Test Case Preparation

siDBa

Data clustering and statistical modelling of CSV datasets

Prepared by

Nishant Rohan Rodrigues	16BCE0098	rohan.rodrigues2016@vitstudent.ac.in
Annam Sai Kaushik	16BCE0527	annamsai.kaushik2016@vitstudent.ac.in
Shubham Vishwakarma	15BCE0334	shubhamvishwakarma.2015@vit.ac.in

INDEX

1. INTRODUCTION

1.1 Document purpose

This document helps us to understand the various types of testing approached undertaken to make the sure the software is bug free and handles the required tasks as specified in the Software Requirements Documents and follows all the guidelines specified in the Software Design Document as well.
This document also specifies all the test cases and the types of errors thrown by the software in case of faulty input or some other faults. It also includes a list of error codes and specifies what each of them means. The usage of the admin portal for debugging is also specified. The admin portal allows developers to understand the working and take care of each module separately during development and Beta testing of the software.
As the software is open source and free for all to update and modify to one's personal use, a well explained test case preparation is very important to understand the working and integration of each module and the entire software.

1.2 Types of testing

The software will undergo two types of testing. These are namely Black box testing and White box testing.

Black box testing is a form of testing where the test cases are executed on the entire software. This means that we do not bother about each and every module. This allows us to make sure the software works as required and mentioned in the SRS document.

White box testing is a form of testing where each and every module is tested separately. This allows the developer to find the faulty module and to make sure that all the modules function independently as required. This means that it is easier to find and rectify the error.

2. TEST CASES

2.1 Black box testing

Input parameters

Filename
This specifies the filename with the entire file path of the required csv data file.

Database link
The link to the mongoDB. This can be either a local hosted instance or the online hosted database. It is also important to specify the username and the password in the url in-case the database is hosted online

Database name
Refers to the name of the database to which the collection is to be added.

Collection name
Refers to the name of the collection to which the data from the csv file is to be added.

Action

Quit
Save and exit the software

Create
Reads the data from the csv file and creates a collection with the given name in the specified database in the given mongoDB instance.

Logs

All the actions taking place on the software are visible in the admin portal as logs to check for errors and to help rectify and bugs present in the software.

In-case of a successful action all the fields get cleared to make space of entry of new data.
If there is a error in the entry, the fields remain as they are to allow the user to find and rectify the errors.

2.2 White box testing

Modules with their input, actions and outputs are specified below

read_csv()

Input – filename
Action – Reads the data from the csv file
Output – Code with message

Accept
{ ‘code’: 200, ‘data’: csv_data}

Error

{ ‘code’: 400, ‘msg’: ‘Missing file or filepath’}

update_database()

Input – dblink, database_name, collection_name, data
Action – creates a connection to the mongoDB instance, finds the required database, creates a collection and add the data from the csv to the collection.
Output – Code with message

Accept
{ ‘code’: 200, ‘msg’: ‘Collection added to database’}

Error
{ ‘code’: 400, ‘msg’: ‘Connection cannot be established.’}

main()

Input – Reads the input from the GUI
Action - Calls all the required modules in specific order, generates and updates the admin portal.
Output – Maintains the log in the admin portal for debugging and development.

```
    return {"code": 200, "data": csv_data}
except:
    return {"code": 400, "msg": "Missing file or filepath"}
```

```
    return {"code": 200, "msg": "Collection added to database"}
except:
    return {"code": 400, "msg": "Connection cannot be estb."}
```

Fig : Codes with messages to the admin portal.