Requirement Analysis Document (RAD)

Project: Data Labeling System

Author: CSE 3063 - Java Project / Group 6
Date: 27/11/2020
Version: 1.4

Purpose of the Document

This document mainly discusses the functionalities of the proposed system and its advantages over the existing system. An overview of the proposed system and its functional and nonfunctional requirements demanded by the stakeholders are provided in the document. Use Cases, Sequence Diagram of the proposed system are also included to illustrate the situation or condition of the proposed system being used by the users.

This requirement analysis document is a very efficient artifact for project iterations. It helps to understand the requirements of project which, in this case, is a data labeling system to solve real-world problems.

TABLE OF CONTENTS

Version Control

Version	Prepared By	Date	Description
1.0	Melik Çağan Oduncuoğlu	27/11/2020	First Version (template)
1.1	Can Karatepe	27/11/2020	Minor changes
1.2	Melik Çağan Oduncuoğlu	28/11/2020	Use case diagram updated and useless parts removed.
1.3	Melik Çağan Oduncuoğlu Can Karatepe	03/12/2020	Added System Domain Model and System Sequence Diagram; updated Use Case Diagram and Glossary.
1.4	Melik Çağan Oduncuoğlu	05/12/2020	Document format has changed.
2.0	Melik Çağan Oduncuoğlu Can Karatepe	16/12/2020	Added new requirements for iteration 2.
2.1	Can Karatepe Melik Çağan Oduncuoğlu	24/12/2020	Updated System Domain Model.
3.0	Melik Çağan Oduncuoğlu Can Karatepe	02/01/2021	Added new requirements for iteration 3 and updated use case and system sequence diagram.
3.1	Melik Çağan Oduncuoğlu	02/01/2021	Use case diagram updated.

Description

1. Brief description of the system

Data labeling is the process of assigning one of the several predetermined labels (a.k.a. class labels, categories, tags) to a group of instances (a.k.a. samples, examples, records and documents) via a user interface by human experts. A group of instances are known as a dataset.

2. Purpose of the system

The purpose of the system is to provide its users a mechanism to work on predefined datasets. These datasets are written in the JSON format and include instances which can be labeled by users, and the labels which can be used to label said instances. This mechanism will then be used to create tagged/labeled datasets which can be analyzed to train artificial intelligence software or make predictions about the data. For example, the system can be used to classify customer comments as positive or negative in an e-commerce website.

3. System variety

In some datasets an instance may labeled with only a single class label. In other datasets a single instance can be assigned more than one class labels. This will be given as an input along with the dataset and a set of class labels that are available for labeling.

In some datasets the whole text is labeled such as in sentiment classification. In others, words, or phrases inside an instance (document) can be labeled such as in Named Entity Recognition problem.

Functional Requirements

Requirement Name	Requirement Statement	Must/Want	Comments
Extensibility of code	We do not have all requirements in the first iteration. Therefore, extensibility is important.	Must	
Multi-user	System will be multi-user supported.	Must	Adding new users to the program should be done easily and quickly.
Labeling instances	A user can label many instances	Must	
Multi-user labeling	An instance can be labeled by one or more users (possibly with different class labels).	Must	
Consistency Checking	The system should present the user with instances they already labeled, based on a probability set in a configuration file. Afterwards, the system should check how consistent the user is with their labeling.	Must	
Labeling mechanism	For the first two iterations, the labeling mechanism will be Random Labeling Mechanism	Must	It will randomly choose one of the labels and assigns it to instance.
Multi-mechanism support	Code must support easily pluggable labeling mechanisms.	Must	Polymorphism and related design patterns will be useful.
Number of Labels	In the dataset file there will be information about max number of labels. If max. number of labels > 1, instances can have multiple labels.	Must	
Scenarios support	System must support scenarios in which words/terms in an instance can also be labeled separately.	Want	This does not have to be necessarily implemented in first iteration, but it should be extensible for next iterations, so we may be able to add that functionality with min. changes.
Metrics Reporting Functionality	System should calculate metrics about its users, its datasets, and the instances of those datasets. These metrics then should be reported to the user by console or file output.	Must	

