****

**DUX**

**Data Utility with eXcellence**

*Your personal Data Scientist*

Software Requirement Specification

Session: 2019-2020

CSE 339

***Team name Creative Engine***

Nikhil JSK 11601789

Prasanth G. 11601526

Anil Raj T. 11601794

Raja Rahul M. 11601793

**Table of Contents**

1. Introduction 4

1.1 Purpose 4

1.2 Description 4

1.3 Scope 5

1.4 Overview 5

1.5 Definitions, Acronyms and Abbreviations 6

1.6 Technologies to be used 7

2. Overall Description 8

2.1 Use-Case Model 8

2.2 Web Architecture diagram 9

2.3 ER Diagram 9

2.4 Workflow description 10

2.5 Assumptions and Dependencies 12

3. Specific Requirements 12

3.1 Supplementary Requirements 12

3.2 Non-Functional Requirements 13

4. System Evolution 14

5. References 15

**Table of Figures**

1. Use case model 8

2. Web Architecture 9

3. ER Diagram 9

4. Upload Segment Workflow 10

5. Process Segment Workflow 10

6. Visualize Segment Workflow 11

7. Model Segment Workflow 11

**1. Introduction**

* 1. *Purpose*

To extremely stream-line the process of Data Analysis in the field of Data science. Helps every data-enthusiast to pre-process the data, visualize, gain insights of the data and also train a machine learning model (depending on the type of data provided) just while communicating with *DUX* in high-level language (English). This tool is highly influential to prove that Data Science is highly powerful, simultaneously demonstrating how easy it is to use Data to produce significant and critical results for any domain in a global perspective. This tool will prove that analysis of any given data can be done ***without*** a single line of code, also no local dependencies

* 1. *Description*

A REST-API service integrated with a user-friendly and compelling UI where the user can upload the data, communicate with the assistant in a high-level language (English), use different graphs to visualize the data points, gain insights about the data in-hand, and train a basic model. And, all of that is done only through typing the text in the tool and not a single line of code, where real-time processing is done in the backend. These are the basic four steps in a conventional data science field, all streamlined to achieve the best productivity along with ease of analyzing the data.

* 1. *Scope*
* Maintain a global perspective
* Make data science compelling
* Operating System independent.
* Useful for amateur data scientists
* Easy access to tools in data science
* Text-based preprocessing, visualization and model building
* Streamline the process of data analysis to the greatest extent
* Train a holistic model without any local-installed libraries
* A viable solution which serves as a foundation for solving global problems**­­­**
* Targeting multiple levels of user types, newbies, beginners, amateurs, professionals in Data science field.

*1.4 Overview*

1. The user is asked to upload a dataset.
2. The user is then prompted to choose between Pre-processing, Visualization or Model building.
3. Choosing pre-process, the user can communicate in the form of text with DUX data assistant which is basically a REST API service.
4. Pre-processing enables user to write text to remove null values, nan values, drop columns in a dataframe, data type conversion, and also encode the string or categorical features to make processing easy for machine learning models. He can do some basic math calculations like finding the minimum, maximum, mean, standard deviation, variance, skewness and kurtosis of any give column.
5. Choosing visualization, the user is directly taken to the visualizations segment where the he can communicate with the assistant to utilize features of different graph and features in Seaborn and Matplotlib library. Simple text is converted to visualizations of the data.
6. The user can also upload an image and select the text in the image and then make a query based on the data.
7. Gaining insights about the data is an integral part of data analytics, this part is included in the visualization segment of DUX. The user is given the inference from the graph.
8. Handling of graphs might be straight forward, if it is about handling direct data columns. But the user can also ask for the correlation between two different columns along with the variance of the third.
9. Choosing model building, the user is given recommendation on the type of machine learning model to be applied on the dataset.
10. Once the user selects between Regression, Classification, Clustering, Natural Language processing, the user is then given the recommendation on the sub-type or the algorithm to be used on the dataset to get the best accuracy or efficiency.
11. Once the training is done, metrics related to the training are displayed. The user is given an option to download the trained model to save loads of time. He can then use the saved model later to get predictions of related data.

*1.5 Definitions, Acronyms and Abbreviations*

1. Data Science – A multi-disciplinary field which uses algorithms and math to extract insights from data.
2. Data Analytics – The process of inspecting, cleansing, transforming of data and modeling.
3. Regression – A supervised machine learning for prediction continuous values
4. Classification – A supervised machine learning model for classifying multi-label or multi-class data.
5. Clustering – An unsupervised machine learning model for grouping similar data points together
6. NLP – Natural Language Processing – Subfield of AI to understand the interactions between human and machines
   1. *Technologies and libraries to be used*

* Python
* OpenCV
* TensorFlow
* Flask
* Flask Restful
* NLTK
* Sklearn
* SciPy
* Seaborn
* Matplotlib
* Keras
* Feature Extraction tool
* React js
* Node js
* Firebase
* API calls (Requests and Fetch)
* Custom library for text highlight.

