Gustavo Pedroso UIN: 423002834

Levi Clark UIN:

Terry Chen UIN: 121007055

Daniel He UIN:

**PHASE I: Database Management System Engine**

**Purpose**

To create a generic programming tool for creating applications that utilize basic relational databases. The database management system solves the problem of managing very large amounts of information in a database in an efficient manner. It does so by storing data and organizes it using relational algebraic operations such as selection, projection, renaming, set union, set difference, cross product, and natural join. A relational database management system allows users to enter data and draw relational connections between different elements of data. This allows users to query useful information reliably and easily.

**High Level Entities**

There are two main high-level entities to the database management system--the parser and the engine.

Parser: The parser takes in I/O, either from a file or from user input, and tokenizes it to ensure that the input adheres to the grammar specifications.

Engine: The engine receives tokens from the parser and calls functions accordingly to perform operations, or queries, on the data within the database.

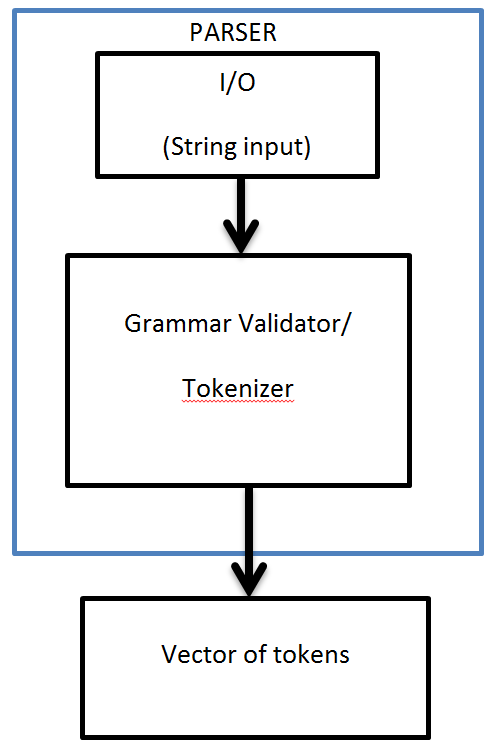
|  |  |
| --- | --- |
|  |  |
|  |  |

**Low Level Design**

**Parser:**

Usage: The parser is used to check a string of input and tokenizes it to determine whether it is a valid token. It is used to provide an interface between the database engine and the user.

The grammar validator receives string input and goes through a recursive tree checking whether each term adheres to the grammar specification. If the terms are valid, then the tokenizer will produce tokens. The parser then outputs a vector of tokens which will be sent to the database engine for processing.



**Engine:** The engine receives tokens from the parser and calls functions accordingly to perform operations, or queries, on the data within the database. The engine provides functions to perform each command and the expression terms from the grammar.

**Benefits, Assumptions, Risks/Issues**

Benefits:

The benefits of the parser/engine design means that we have a database system that can reliably and efficiently manage information.

Risks/Issues:

If large datasets are used, then main memory will become full.

Assumptions:

We assume that small datasets will be used because everything will be held in the main memory.

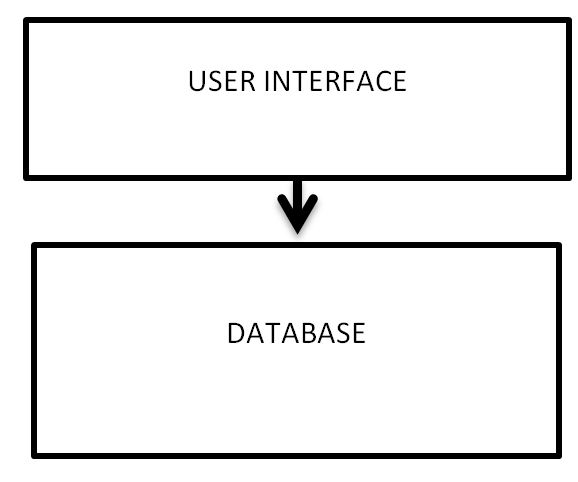
**PHASE II: Database Application**

**Purpose**

The purpose of the database application is to provide an application that will run atop the database engine that will be an interface for the user to use the database. This program provides the bulk of the user interface, allowing the user to display menus, enter data and commands, and show results. With the interface, it allows the user to easily perform operations on the database without having to know all the details of database queries and commands. With this application, the database application allows a hairdresser company to manage hairdressers, customers, and appointment times.

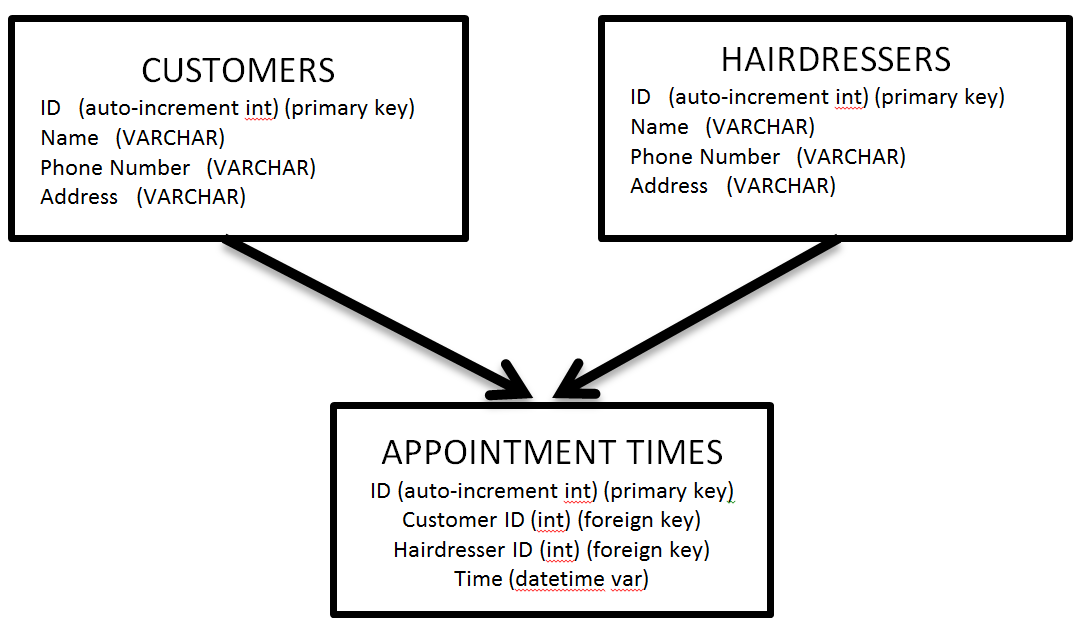
**High Level Entities**

User Interface provides an interface for the user to easily interact with the database engine such as performing operations or query data. It allows the customer to enter in his/her information, select a hairdresser, select an appointment time, and determine the cost.



**Low-Level Design**

The low level design contains three entities-- customers, hairdressers, and appointment times. It also specifies their relations.



**Benefits, Assumptions, Risks/Issues**

Benefits:

The benefits allows users to easily use the database in a modern application that limits user input error through a GUI format. It also allows the customers to interact with the hairdressers and allows a convenient way to schedule appointments rather than using phone calls or paper.

We minimize risk by preventing users from doing terrible things like dropping a table.

Risks/Issues:

If users attempt to set up appointment times simultaneously for a same time slot then there might be a conflict if the hairdressers are the same (race condition). We minimize risks with a graphical interface because user input is limited.

Assumptions:

We assume that hairdressers will not be double booked for identical appointment times.