Project Name:
Software Architecture
Version 2.0

Data Plumbers

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

December 5, 2018

Contents

Credits

Name	**Date**	**Role**	**Version**
Tony Chen	December 05, 2018	Document Owner	2.0
Zachary Richardson	December 05, 2018	Document Owner	2.0
Bennett Robinson	December 05, 2018	Document Owner	2.0
Kyler Ramsey	December 05, 2018	Document Owner	2.0
Samuel Nayerman	December 05, 2018	Document Owner	2.0

Revision History

Name	Date	Reason for Changes	Version
Tony Chen	November 03, 2018	Initial document created.	1.0
Tony Chen	November 05, 2018	Added sequence diagram for section 3.3.1.	1.0
Kyler Ramsey	November 06, 2018	Added deployment diagram for section 3.2.1. Added sequence diagram for section 3.3.2.	1.0
Tony Chen	November 06, 2018	Added dialog map for section 3.2.2 and Introduction.	1.0
Zachary Richardson	November 07, 2018	Added diagrams for section 3.3.2, 3.3.3.	1.0
Bennett Robinson	November 07, 2018	Added activity diagrams for section 3.2.3 and 3.3.5.	1.0
Samuel Nayerman	November 07, 2018	Added activity diagrams for section 3.3.6	1.0
Zachary Richardson	December 05, 2018	Added Problem Description and Solution summary.	2.0
Tony Chen	December 05, 2018	Added description to Dialog Map and section 3.3.2. Changed Diagram Map based on prof's feedback. Added a bit more to Solution Overview. Added Testing Process and Test Example subsextions.	2.0
Kyler Ramsey	December 05, 2018	Added description for test section. Added descriptions for sequence and deployment diagrams	2.0
Bennett Robinson	December 05, 2018	Added description for adding/ modifying classification(s).	2.0

1 Introduction

This document summarizes the functionality covered by the design and scope of the design within the Data Discovery Workbench, which is focused solely on the Data Classifier component. For more information please see the Vision and Scope and SRS documents.

The Data Classifier system will run on a web server along with two application servers, one responsible for the ML classification aspect and the other as the backend layer for our MongoDB database. The web browser will serve as the main client for the UI.

2 Problem Description

As defined by MarkLogic, our system will allow a user (typically a Data Scientist) to classify and organize data within multiple datasets. This data may come in various format and file types.

3 Solution

Our system uses a machine learning component to classify and analyze data sets. Coupled with an intuitive user interface, this system shall allow user's to interact with and learn from their data classifications.

3.1 Overview

The solution is comprised of three primary features: uploading dataset, viewing and editting data classifications, and exporting the data classifications. The application interfaces will be built with simplicity in mind so that users can easily navigate and use our application without discoverability issues. These features will all be made available through an interactive web application. User data will be protected through a user authentication process before being able to access our application.