*2.1 Use-Case Model*

* The use-case model shows the features and options for user, from frontend perspective.
* Five different segments
  + Upload Dataset/Image
  + Load a trained model
  + Pre-processing
  + Visualization
  + Model Training
* User has the entire control over the program. There is no admin.
  1. *Web Architecture diagram*

DATABASE

Process Invoke

REST CALL

Firebase

Python Flask API

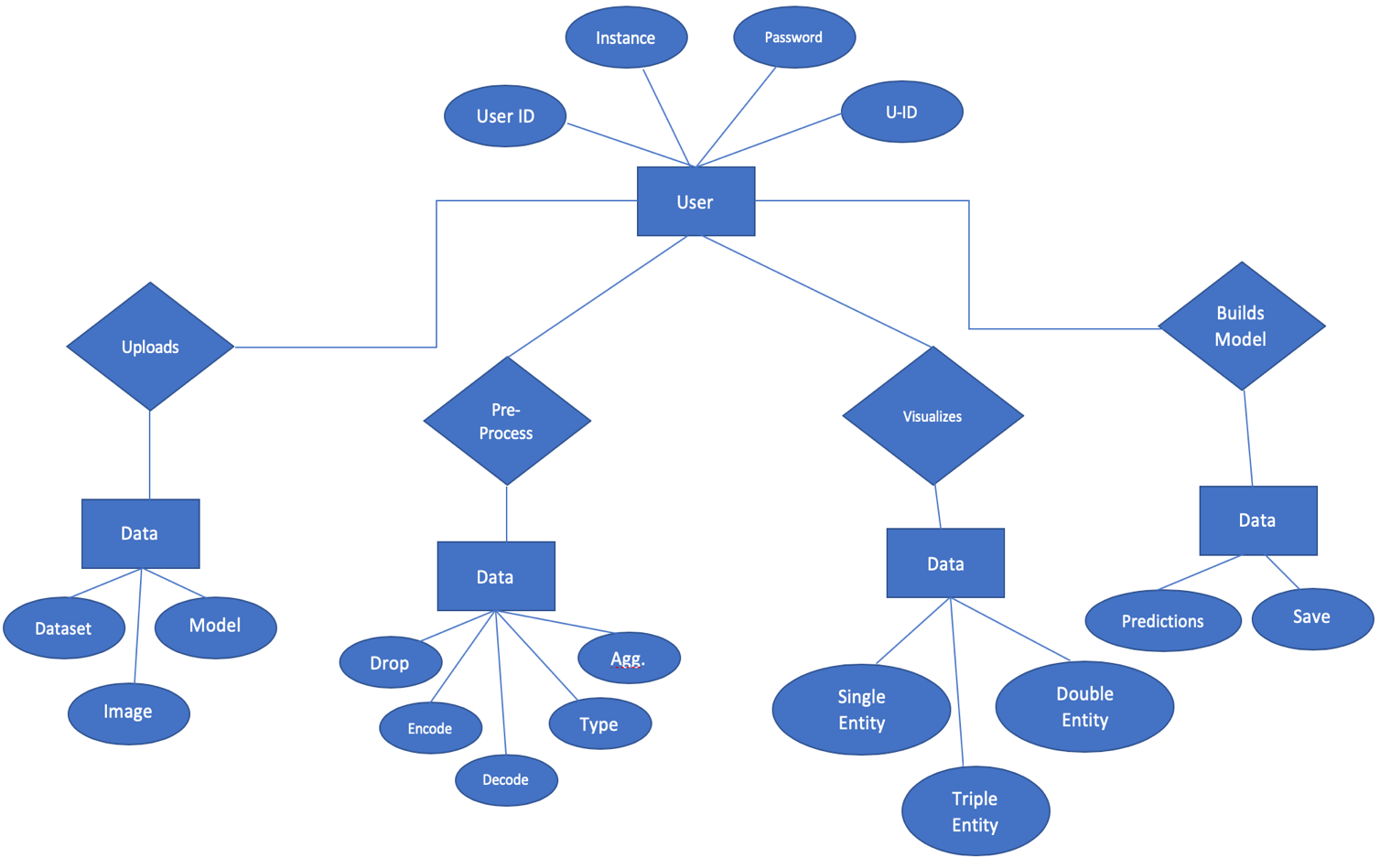
Node

Server

DATABASE

DATABASE

* This is a simple yet elegant web architecture of DUX.
* REST API calls are implemented at every level of communication between server, database and UI.
* This architecture is highly reliable and used universally.
  1. *ER Diagram*

