Dylan Barnes and Jacob Visser

CSci 455, Spring 2016, Dr. Hassan Reza, Ph.D.

DC# Manager Project Analysis and Specification

DC# Manager Project Analysis and Specification Table of Contents

[Revision History 2](#_Toc446171648)

[1 Introduction 3](#_Toc446171649)

[2 Problem Analysis and Solutions 3](#_Toc446171650)

[2.1 Data Storage 3](#_Toc446171651)

[2.2 Labeling, Storing, and Queries 3](#_Toc446171652)

[2.3 File Output 3](#_Toc446171653)

[3 Requirements List 3](#_Toc446171654)

[3.1 Data Conversion and Labeling 3](#_Toc446171655)

[3.2 Storing Data in a Database 4](#_Toc446171656)

[3.3 Query Information 4](#_Toc446171657)

[3.4 File Creation 4](#_Toc446171658)

[3.5 Accessibility 4](#_Toc446171659)

[3.6 Graphical User Interface 4](#_Toc446171660)

[4 Code Diagrams 5](#_Toc446171661)

[4.1 UML Class Diagram 5](#_Toc446171662)

[4.2 UML Use Case Diagrams 6](#_Toc446171663)

[5 Prospective Planning 7](#_Toc446171664)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| VERSION | TEAM MEMBER | DESCRIPTION | DATE |
| 0.1.0a | Dylan Barnes | Initial Document Layout and Introduction | 01/25/2016 |
|  |  | Add Sub-Clause 2.1 *Data Storage* |  |
|  |  | Add Sub-Clause 2.2 *Labeling, Storing, and Queries* |  |
|  |  | Add Sub-Clause 2.3 *File Output*  Add Sub-Clause 3.1 *Data Conversion and Labeling*  Add Sub-Clause 3.2 *Storing Data in a Database*  Add Sub-Clause 3.3 *Query Information*  Add Sub-Clause 3.4 *File Creation*  Add Sub-Clause 3.5 *Accessibility*  Add Sub-Clause 3.6 *Graphical User Interface* |  |
| 0.2.0a | Dylan Barnes | Rename Document  Add number formatting to Main Clauses  Update Requirements under *Requirements List*  Add Main-Clause *Revision History*  Add Main-Clause 4 *Code Diagrams*  Add Sub-Clause 4.1 *UML Class Diagram*  Add Sub-Clause 4.2 *UML Use Case Diagrams*  Add UML Class Diagram 4-1 to *Code Diagrams*  Add UML Use Case Diagram 4-2 to *Code Diagrams*  Add UML Use Case Diagram 4-3 to *Code Diagrams* | 03/19/2016 |

# Introduction

DC# Manager is a database management system with numerous uses. It is part of a larger open source software suite, known as Data Control #. Main features involve storing data from various file formats from either local or online sources, sifting through the obtained data using an intuitive search feature, portability due to lack of language dependencies, and the ability to export data from the database into a useful file format. Further details on these features can be found in the “Requirements List” section.

This document will explain everyday issues that DC# Manager plans to solve, as well as a list of requirements and features necessary to accomplish aforementioned goals. These requirements will be technically expanded in future documentation; this document simply provides a high-level overview.

# Problem Analysis and Solutions

## Data Storage

In typical database systems, data is typically stored in a uniform manner. This manner typically involves user input or a standard file type, such as XML for example. This limits the database, as the user or programmer must manually perform conversions if they wish to store data in other formats.

DC# Manager plans to fix that. In collaboration with DC# Converter, it will be able to properly label and store data in the database from numerous forms of input and data formats, as well as directly pulling data from the internet. This will allow any database hooked up to DC# Manager to automatically and efficiently store various forms of data.

## Labeling, Storing, and Queries

When storing information in a database, there must be some means of labeling the data. This labeling is typically straightforward, as there is a specific type of input.

However, in the case of DC# Manager, there are various different data formats that may be stored in the database. This means a uniform labeling method must be creating, along with a “smart” system that determines how to label the data. Thus, once the content is properly labeled and placed in appropriate tables, related data may be queried via a search feature.