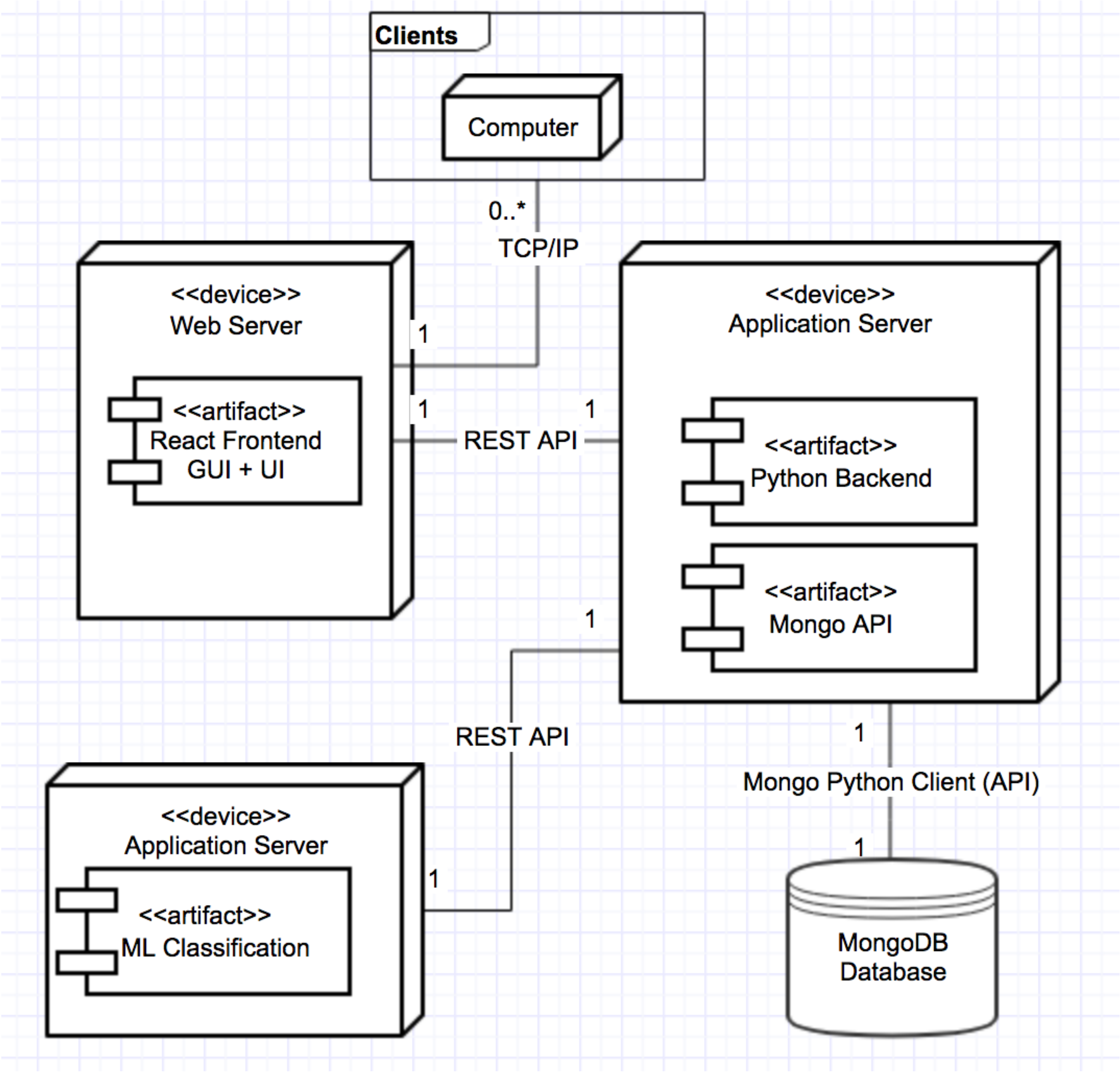
The system shall be divided into four components: React frontend web server, Python backend server, machine learning classifier, and persistent database.

3.2 Components

The frontend web server shall allow users to interact with the system, presenting options and useful information. The backend server shall process a user's requests, interacting with both the database and classifier to retrieve the required information. The classifier will rely upon machine learning models and libraries to classify datasets fed into it by the backend. The database will be used to persistently store information required by the system for future usage.

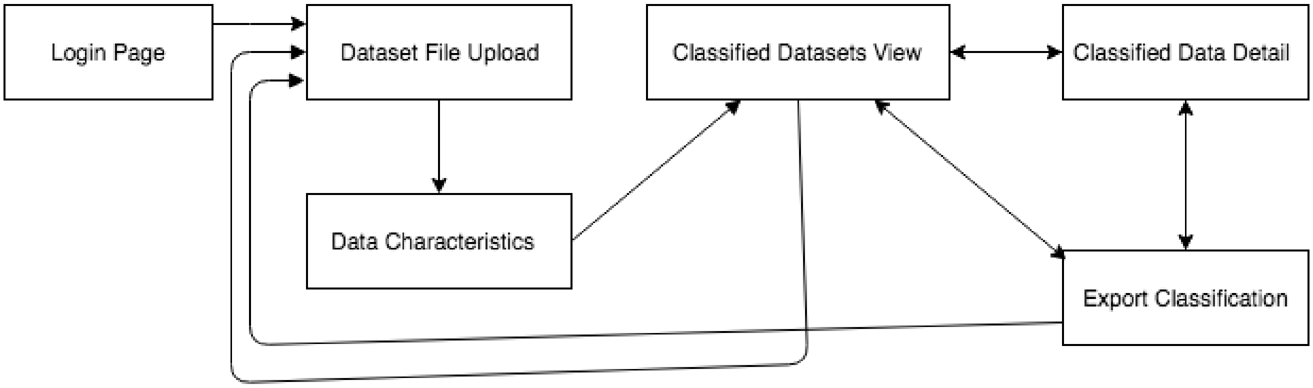
3.2.1 Deployment Diagram: Software Deployment

