**Project Design/Implementation Document**

1. Title  
   A short program **title**, the **author's name**, **author's ID**, and **date**.

Title: Movie Database

Author’s Name: Jake Billings

Author’s ID: 105955110

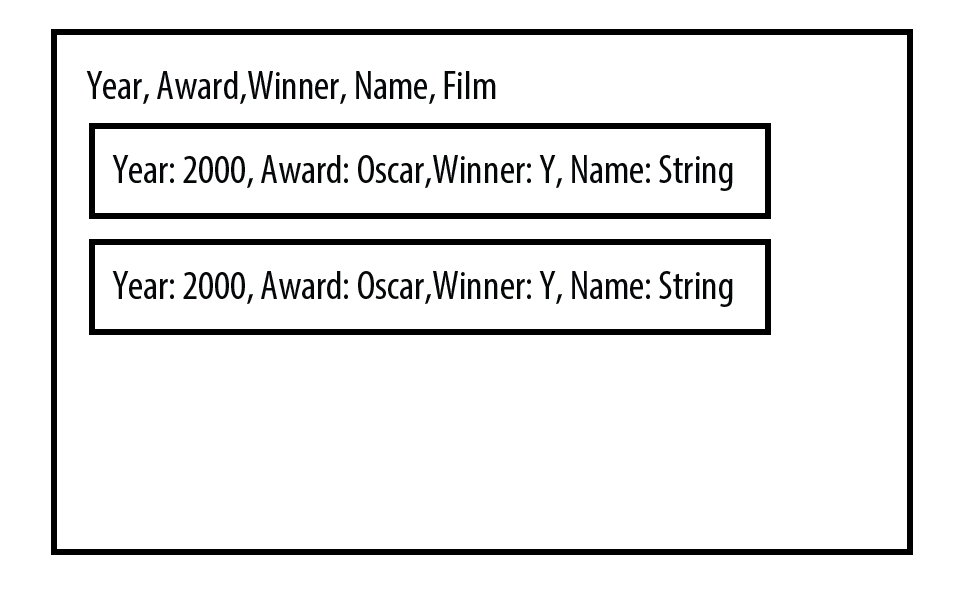
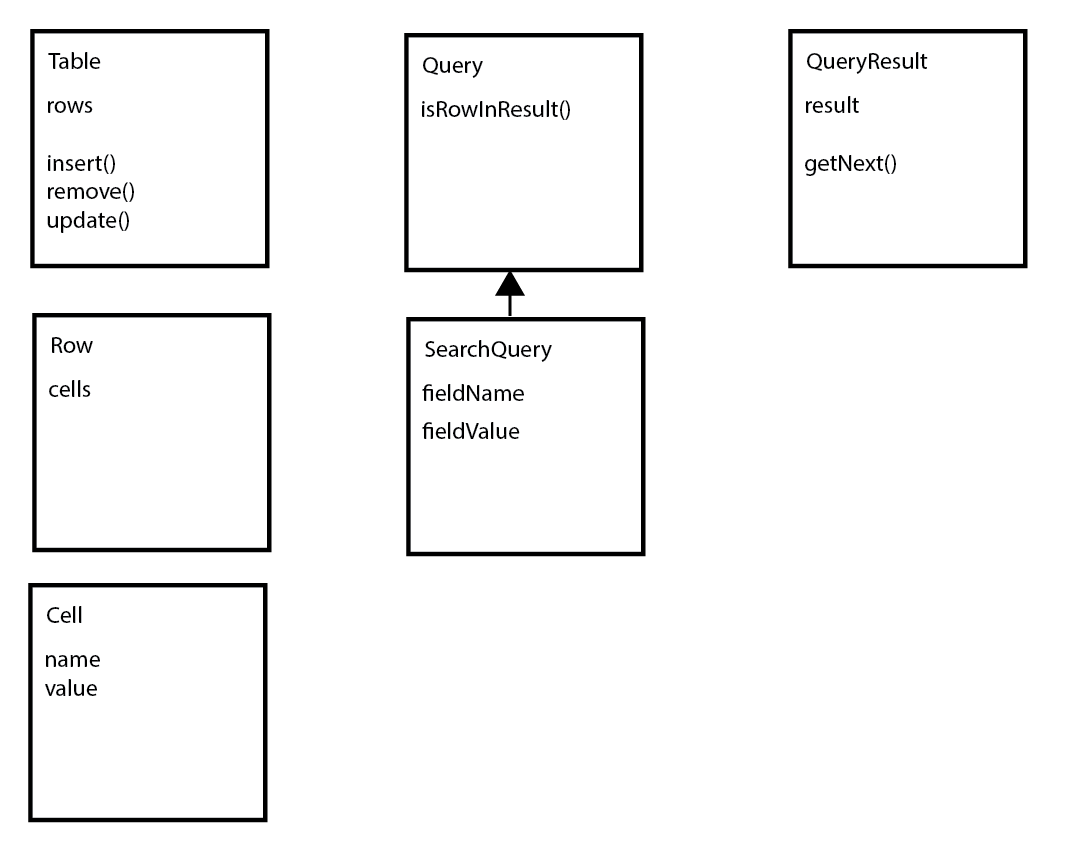
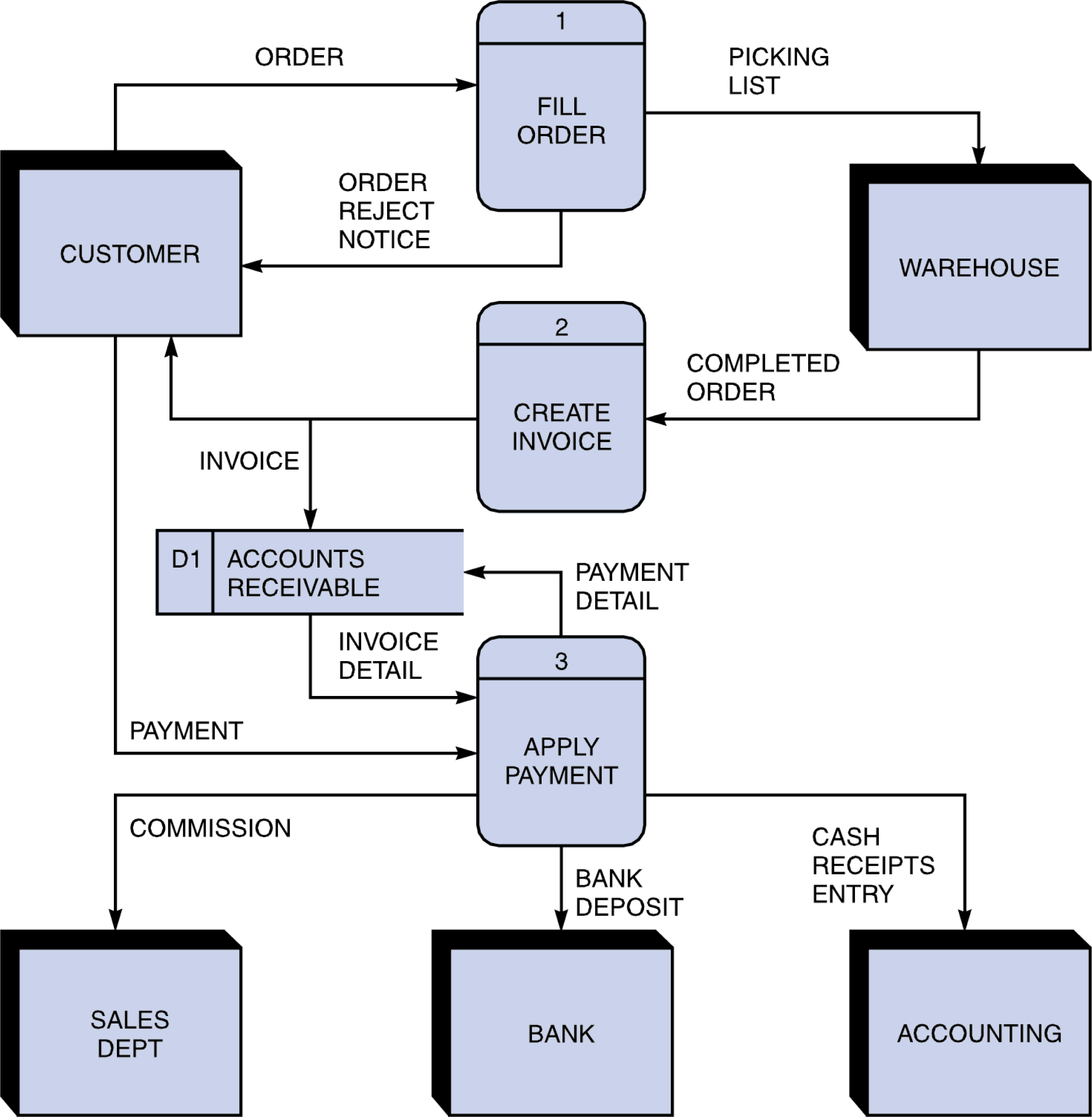
Date: April 11th, 2018

1. Problem DescriptionA brief description of what this program does.

This program loads a amount of csv data about movies, actors, and actresses. Then, it provides a utility to sort and search the data. The database will provide the ability to add, search, modify, sort, and delete records.

