Spartan Snacks

Team Name: Bad Recursion (BRB)

Date: 12/09/2019

Team Members: Michael Calvo, Isaac Taylor,

Tammy Ogunkale, Harinder Badesha

Stakeholders: Ike Quigley, UNCG

Table of Contents

1. INTRODUCTION

1.1. Statement of Purpose

This is the System Requirements Document (SRD) for the SpartanSnacks project. It is designed to define the requirements of the project and it covers the work of the project. This document contains functional analysis of the program and the system requirements needed and is intended to undergo changes during the lifetime of development.

1.2. Intended Audience

The intended audience for this web application is for the faculty, staff, and students of the University of North Carolina at Greensboro, as well as all local inhabitants in the city of Greensboro, North Carolina.

1.3. Definitions/Jargons

- Spartan (from the website name; SpartanSnacks)- The official mascot for the University of North Carolina at Greensboro.

1.4. Project Scope

Primarily, the scope pertains to helping students decide the vast selection of local food choices in the city of Greensboro. We want to have a selection of cuisines type where the user chooses one or several cuisines and gets a list of restaurants to choose from. We want to make a website that is easy to navigate through and pleasing to look at and plan on using an external API to get information on restaurants in Greensboro and a database to store how many people use our website.

1.5. Technical Challenges

Only one member of the team is familiar with PHP, so it took some time for everyone in the group to learn PHP. Each team member was also new to implementing an API. Our original code did not factor in the possibility of the zomato API being shut down, so we had to modify it to have a translator class in case zomato was no longer working. Translator classes also have to be learned on the go in order to communicate with the Zomato API. Another challenge was getting the database to work as we had difficulty in starting it up and deciding what we wanted to store since there were many options.

1.6. References

References used in this project include:

- Unsplash.com
- Zomato API model

Unsplash.com was used to acquire free non-copyrighted images for our web-application. Two images were gotten from the website for this project. They were used for the header and footer of the website. The Zomato API model is being used for our external API that will give us information of the restaurants in the Greensboro area.

2. OVERALL DESCRIPTION

2.1. Product Features

This product will use an API that gives information and data on local restaurants, as well as use a database to store how many times the website is being used and what color theme our users pick for their results. This product will also display timed events where it will display cuisines of the day depending on what day it is. It also gives users the opportunity to pick what kind of cuisines they are feeling for the day from the checkboxes.

2.2. User Characteristics / Classes

The user will be able to choose the color for the theme of their results and that will be stored along with their UUID. There is also a new user search which is an object to allow the administrator store details of the user search.

2.3. Operating Environment

This website application will be fully operable on any operating system that supports web browsing. It will be functional on Windows, Mac, Desktop, Mobile, etc. and work on browsers such as Google Chrome, Safari, Firefox, Internet Explorer, etc.

2.4. Design Constraints

We want to store the amount of times any user uses our website(clicks the go button) via using their IP address so it's more accurate for different users, however implementing a design to store user's IP address is a constraint that we figured out how to do too late into the project. So it would be implemented later on in the maintenance stage after the website has been presented and launched. We also wanted to have distance be a filter for our users to choose, so the restaurants would be closest to them. However, this is a difficult task and will not be implemented into the project as getting an accurate user distance from the API was not feasible before the deadline of the project.

2.5. Assumptions / Dependencies

We are depending on our API and are assuming that it would still be functional in order for our project to work. We use the API to output restaurant names, addresses, and results. Based on this, we are assuming that the user lives in the Piedmont Triad area to use this website as it only gives out locations in greensboro. We are also assuming that the user knows english as there is no language translator feature on our website.

3. <u>FUNCTIONAL REQUIREMENTS</u>

3.1. Primary Functions

The primary function of this program is to take an API of restaurant information that includes names, address, cuisine type, and ratings and to traverse through the data and find a restaurant based on the cuisine type or rating that the user chooses.

3.2. Secondary Functions

The secondary functions include a database that will store the amount of times a user clicks the “go” button indicating our website has been used, and also store the color theme our users use. It will always use the API data to determine a “cuisine of the day” type for users to see by having set cuisines for each day of the week, and outputting the restaurants linked to that cuisine.

4. TECHNICAL REQUIREMENTS

4.1. Operating System / Compatibility

This project will be compatible with any device that supports web browsing on computer, laptop, or phone. It will be compatible with safari, Internet Explorer, firefox, and Google chrome. So in order to use the website you will need a device that can support web browsing and a network connection. Depending on users settings in google chrome, our website can look different than intended. So, in case our website looks different than intended, it's important for users to clear their cookies or clean their settings on Google Chrome.

4.2. Interface Requirements

4.2.1. User Interface

The user needs a laptop, a computer or a phone that supports web browsing to use our website. They can interact with the website using a mouse, a keyboard, and touchscreen on a smart device.

4.2.2. Hardware Interface

The user does not need any other hardware device to connect to our website other than what it is built on.

4.2.3. Software Interface

Software used for this project includes a Windows and Mac Operating System, a programming environment; Netbeans, in order to write the php language, a host for our website; XAMPP, and a SQL database which runs on phpmyadmin. GitHub and Git bash were also used for collaborating and sharing updates on code in order to make the software development process more efficient.

4.2.4. Communication Interface

Our website requires an API that holds information of restaurant names, cuisine types, ratings and address for a set location in order to function.

5. NONFUNCTIONAL REQUIREMENTS

5.1. Performance Requirements

The SpartanSnacks website can only handle 1000 calls to the API per day, so that limits the use. SpartanSnacks would not allow the user to proceed with its functionality without selecting at least one cuisine from the checkboxes. It also would not run without internet connection.

5.2. Safety/ Recovery

If the website crashes due to the zomato API not functioning anymore, it could be replaced with a new API which has to meet the requirements of our API adapter. Also, a corrupted database could cause a system failure. If that happens, a new database could be created and connected to the website using the adapter.

5.3. Security Requirements

No important user information is being stored, but as a security feature, the user object has a uuid for security in the database. It prevents people from knowing how the color chosen is being stored.

5.4. Policy Requirements

SpartanSnacks implements the policies on the zomato API website. And these policies apply to all the users of the SpartanSnacks website.

5.5. Software Quality Attributes

5.5.1. Availability

The website is always going to be available to users because it is hosted on 000webhost. It would only be down for maintenance and upgrade every six months. And the maintenance should only take a couple of hours.

5.5.2. Correctness

SpartanSnacks successfully performs its function of providing users with a list of restaurant options from an API based on their filters. It also provides the right link to the restaurant they eventually choose thanks to the API as well. On the back-end, it stores the amount of time the go button is clicked, as well as the user color selection, as is intended.

5.5.3. Maintainability

There are plans to do a maintenance of the website every six months, based on new features that might have to be added. The maintenance interval might also vary depending on how consistent and functional the Zomato API being used is.

5.5.4. Reusability

SpartanSnacks can be reused by other developers to gather more information about anything food related. It can be done by changing the API calls being made. It can also be used with a completely different API that meets the requirement of the Adapter.

5.5.5. Portability

The website should run on any platform and is available to users with an internet connection on any device. It is also able to run on any available server.

5.6. Process Requirements

5.6.1. Methodology

We used an Agile Development method for this project. There were sprint cycles and scrums which helped in making sure all the developers were up to date with the changes in the project, no matter what stage the project was in.

5.6.2. Time Constraints

We are constrained to complete the project by the end of UNCG's Fall 2019 Semester.

5.6.3. Cost and Delivery Date

There is zero cost for this project. Our project delivery date is planned for December 9th, 2019.