## File Output

There are cases where we may wish to merge specific content from two separate files. With DC# Manager, this will be possible, regardless of the source. There will be functionality which allows users to choose a specific output format, and then create output based on a set of queried content.

# Requirements List

The following requirements list will describe the features and requirements that will be in the initial release of DC# Manager. For the sake of readability, the requirements will be grouped based on their similarities.

## Data Conversion and Labeling

* Shall convert various forms of input, such as XML, JSON, and CSV.
* Input files may also be pulled directly from a web URL.
* Shall initially be done manually, conversions may later be done by DC# Converter.
* Once content is parsed, it shall be labeled.
* Labels shall be determined via a “smart” system.
* Smart system shall be separated by file type, easily allowing the addition of new file type support.

## Storing Data in a Database

* DC# Manager shall connect to a local or cloud database via a database connection string.
* Tables, rows, and columns shall be dynamically created from DC# Manager, thus there is not a need for manual maintenance of the DB contents.
* Content shall be periodically added to the database from local or online sources, thus keeping stored information up to date.
* Converted content, once labeled, shall be dynamically stored in an appropriate table.

## Query Information

* Users shall be able to query for information in the database via a GUI, C# Library, or supplying information via command line arguments.
* Query shall span across multiple tables.
* Query input shall attempt to function similar to Google, in the sense that a search should be simple.
* Queried content may be selected for output to a new file.

## File Creation

* After selecting queried content, user shall select an output format.
* Depending on the format, DC# Converter shall attempt to parse the selected content into an output file.
* Depending on the file type, additional options may need to be specified.

## Accessibility

* DC# shall be portable and language independent.
* In order to be language independent, there shall be GUI support as well as command-lines support.
* Command-line support shall allow other programs in different languages to use DC# functionality.

## Graphical User Interface

* Shall contain a simple interface for basic users.
* Shall contain one search field, as well as a search button.
* Results shall be presented to the user via a data grid.
* This page shall also include additional checkboxes for file creation settings. The list of elements may be re-organized before outputting. Further details have not been decided yet.

# Code Diagrams

## UML Class Diagram

Figure 4-1 contains the current UML Class Diagram of DC# Manager. Due to the use of an Agile methodology, this diagram will continue to evolve as future versions are released.