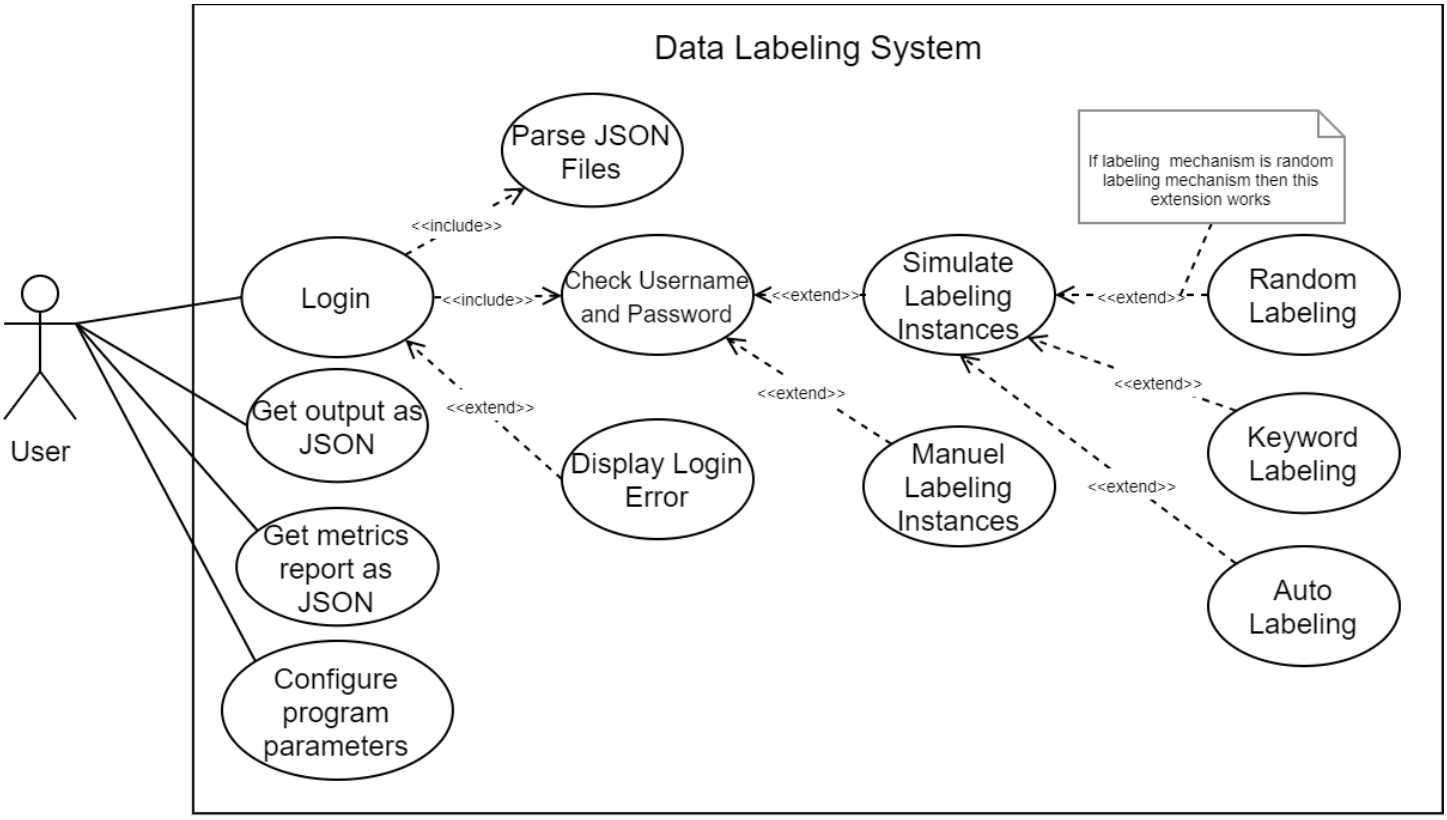
Functional Requirements (cont.)

