Software Requirements Specification

Recipe Finder by Team Recipe Bytes

COP4331	Fall	2013

Home Template

Modification History

Version	Date	Who	Comment
v1.0	09/10/2013	V. Marconi	Setup Document and Formatting.
v1.1	09/13/2013	V. Marconi	Filled Section 1 and developed Coding standards.
v1.2	09/14/2013	V. Marconi	Filled out Section 2 and assigned Ronald and Brian to fill out Event Table, and make Use Case Diagram.
v1.3	09/16/2013	V. Marconi	Filled out Section 3.
v1.4	09/16/2013	V. Marconi	Adjusted the Applicable Standards and Changed the Coding standards to our designed standard.
v1.5	09/18/2013	V. Marconi	Filled in Event Table and Use Case Diagram from Ronald and Brian's work.
v1.6	09/20/2013	V. Marconi	Reviewed and adjusted Section 3.
v1.7	09/21/2013	V. Marconi	Filled in Use Case Description.
v1.8	09/21/2013	V. Marconi	Adjusted Document Formatting.

Team Members

Name	Email	Number	Web-Page
Vincenzo Marconi	vincenzorm117@knights.ucf.edu	(954)778-0251	LINK
Rachel Kinner	rachel.kinner@knights.ucf.edu	(321)345-3215	LINK
Ronald Hyatt	enishi@knights.ucf.edu	(321)298-0459	LINK
Matt Bald	matt.e.bald@gmail.com	(941)914-4487	LINK
Brian McCormick	mccor140@knights.ucf.edu	(850)363-1052	LINK
Josh Barnett	jbarnett@knights.ucf.edu	(612)709-3144	LINK

Content of this Document

1: INTRODUCTION

Software to be Produced

Based on the epicurious food software, we are developing a recipe finder that appeals to college students. The application will provide a variety of of common college foods. The application will provide a food safety warnings against allergens. It will include a button search that enables users to narrow down their choices quickly into several classifications. The main filtering criteria will be the type of recipe to be produced categorized by the amount of time it takes to cook given by the categories: On-the-Go, Quick and Easy, and Chef. Finally, we are also adding an exclude feature to the search so users can get just the food they want. For more information see the Concept of Operations.

Reference Documents

- Concept of Operations
- Project Management Plan

Applicable Standards

- Html Coding Standard: chosen to maintain the same structure for all documents. Same colors to maintain the same flow.
- C# Coding Standard: Chosen so everybodies code is readable, easier to maintain, and it reduces the chance of producing errors.

Definitions, Acronyms, and Abbreviations

OS - Operating System HTML - Hyper Text Markup Language CSS - Cascading Style Scripts

2: PRODUCT OVERVIEW

Assumptions

- Availability: It is assumed that users will have access to a computer with a Windows OS.
- Capability: It is assumed that users have access to a computer with at least a Dual Core 2.33 Ghz Processor, including 1 GB RAM, and Windows 7

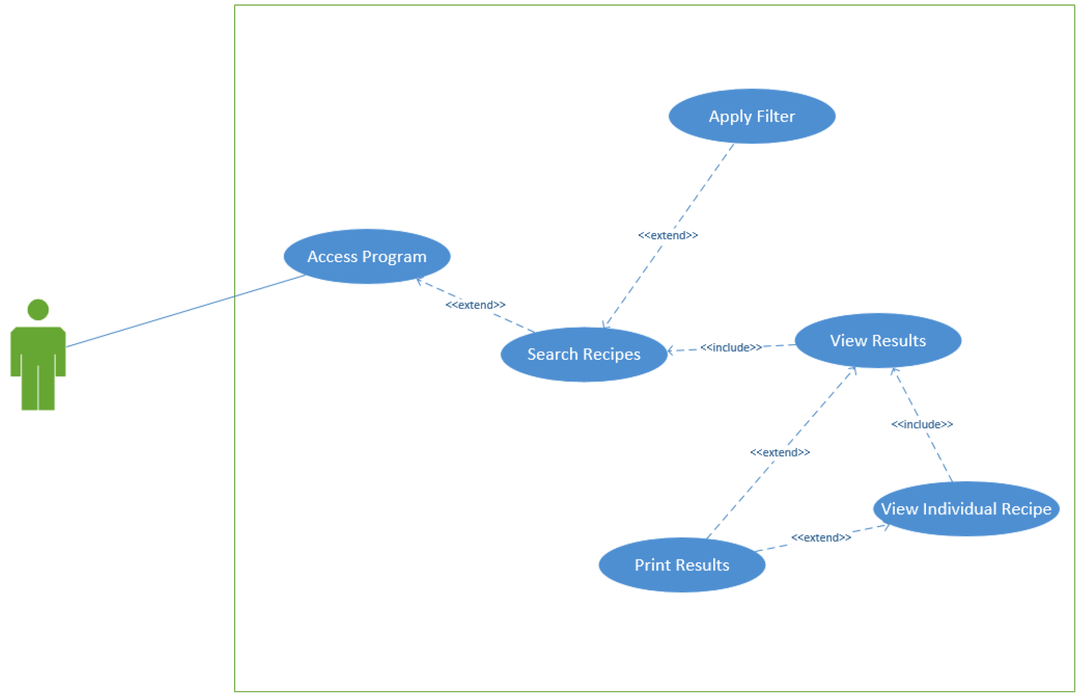
Stakeholders

- Developers: The developers have invested time and energy into creating the product.
- Busy College Student: The product will enable the busy student to have easy access to quick, on-the-go recipes.
- Health Concerned Student: The product will enable the health concerned student to have access to healthy recipes with detailed nutritional facts.
- Adventurous Student: The product will enable the adventurous student to have access to a wide variety of recipes.