Created by Kyler Ramsey



3.2.2 Dialog Map: Software Interface Flow

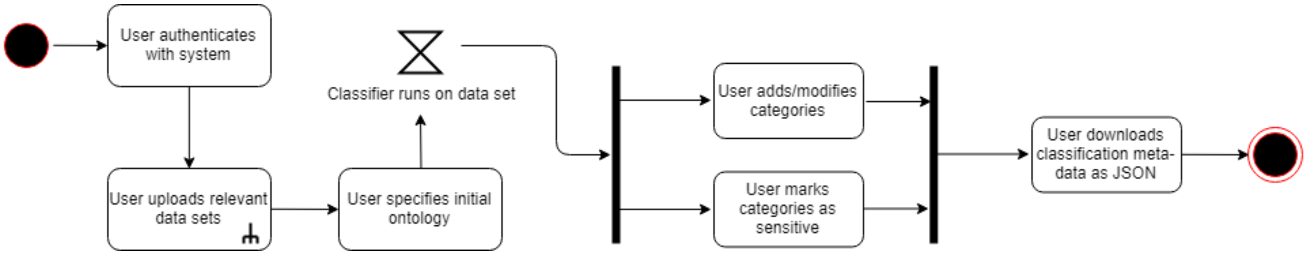
Created by Tony Chen



The Dialog Map depicts the general user workflow between the software interfaces in our Data Classifier web application. The user will have to enter our application through the Login Page authentication process before being able to be granted access. The user will then be able to start uploading and categorizing/defining their datasets to the ML to be processed. After ML learns about the data ontology and dataset from the user, it will attempt display a set of classifications to the Classified Datasets View where the user is able to view and edit their classification categories pertaining to a dataset. The user will also have the ability to export their classifications to a JSON file after data classification is complete.