**

* 1. *Workflow Description*

***Step - 1***

Step-3

Store Changes in Db

Step-2

Apply Functions

Pre-Processing

Drop

Aggregation

Encoding

Decoding

Type Conversion

Merge

Work Flow

Upload Datasets

START

Step-2

Store in DB

Operations

Dataset

Image

Dataset

Model

Model

Dataset

Image

* Use-case – The user uploads the data in either of the three different forms. The uploaded data is stored in the firebase
* Work flow -- Demonstrates the backend-perspective of the data that is to be processed

***Step – 2***

Work Flow

* Use-case – The user chooses to modify or pre-process the data. Different options (included in use-case) can be performed on data.
* Work flow – Demonstrates the data storing process in backend.

***Step – 3***

\

Work Flow

Model Building

Save Model in DB

Stop

Build Model

Step-4

Analyze Text

Plot Graphs

Step-3

Prediction

Step-4

Predictions

Save Model

Work Flow

Graph Output

Inference

Visualization

* Use-case – The user visualizes the data with upto three different variables, also the inferences are automatically shown.
* Work flow -- Demonstrates the backend-perspective of the data to be processed

***Step - 4***

* Use-case – Recommended model for the dataset is given to the user. The user chooses to build a model of his choice. Predictions are then returned, the trained model can be saved to DB or downloaded.
* Work flow -- Demonstrates the backend-perspective of the model training and results process.

*2.5 Assumptions and Dependencies*

* User needs to have a basic understanding of data science.
* User inputs are expected to be according to the documentation.
* Assumed that the data is not more than ~200MB
* File formats supported are csv, tsv, jpg, jpeg, png
* Only python machine learning models are supported
* Training time varies according to the data uploaded by the user

*3.1 Supplementary Requirements*

* To maintain a global perspective the server should be well maintained and should support multiple threads for better handling of the requests
* Large scale handling of the data requests is required for the server.
* Documentation must be well written and followed throughout the process of development for better user-experience
* Making the project open-source can yield better results, as there is a lot of scope and expected enhancements for the project

*3.2 Non-functional Requirements*

*Service Availability*

The service in the server is expected to run at all times except when there is a maintenance scheduled by the admin team. The maintenance is scheduled & reported to the user before-hand.

*System Security*

The system is a secure service. The RESTAPI calls used takes in the system-generated security tokens.

*Throughput*

The system is designed to handle asynchronous requests. There can be multiple processes processed and handled at the same time, increasing throughput and memory utilization.

*Reliability*

The system is highly reliable as it handles multiple asynchronous requests. If there is any service issue, it is handled by the backend with intimating the UI about the service changes.

*4.0 System Evolution*

This system is designed to address the problem “*Data noise removal, data pre-processing, visualization and basic model training are time consuming*”. The evolution of this kind of system is endless. It not only addresses a global problem, but provides a viable and feasible solution to solve the problem by streamlining the process to the greatest extent.

* Adding another step of Data analysis – *Fine tuning* the model. This step might be tricky but can be done using backward elimination of features. If this can be done it would be highly useful and serves a critical process element
* Analysis huge corpus of data using Natural Language Processing to support NLP features
* Scaling the project to a higher level to support Big Data Analytics
* Increasing the catalogue of available Machine learning models to support higher custom data analytics
* Support to create ensemble models by passing output of one model to another to yield better results.
* Utilizing the REST API service, many applications can be developed. Just create a UI and then use the existing service to add the backend-magic. Chatbots, Desktop applications, Enterprise-level applications, IOS and Android applications can all be supported with the help of the service created.
* Establish a version control system which handles the present work items created, like different model version with different parameters and many more.
* Save the entire process of steps made to create a pipeline, which can later be used to do the same process all over again just in one go.

***Evolution is Endless***

*5.0 References*

1. <https://github.com/nikhiljsk/movie_data_analysis>
2. <https://www.quora.com/What-are-the-functional-and-non-functional-requirements-in-software-engineering>
3. <https://www.studytonight.com/dbms/er-diagram.php>
4. https://www.ibm.com/developerworks/community/files/form/anonymous/api/library/3cf803c7-f973-4051-99a8-2949fd4ceab1/document/e6bd87f7-a158-4997-b5b6-2a8e2f3e11ed/media/SRS\_Sample.doc