Project Name: Software Requirements Specification version 1.0

Team Name
Computer Science Department
California Polytechnic State University
San Luis Obispo, CA USA

October 10, 2017

Contents

Credits

Name	**Date**	**Role**	**Version**
Geraldo Macias	October 9, 2018	Lead Author of Other Nonfucn-tional Requirements	1.0
Matt Yarmolich	October 9, 2018	Lead Author of System Features	1.0

Revision History

Name	**Date**	**Reason for Changes**	**Version**
Geraldo Macias	October 8, 2018	Completion of Other Nonfunctional Requirements	1.0

1 Introduction

1.1 Purpose

Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.

1.2 Document Conventions

Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.

1.3 Intended Audience and Reading Suggestions

Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.

1.4 Project Scope

Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here. An SRS that specifies the next release of an evolving product should contain its own scope statement as a subset of the long-term strategic product vision.

1.5 References

List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.

2 Overall Description

2.1 Product Perspective

Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.

2.2 Product Features

Summarize the major features the product contains or the significant functions that it performs or lets the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or a class diagram, is often effective.

2.3 User Personas

Provide several fictitious profiles of individuals who might interact with your system. Each should give some context identifying their relevant skills and motivations for using the system. Each should be unique from the others to highlight different modes of use, reasons for use, and challenges.

2.4 User Classes and Characteristics

Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy. You might use a table like this:

User Class	**Description**
Employee	This is a description of an employee
Administrator	This is a description of an administrator

2.5 Operating Environment

Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.

2.6 Design and Implementation Constraints

Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).

2.7 User Documentation

List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.

2.8 Assumptions and Dependencies

List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).

2.9 Business Rules

List any business rules. Identify each one uniquely.

3 Use Cases

3.1 Use Case 1: Select Proper Classification Category

This use cate details the the path of the data analyst providing feedback to the machine learning algorithm written by Matt Yarmolich

Use Case ID:	1
Use Case Name:	Select Proper Classification Category
Created By:	Matt Yarmolich
Last Updated By:	Matt Yarmolich
Date Created:	October 9, 2018
Date Last Updated:	October 9, 2018
Actors:	Data Analyst
Description:	A Data Analyst accesses the data classification front end by feeding it information from an HTML-5 input tag and having the information parsed by the system. .
Preconditions:	1. the Data Analyst is logged into the system. 2. the Data Analyst is a registered user for the system. 3. the Data Analyst has a dataset that needs to be analyzed.
Postconditions:	1. the dataset has been correctly parsed by the data classifier 2. the classifier has outputted a variety of tags for verification by the Data Analyst. 3. the Data Analyst is capable of correctly identifying the dataset
Normal Flow:	1.0 Select the correct category from the data classifier

	1. The Data Analyst uploads the dataset to the front end of the system 2. System displays upload status 3. System displays upload complete and feeds the data to the backend 4. The backend classifies the initial column data 5. The backend classifies the rest of the data based on past data columns and previous data fed through the system 6. the backend outputs the results of the data in a web-parsable format (JSON) 7. The front end parses this output and displays it to the data analyst along with the columns in question 8. the Data Analyst reviews the data and selects the correct category based on their domain knowledge 9. The front end feeds their result back to the data classification backend 10. System displays the correct tagging of the data set
Alternative Flows:	1.1 No results from backend (branch after step) 1. The system outputs no previously learned categories 2. The Data Analyst inputs a new category for the data set 3. Return to step 8.
Exceptions:	1.0.E.1 Current time is after order cutoff time (at step 1) 1. System informs Patron that it's too late to place an order for today. 2a. Patron cancels the meal order. 2b. System terminates use case. 3a. Patron requests to select another date. 3b. System restarts use case.
Includes:	None
Priority:	High
Frequency of Use:	Approximately 400 users, average of one usage per day
Business Rules:	BR-1, BR-2, BR-3, BR-4, BR-8, BR-11, BR-12, BR-33

Special Requirements:	1. Patron shall be able to cancel the meal order at any time prior to confirming the order. 2. Patron shall be able to view all meals he ordered within the previous six months and repeat one of those meals as the new order, provided that all food items are available on the menu for the requested delivery date. (Priority = medium)
Assumptions:	Assume that 30 percent of Patrons will order the daily special (source: previous six months of cafeteria data).
Notes and Issues:	1. The default date is the current date if the Patron is using the system before today's order cutoff time. Otherwise, the default date is the next day that the cafeteria is open. 2. If Patron doesn't want to have the meal delivered, the precondition requiring registration for payroll deduction is not applicable. 3. Peak usage load for this use case is between 8:00am and 10:00am local time.