1. Overall Software ArchitectureA brief description of major functions and their main roles in the program. You need to explain how the entire program is constructed and how the functions are related each other. You don't have to explain every little function. A diagram to display relation is very useful to get an overall picture. Here's an example of the diagram for a different program but you'll get an idea.

This program will abstract the data tables it is loading to hash maps. Each dataset will be represented as a “table.” Each table will consist of a set of “rows.” Rows are records. Rows are collections of “Cells.” Each cell is a key, value pair. Each row functions as a map. A “Query” is an abstract class that defines a subset of a table using a function. Queries will support searching and matching on any property. A table will accept a query and return a “QueryResult.” A query result will be a collection of “affected” rows. Affected rows are the results of queries. The program will load each row of the CSV file as a row and then insert it into a table. The table will keep “indexes” in the form of binary search trees for every field. This will be memory-inefficient but will result in quick query times. A menu system will generate and execute queries.

  
             
****Figure 1

* + Usually your start and end points (or program exit points) are denoted with a circle
  + D1 is an external data file (arrows point to whether it is input or output)
  + Arrows are labeled with the (general) input and output types…list major data structures or results like input of “Unsorted List and output labeled as “List Sorted by ID”
  + Not listed on the diagram above, but necessary for this project, you should list the major classes, functions and member functions in each block (under the name of the block)

1. Input RequirementsA detailed **list of all external inputs** (from files or keyboard) including a description of the **data type** and **range of valid values** for each input. For input file format and interactive user input, you need to write what data type is used for every field and valid value and length.

File Input: two CSV files to load the initial data tables

Must be CSV formatted (commas between cells and newlines between rows)

Must contain all required data types for the project

(Year,Award,Winner,Name,Film, Etc…)

Keyboard Input: input into a menu system that generates queries

Numeric values are valid for menu options

Strings are valid for searches

Field names are valid for sorts

1. Output RequirementsA detailed **list or description of all outputs (**to files) including a description of the **data type** and **range of valid values** for each output.

Output:

Strings representing menu structure (easy UI stuff)

Tab-delimited rows of sorted/searched data

These will be strings from the Table strcuture

1. Problem Solution Discussion  
   A summary description of the solution steps with algorithms analysis (1 paragraph, approximately 100 words). If any unusual techniques or algorithms are used that need further explanation, and additional paragraph may be used.

A map structure will be used to represent each row. A binary search tree (BST) index will be created for each field in each table in order to optimize searches. Traversal of this tree from top to bottom will yield O(log(n)) searches. Sorting will be performed using bubble sort if there is no noticeable delay. Otherwise, a more efficient sort will be used. Inheritance will be used to make queries generic.

1. Data StructuresA description of choice of your data structures and justification. Of course the main data structure for the database is Binary Search tree. But for some internal operations you may have to use a few sub containers. So include a brief explanation for your choice. For example, "I have considered DS1, DS2, and DS3. Their pros and cons are summarized as follow... I choose DS1 over DS2 and DS3 because ...."

I will use a map to represent each row/record because it will allow for general algorithms that perform searching and sorting without ridiculous templating and typing. I have considered using C++ classes, and I believe this will result in a large amount of duplicate code. I will use a vector to represent query results due to its STL optimizations. A LinkedList could theoretically provide great performance if the order doesn’t matter. However, STL optimizations likely beat my LinkedList. Additionally, sorting algorithms must be performed, and this will not perform well on a LinkedList.

1. User Interface SchemeUser interface scheme should show the menu items at top level and items in sub menus and how to navigate through menus.

The use interface will be as generic as possible. The goal is the build a query object, send it to the database, display the result, and repeat.

The first menu will select query type. (E.g. search/sort/insert)

The second menu will collect all required fields for the query. (E.g. field name and value)

Then, it will execute and display a table of results.

1. Status of Application  
   Note what IDE you developed the Final Project, and whether it compiled and operated properly on the csegrid. Note any requirements that were not met (per the project description). Note any known bugs or issues. If you did extra credit, note the status and what kinds of reports you provided.

This application has not yet been written. This is only a design document.