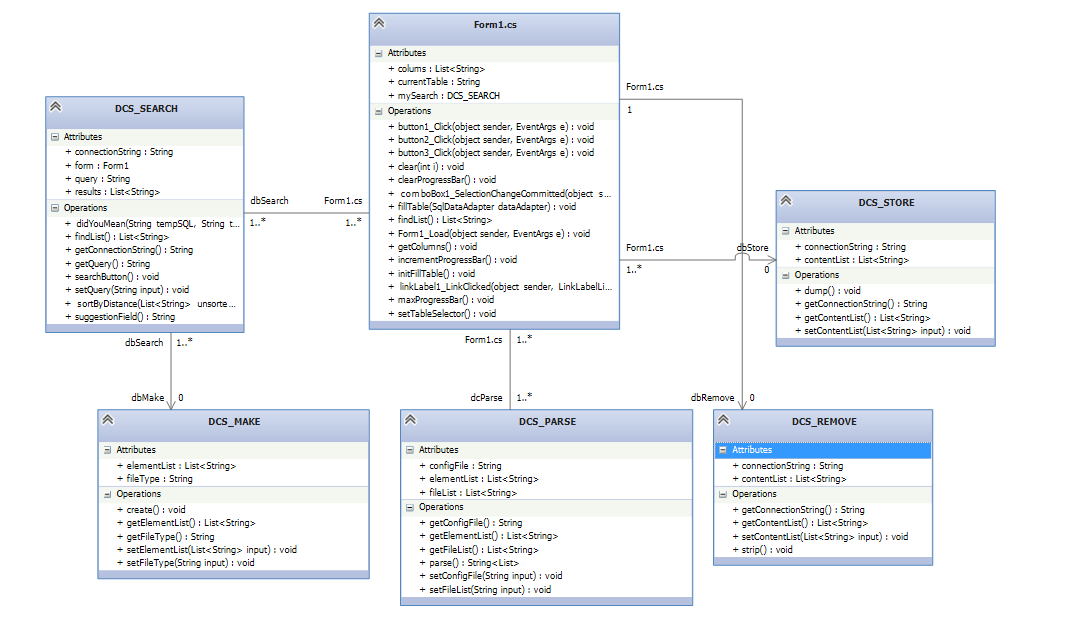


Figure 4-1: UML Class Diagram for DC# Manager

## UML Use Case Diagrams

Figure 4-2 and 4-3 contain the current UML Use Case Diagrams of DC# Manager. Due to the use of an Agile methodology, new diagrams may be added as features become available.

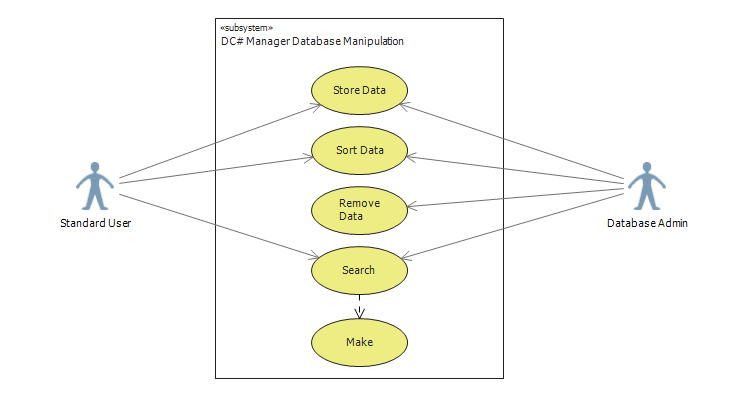


Figure 4-2: Use Case Name: Database Manipulation. Use Case Purpose: The purpose of this use case is to allow the database information to be accessed by the authorized actors (i.e., users). Precondition: User is authorized. Post conditions: Database records will not update unless the user saves changes. Constraints: None. Assumptions: None.

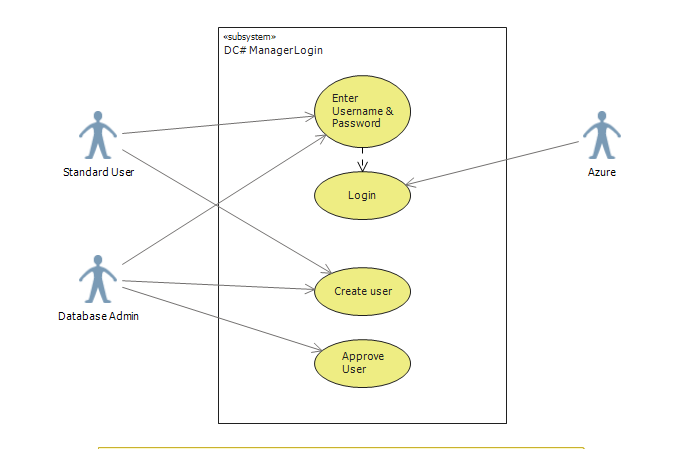


Figure 4-3: Use Case Name: Login. Use Case Purpose: The purpose of this use case is to allow access to the application to authorized actors (i.e., users). It also allows the creation of new users. Precondition: User is authorized if attempting to log in. Post conditions: Database admin must approve new user requests before Azure can approve the login. Constraints: Azure must authorize login based on approved users. Assumptions: Users are only approved by Azure if they are a valid user.

# Prospective Planning

The requirements list above describes the features that are planned for release by the end of the Spring 2016 semester. Once completed, additional features will be added to the DC# suite that allows multiple applications within the suite to take advantage of one another. For example, DC# Manager will be able to convert numerous file types by using the methods from DC# Converter. After we have added additional features and ensured stability, the DC# software suite will be released as open source software.