3.2.3 Activity Overview: Generalized User Flow

Created by Bennett Robinson

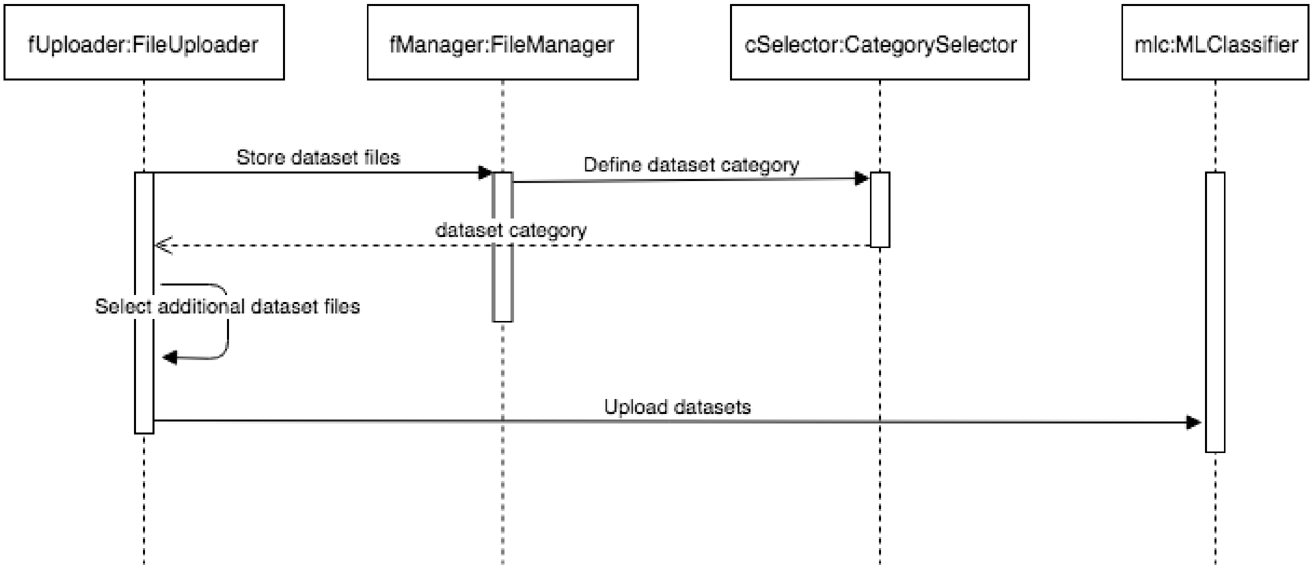

User authenticates with system
User uploads relevant data sets
User specifies initial ontology
Classifier runs on data set
User adds/modifies categories
User marks categories as sensitive
User downloads classification meta-data as JSON

This diagram covers the general user flow throughout our program. Each box represents a different use case and displays what use cases are accessible after the completion of the current use case. The user begins by authenticating with the system. After, the user is able to upload relevant datasets and specify the initial classification ontology. Once the classifier runs on the data set, the user is presented with two use cases. They can either add or modify categories, or mark select categories as sensitive. Finally, the user is then able to export the classification metadata as a JSON file.

3.3 Design

3.3.1 Sequence Diagram: Upload Dataset Files

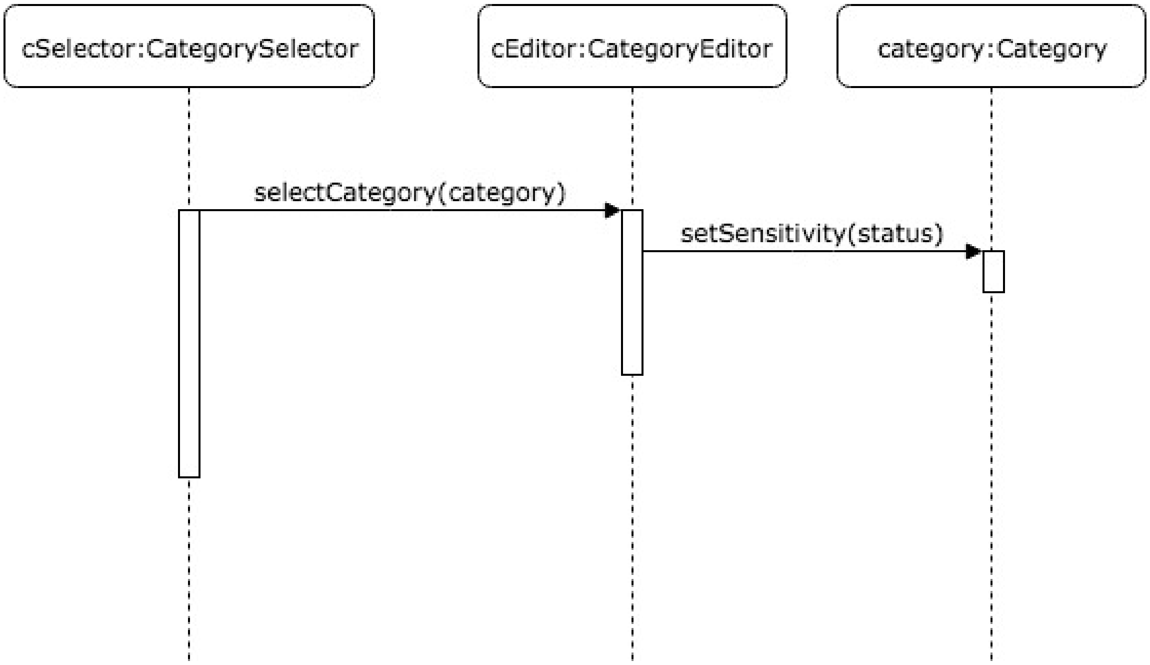
Created by Tony Chen



The user will be presented with the file uploader as an entry point to the general workflow of the application for uploading their datasets. Once the user uploads their dataset files, the user will then be presented with a data characteristics interface where the user will define their data ontology so that machine learning can use that information, along with the dataset's meta-data, for the classification process. The user will also be given another chance to upload additional dataset files along the way before the datasets and their data characteristics info gets sent to the backend server from the web application.