3.2 Use Case 2: Whatever the next one is

This is a casual use-case. Note that there is a label in the LaTeX so you can refer to it as being in section 3.2 on page 8.

1. This is step one.
2. This is step two.
3. This is step three.

4 System Features

This system will be primarily used for data classification and adding meta-tags/titles to unidentified datasets inputted by a user from various formats including CSVs, JSON, and SQL. The main purpose of this product is to give Data Scientists, Data Engineers, and Data Analysis's insight on what kind of data they are collecting, and the ability to verify or modify any tags generated by the machine learning processes in our backend. This will be done through a front end written for the web interfacing with most modern browsers with javascript enabled. This front end will be powered by a backend written in Python leveraging a machine learning library such as TensorFlow. This backend will take data in from a recognized data format, parse the data into a data structure for temporary storage, feed it into this library and spit out possible classification tags. These tags will then be sent to the front end for verification by the data analyst for verification/modification. This data analyst will be able to see the data in question in order to judge the classifiers accuracy and to teach it about new datasets/data types by piping these recommendations back to the machine learning algorithm.

4.1 Machine Learning Python Backend

4.1.1 Description and Priority

This feature of the product will allow data being fed into the product to produce meaningful data for output/viewing by the data analyst. The benefit of this algorithm is that it will provide data to the front end for easy viewing and provide an endpoint for where feedback from the data analysts. In terms of cost, this is probably going to be a fairly expensive feature to implement as its the main focus of this project and without it, the rest of the project is useless. However there are a ton of resources for implementing this feature minimizing the potential development time required. Finally, this system requirement is of High priority due to the fact it is the main feature that makes our product useful.

4.1.2 Stimulus/Response Sequences

The system shall take in a dataset from the front end and then feed this dataset into the machine learning algorithm column line by line. This step will be done by the data parser which will read in the columns line by line and separate them (detailed below). This data will then be compared to previous inputs taught by the algorithm by the Software Engineers developing the program. With these comparisons, the algorithm shall develop a result that fits the dataset, or output unknown if it doesnt have any idea what the dataset is of. It also will look to previous classifications contained by the dataset

4.1.3 Functional Requirements

Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

1. REQ-1: The system shall be able to take in a specific data column and be able to identify it based on previous machine learning techniques used on the system and

output a string of what it thinks it should output .

2. REQ-2: The system shall be able to learn new classifications given to it by the Data Analysts as feedback from the front end.

3. REQ-3: The system shall be able to learn categories over time as training is provided to it by the Software Engineers developing the system, and by data analysts feeding it new categories/training.

4.2 Data Classification Front End

4.2.1 Description and Priority

This feature set will allow the data analyst to interact with the output of the machine learning algorithm for further classification and viewing of what the algorithm outputs. This feature set is another high priority feature as it allows our algorithm to be interacted with by the client and provide meaningful responses to help further teach the algorithm about new datasets and expectations.

4.2.2 Stimulus/Response Sequences

List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.

4.2.3 Functional Requirements

Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

1. REQ-1: WIP

2. REQ-2: WIP

3. REQ-3: WIP

5 External Interface Requirements

5.1 User Interfaces

Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.

5.2 Hardware Interfaces

Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.

5.3 Software Interfaces

Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

5.4 Communications Interfaces

Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

6 Other Nonfunctional Requirements

6.1 Performance Requirements

The customer has specified that performance is not a concern.

6.2 Safety Requirements

During the classification stage, safety is not a concern. After the catalog phase we will begin redacting sensitive information.

6.3 Security Requirements

The software will have two factor authentication and will only allow verified users to access the systems databases.

6.4 Software Quality Attributes

1. The system shall have a learning curve of 24 hours. This applies to all users ranging from a sports analyst to a data scientist.

2. The system shall be available 99.99% of the time.

3. The system shall be able to maintain one million datasets.

4. The system will be portable so different companies can use our solution.

7 Other Requirements

Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.

A Glossary

Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.

B Analysis Models

Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.

C Issues List

This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.