Requirement Name	Requirement Statement	Must/Want	Comments
Realtime Metric Reports	The calculated metrics should be updated and written on each change, so even if the program ends abruptly, the reports can be read and analyzed.	Must	
User Interface	In addition to bot users, human users should be able to interact with this system. To do that simple command line user interface must be implemented.	Must	
User Login	When the software is ran, system should ask user to username and password. If username and password are left blank, then a bot mechanism will run the system.	Must	

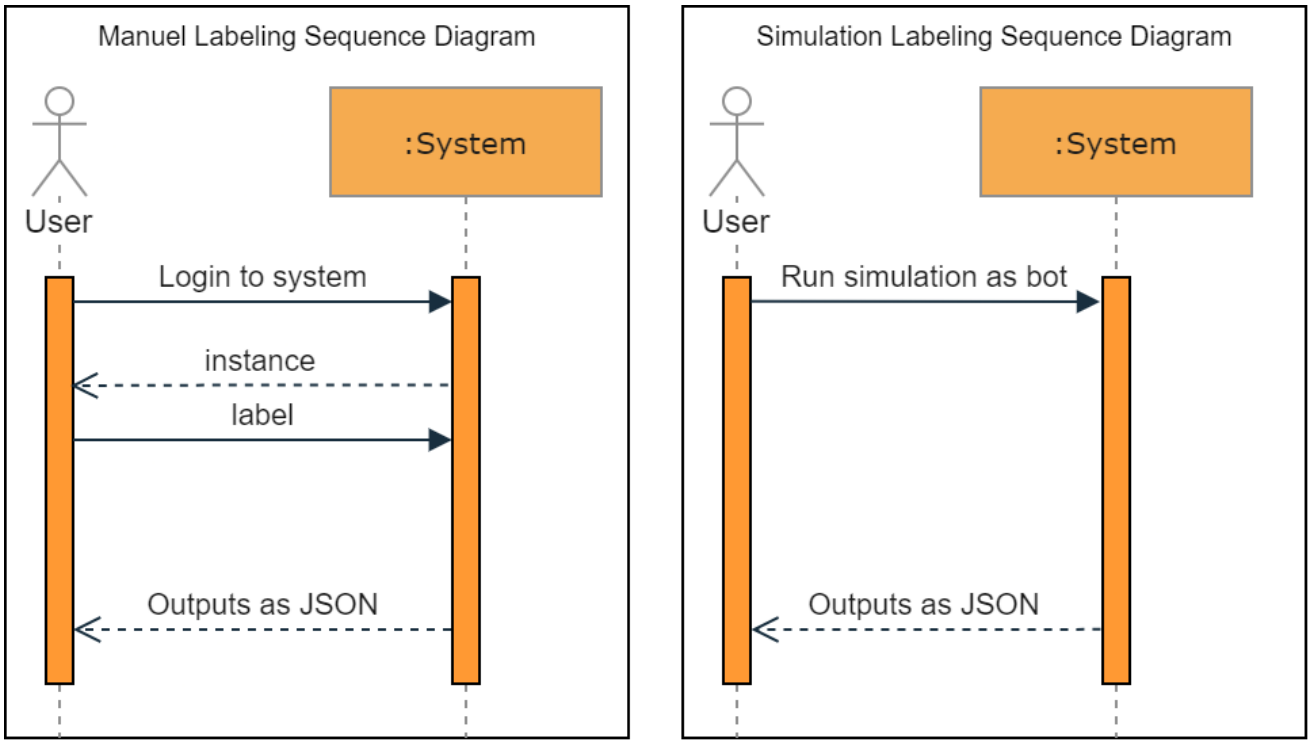
Non-Functional Requirements

Requirement Name	Requirement Statement	Must/Want	Comments
Java	Coding must be done in java	Must	
Store Data	No databases allowed. All data must be in json files	Must	
GitHub	Opening a GitHub repository.	Must	Done.
Open folder for each set of iteration artifacts	Open a folder like “Iteration 1” and store iteration artifacts in that folder	Must	
Requirement Analysis Document (RAD)	Brief description about project, glossary, and list of functional and non-functional requirements. Add domain model and optionally an SSD.	Must	In progress.
Design Class Diagram (DCD)	An UML class diagram showing domain classes.	Must	