3.3.2 Sequence Diagram: Label Category As Sensitive

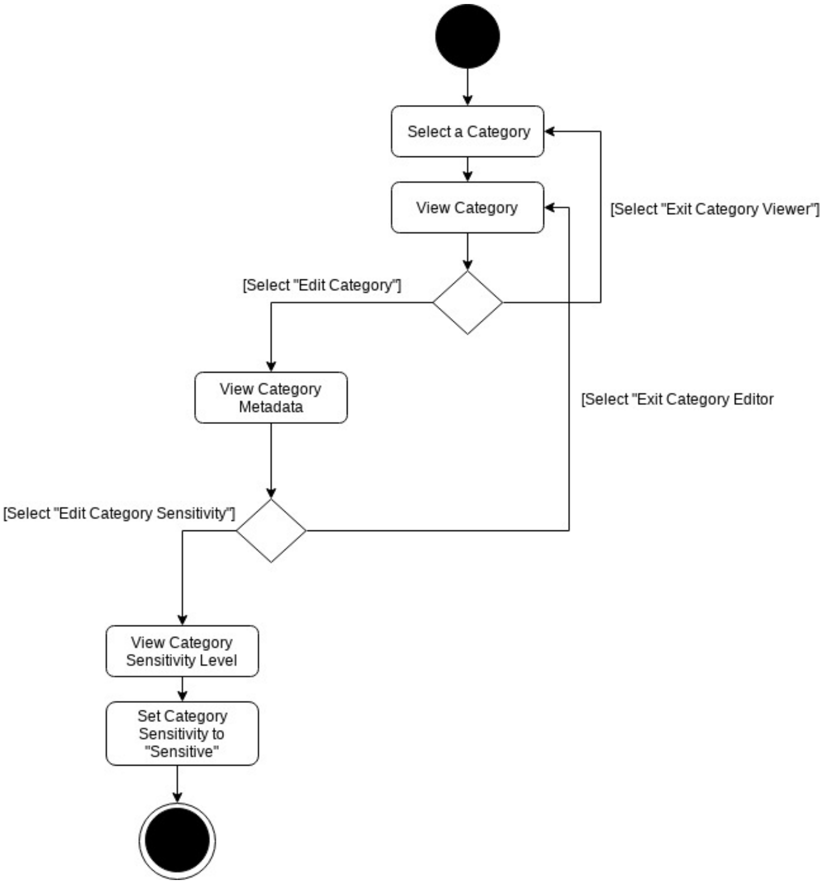
Created by Zachary Richardson



The above diagram shows the simple flow of actions allowing a user to label a category as sensitive. This is a high level diagram of the same process shown below.

