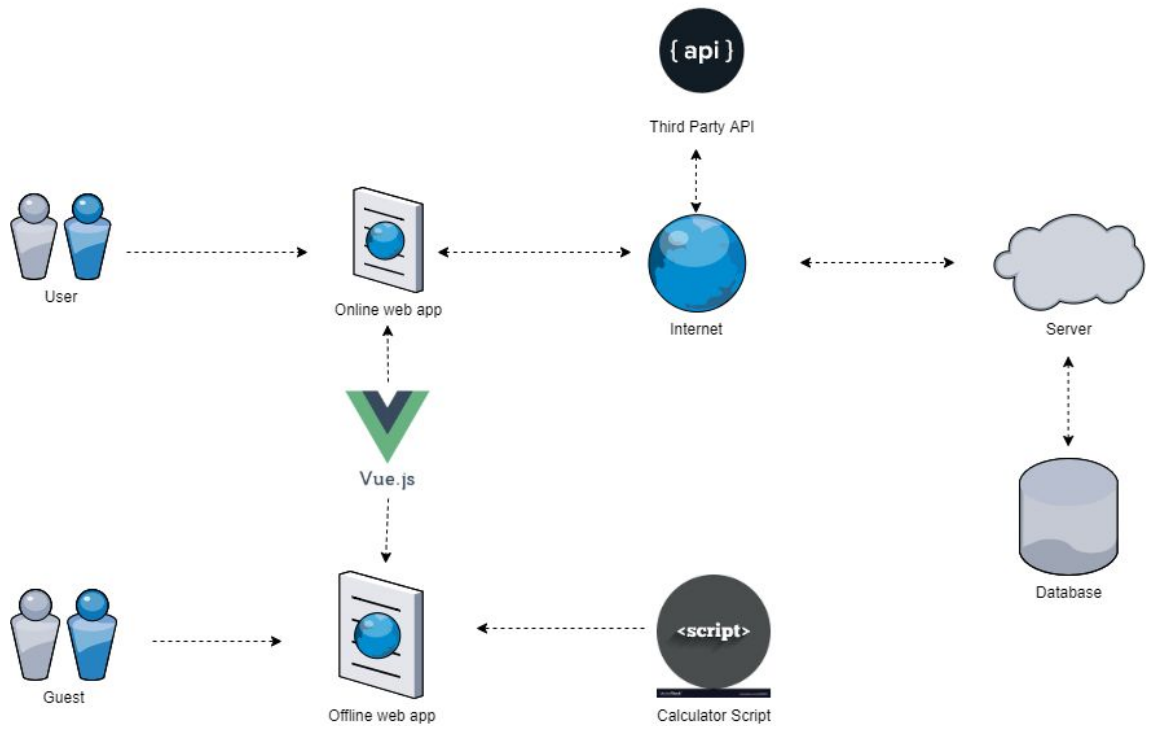

Figure shows the system behavior of Notato Calculator Application ([Diagram])

2.2 User Characteristics

Guests: will be unauthenticated users, who have the ability to access most of the functionalities offered by the application (simple and scientific calculators, and unit conversion).

Users: these will be authenticated users who will have extra permissions to access the randomizer and save the result, and also export the result.

2.3 Mandated Constraints

The application will run on any browser. This platform was chosen as the result of the requirements from the client.

3 Nonfunctional Requirements

3.1 Operational Requirements

We want the application to be straightforward, easy to use and offer an interactive user interface. So almost certainly all users who access the application will have a clear understanding of what functionality does what.

3.2 Performance Requirements

The application is expected to work even if the application goes offline for some certain functionalities. Given that that specific page has been fully loaded before internet disconnection, the parts that need to be still functional are the simple and scientific calculators. But if the page hasn't been fully loaded, none of the functions will work if there is no internet connection.

3.3 Security Requirements

The application should offer a secure and uninterrupted authentication process to users. The storage of user's password should be hashed to prevent the lookup for password in the database, or the case of the breach in the database.

3.4 Documentation and Training

The Notato Calculator application will be delivered to users as a download without documentation or training. A user guide and system documentation will be provided to project stakeholders.

3.5 External Interface

3.5.1 User Interface

The user interface will be straightforward as we want users to get used to the application as fast as possible. The transition between different modes should be seamless and eye-catching.

3.5.2 Software Interface

The simple and scientific modes of the calculators will be calculated on the client side. In addition, the system utilizes a third-party API for the conversion of different units, while the backend server will be the interface between the randomizer function and the user.

4 Functional Requirements

4.1 Required Features

4.1.1 Use Case: Simple Calculator

Diagram

Description: This includes only algebra operations

Actors: Guest & User

Value: ?

Cost: ?

Preconditions:

- Required the page to be fully loaded in order to enable offline mode

Basic Path

1. User enters the application
2. The simple mode calculator should be the first thing that the user sees.
3. User enters the numbers and operations to compute.
4. Application compute and show the result of the calculation.
5. Should the user choose to clear the current input, the user can re-enter the current number again. (not sure the word for current input)
6. If the user chooses to clear everything, then all previous inputs will be erased, and the user will be able to start again.

4.1.2 Use Case: Scientific Calculator

Diagram

Description: Includes algebra operations and most of the scientific operations available on the physical scientific calculator.

Actors: Guest & User

Include value, cost ?

Preconditions:

- Required the page to be fully loaded in order to enable offline mode

Basic Path

1. User enters the application
2. System will display different modes user can select on the drawer
3. User select scientific mode
4. (The same path applies as the simple calculator)

4.1.3 Use Case: Unit Conversion

Diagram

Description: Users can convert to/from different units of measurements, dimensions, weight, etc...

Actors: Guest & User

Preconditions:

- Stay connected to the internet in all time

Basic Path

1. User enters the application
2. System will display different modes user can select on the drawer
3. User select unit conversion
4. Application loads the interface corresponding to the unit conversion
5. User enters the number to convert from.
6. User selects the unit from convert the number from
7. User selects the unit that the user wants to convert to.
8. Application makes a POST request to the third-party API, and receives the response back.
9. Application parses the response data to match the format that it needs.
10. Application displays the converted number.

4.1.4 Use Case: Randomizer

Diagram

Description: Team generator, decision maker, etc...

Actors: User

Preconditions:

- Stay connected to the internet in all time
- Users must be logged in.

Basic Path

1. User enters the application.
2. User goes through the authentication process.
3. User selects the randomizer mode.
4. Application renders a new interface.
5. User selects different types of the functionalities in the randomizer. (Team generator, decision maker, random picker, custom list)
6. User enters/pastes the list of entries to the textbox.
7. User clicks the generate button.
8. Application makes a POST request to the server, which will generate the value according to the type of the function selected.
9. Application receives, parses the response data.
10. Application displays the result.
11. User clicks save result.
    a. Application makes another POST request indicating to the server that the user wants to save this result to the history book.
    b. Server responses.
    c. Application displays saved success.
12. User chooses to export the data as an XLS file.
    a. Application makes one more POST request, requesting the server for the generated file.
    b. Server responses with the generated XLS file.
    c. Application prompts the user to choose the saving directory.
    d. User chooses the directory and clicks save.
    e. Application will save the file to that specified directory.