Design Sequence Diagrams (DSD)	UML sequence diagram(s) showing interactions between software objects of system.	Must	
Java Code	Includes classes. Each class must be in a separate .java file named with the name of the class.	Must	Repo size should not exceed 10mb. (java files, json files and necessary libraries)
Project Evaluation Metrics	This is a spreadsheet providing your measurements for the metrics. The metrics will be given as columns.	Must	“CSE3063 Fall2020 Java Project Evaluation Metrics” file as example.

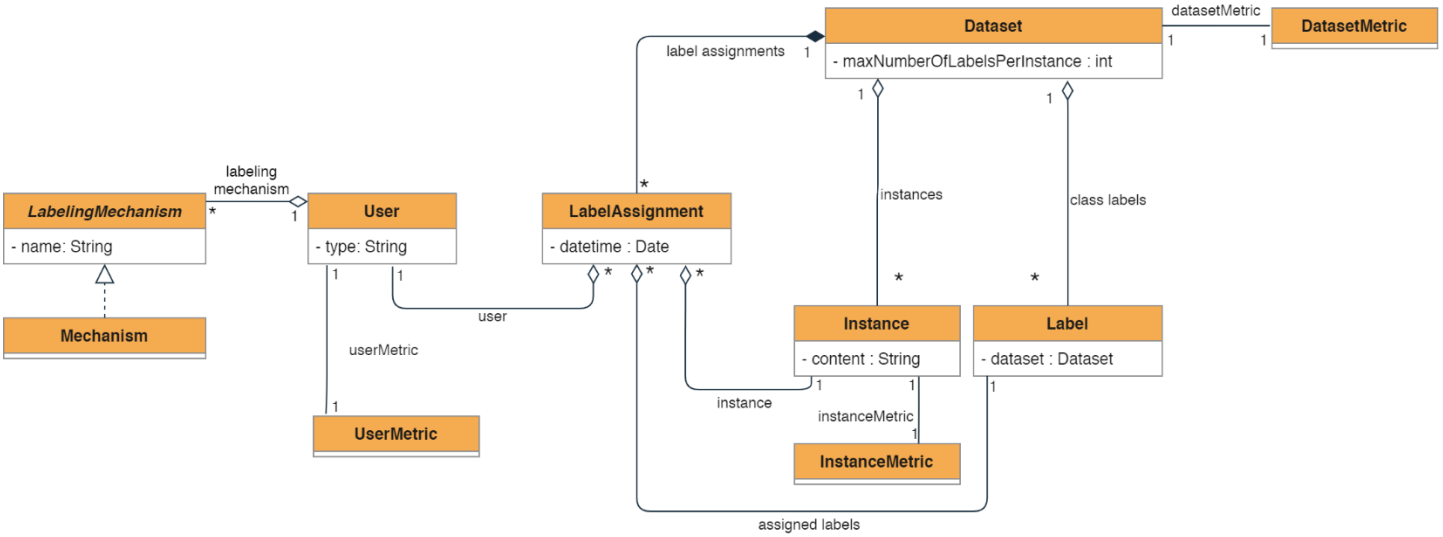
Use Case Diagram



System Sequence Diagram



System Domain Model



Glossary

Actor: Any person who will be using the system

Domain Model: A domain model is a conceptual model of the domain that incorporates both behavior and data.

Functional Requirement: Interactions between the system and its environment independent of its implementation.

JSON File: A JSON file is a file that stores simple data structures and objects in JavaScript Object Notation (JSON) format.

Non-Functional Requirement: User-visible aspects of the system that are not directly related with the functional behavior of the system.

Sequence Diagram: Behavior of a use case (or scenario) that is distributed among its participating objects.

Use Cases: Abstraction that describes a class of scenarios.