3.3.3 Activity Diagram: Label Category As Sensitive

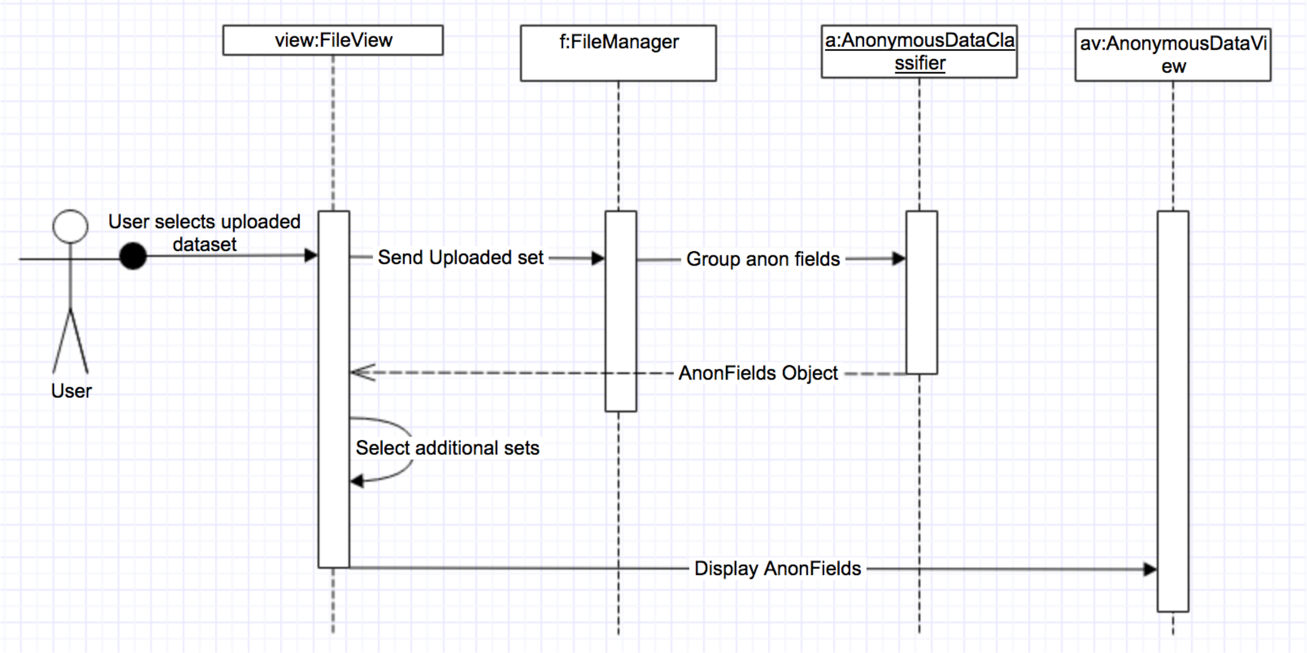
Created by Zachary Richardson


Select a Category
View Category
[Select "Exit Category Viewer"]
[Select "Edit Category"]
View Category Metadata
[Select "Exit Category Editor
[Select "Edit Category Sensitivity"]
View Category Sensitivity Level
Set Category Sensitivity to "Sensitive"

This diagram is meant to show the sequence of actions taken by a user while labeling a given category as sensitive. Upon viewing available categories, the user will select a specific category they wish to label as sensitive. This will launch a screen giving different options the user can edit for the selected data set, including a toggle for the category's sensitivity. At any point during the process, the user may exit the menu and return to the previous screen.