Event Table

Event Name	External Stimuli	External Responses	Internal data and state
Select Ingredient	Mouse Click	Change In Checkbox Visual	Ingredient value set to True or False
Start Search	Mouse Click	Button Depresses	Filter's recipe results based on user's selections and displays results on tab/window
View Recipe	Double Click	New View Appears	Fetches selected recipes data from storage
Print to File	Mouse Click	File Dialog Box Appears	Creates new text file at save location for all selected recipes
Multi-Select Recipes	Mouse Click	Highlighting Line and Change in Checkbox Visual	Set selected value for selected recipes to true
Select Filters	Mouse Click	Change in Checkbox Visual	Filter value set to True or False
Program Startup	Double Click	Application Window Appears	Loads all recipe data before execution begins

Use Case Diagram



Use Case Descriptions

<u>Typical Use Cases</u>

- User opens application and makes a selection of the food he/she is looking for, and the application returns the results.
- User opens the application and then closes the application.
- User opens the application, runs a search, selects an individual recipe and prints the result.

<u>Some possible Exceptions</u>

- User runs a search with several selections and filters and the application crashes due to excessive use of memory and processing power.
- User goes to print result and the program fails to communicate with the printer software to print the file.

3: SPECIFIC REQUIREMENTS

Functional Requirements

- No validation or parameters required. All input is managed through selection of buttons.
- Appropriate results are presented when the corresponding selections are made; the correspondence being the selected ingredients and filters are matched to recipes.

Interface Requirements

- Data input: combination of button presses to select ingredients
- Data output: recipe from ingredient selections
- Domain: set of ingredients
- The data set should not be too specific. Each ingredient would be a generic of its species like Apple instead of Green Apple
- Each item sent will be very accurate for the program: its just a simple combinations permutations algorithm
- Ingredients list will be the most frequently accessed resource, then the recipes themselves would be second
- No timing issues will exist as the system is independent, it carries its resources, and the number of resources will be a small population.
- No data will sent or received, all data will exist within the program as it is stand-alone.
- Search data must be very accurate for the system to find its results.

Physical Environmental Requirements

Software must run on a platform supporting the Windows Operating System. All other external environmental concerns affect the computer, and not the software directly. The user just needs to keep his computer safe.

Users and Human Factors Requirements

- Support: The system should support all types of users who speak the English language
- Skill: The minimum amount skill required to operate the software is that of a 6 year-old. The program is moreso a food picker as it is a finder; the only operational skill required is button pressing and reading.
- The application will only provide visual stimuli, so the only accommodations provided to particular users would be the font-size and font color for those who might be visually impaired
- The system cannot be misused in any way, unless infected by a virus which at that point there is really nothing that can be done

Documentation Requirements

The level of documentation will be moderate (done by back-end coders) and it will be all online on the repo or the linked by the website. The user must have moderate to advance knowledge of C# mainly. Other programming documentation will not be too difficult to follow, and following these documents involves SE knowledge: which is a lot of common sense

Data Requirements

The only function used is a combinations function that matches button selections to recipes. The data retained will be the recipes, ingredients, nutritional facts, and health hazards. All of this data will be kept within the program as encrypted resources.

Resource Requirements

- Maintenance: With a Phased Life cycle model, developers will be developing and sustaining the application over time until project is terminated at which point maintenance will not be issued anymore.
- Space: This program will probably occupy at most 60 MB of space on the hard drive. Users will be notified so they can prepare their computers to have enough space.

- Performance: This program will use very little of the CPU so no air conditioning, heating, cooling, or power is required.
- Funding: Small budget because application is small little funding is required
- No hardware/software tools external to a Window platform will be required

Security Requirements

- Access: No authentication required. Program is free, freely accessible, and offers complete control
- If we have enough time: One of the implementations to add is a taste feature that based on your past recipe selections, the system will try to determine which foods you prefer
- The program thread will be ran from a user id and will only execute (no read and write)
- No direct precautions should be taken apart from the keeping your computers hardware and software safe like you normally would
- Recovery Plan: store backup executable so that if the program fails to run, the backup will be run

Quality Assurance Requirements

- Reliability: depends on the condition of its OS and computer hardware, and the accuracy of its recipes and nutritional facts
- Availability: depends on users ability to access the program either on their computer or a download link
- Maintainability: Since Phased Model is used maintenance will be reflected on the development stages
- Security: No concerns
- Portability: depends solely on portability of users laptop or access to a Windows platform
- Quality: Demonstrated on how effective recipes are in nourishing students and saving them time
- Faults: the system will have little faults as it will be constantly worked on, but faults will be isolated by good old fashion debugging and feedback
- No prescribed mean time between failures, failures will never show up on the production side, they will only show on the development side
- No prescribe time the system must be available; application is completely stand-alone so it has no dependencies