3.3.4 Sequence Diagram: View Anonymous Categories

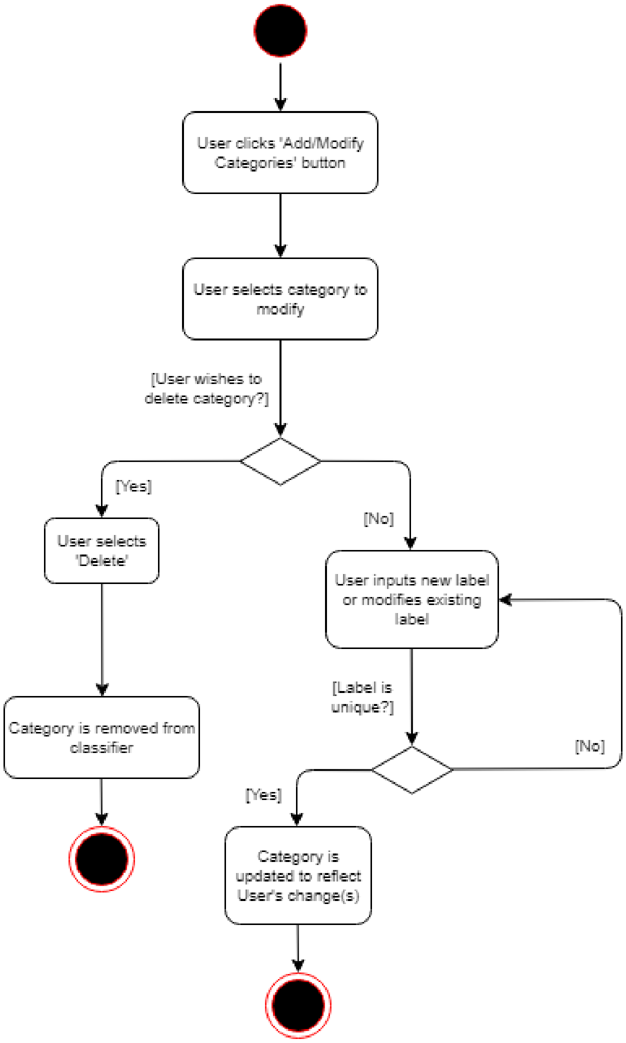
Created by Kyler Ramsey



This is a sequence diagram that covers the use case of viewing categories classified as anonymous. The user is able to select multiple fields (asynchronously) to be categorized as anonymous before the view updates to reflect the additions.

3.3.5 Activity Diagram: Add or Modify Categories

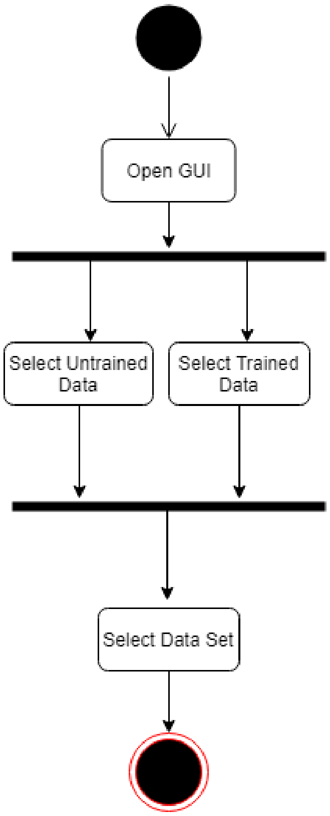
Created by Bennett Robinson


User clicks 'Add/Modify Categories' button
User selects category to modify
[User wishes to delete category?]
[Yes]
[No]
User selects 'Delete'
User inputs new label or modifies existing label
Category is removed from classifier
[Label is unique?]
[No]
[Yes]
Category is updated to reflect User's change(s)

This is an activity diagram covering the use case of adding or modifying categories. It shows the possible states and paths for the user to follow during the use case. As shown in the diagram, once a user clicks 'Add/Modify Categories', they may select a category to modify. If they choose to delete a category, they select which category (or categories) to delete, and the selected categories are removed from the model. If the user instead chooses to modify or add additional categories, they may input new labels or modify existing ones. If the new or modified labels are determined to be unique from existing labels, the model is updated to reflect the additions or changes.

3.3.6 Activity Diagram: View Dataset

Created by Samuel Nayerman



4 Test

The backend (Python/Flask) portion of our software application will incorporate Travis Continuous Integration and use the testing suite PyTest as a test harness for testing our server routes. The frontend (React) portion of the system will incorporate Mocha for unit testing. We intend to follow the Agile proposed TDD method of development, where tests are written before development. In addition, code that will be pulled to the main production branch of our repositories will first have to be reviewed by other team members through a formal Pull Request on Github in order to maintain high quality code throughout the development phase.

4.1 Testing Process

The developers who wrote their own module of code are responsible for creating unit tests for testing new code functionality before being allowed to be pushed onto production. The same principle also applies to modifying the existing codebase. For example, adding new functionality on top of a module. All testing suites must pass before handing it off for production.

4.2 Example Tests

4.2.1 Acceptance Testing

Story: As a data scientist, I always see the data classifications displayed in my account whenever I upload large datasets so that I can easily analyze and be able to edit and export the classifications for external usage.

1. Verify that data classifications are placed in the correct set of containers holding the dataset.

2. Verify the classification of the dataset are correctly updated when dataset ontology gets edited.

3. Verify classified datasets can be readily exported on demand once its saved to the database.

4. Verify the info/messages when there is an error in classifying datasets.

5 Issues

We still have to consider how we will deploy our application in production and what types of servers will be running our software. Currently, most groups are gravitating towards Amazon S3 for storage since we are given free AWS credits. Moving forward, we need to make architectural changes to accommodate deployment in an Amazon AWS/S3 instance.

A Glossary

- ML - Machine Learning

- UI - User Interface