Project Proposal

Team Members: Jon Casstevens, Stephen Weisenberger, Drew Brown, Chris Dieter

Team Name: The Database Delinquents

Application Name: Miami Dining Catalog

1. **The project Proposal and Basic Design**
2. **Project Proposal**

*What is the goal of your application? What problem will it try to solve?*

The goal of our application is to allow the user to find out what is available to eat at any of the dining halls on campus. Users will be able to perform queries based on what is currently open for business, the users budget, and the users food preferences based on cuisine or food items. Miami’s current website with the dining halls is difficult to navigate, no menu options are listed for the dining halls, and no prices are listed.

*Why is this application relevant/ meaningful? Who may use it and why? What are some difficulties associated with your project?*

This is relevant and meaningful for us as students who eats at the dining halls on campus and will allow us to search what food is sold where, when and at what price. Anyone should be able to use it because the point of this is to allow people to look up what food is sold in various places. This information is vital to creating a realistic and reasonable budget. The difficulties will be in deciding how much information about each place to access and what all to allow the users to query. Other difficulties could include calculating price changes, calculating in tax, calculating in meal plan discounts, and menu changes.

1. **Project Basic Design**

*A project description, i.e., an English description of the problem domain describing the entities and their attributes, their relationships, constraints as seen in the textbook for the Company database and for example, problem 1 of homework 4.*

Our project is a food service database for the food served at Miami University's dining halls. The FOOD\_SERVICE database keeps track of each dining hall on campus and their menus. Currently, the web application that Miami uses to present dining hall details to students (and customers in general) is cluttered and disorganized. Our goal is to provide a clear, concise center of information for the Miami dining hall customer.

* Miami’s food service is organized into dining hall locations to eat on campus. Each dining hall has a unique name, location(address), and an admin who can edit the items on the menu at their location. Each dining hall has a specific cuisine that it serves and a restaurant style(buffet or ala carte). Each dining hall also has its own hours and days of operation.
* A buffet styled dining hall has only a cuisine and a price for its menu. Its menu can change from day to day, which ONLY switches up the main entree and the sides that go with it (fruit, soup, etc. do not get switched out). The date corresponds to what meal is being served in the dining hall.
* A ala carte styled dining hall has a detailed menu for its cuisine, including an item name, item price, and item description. Its menu stays the same from day to day.
* Some dining halls offer discounts to customers that have Diplomat or Express meal plan. Those who have a Diplomat meal plan receive a 50% discount at buffet locations and a 30% discount at a la carte locations. Express meal plans give the customer a 30% buffet discount and tax exemption from every dining location. Faculty are can purchase the Express meal plan.
* Admins are able to change the menu items and prices for the dining halls they are in charge of. Each admin is given a username and password for security.

*A list and description of all the features that you will include in your database application, i.e., the kinds of operations the user can perform on your database.*

* The application upon startup will display an interactive GUI that has two drop down boxes and a search button. The first dropdown box will be filled with all of the dining halls, so that the user can choose which hall he or she wants to obtain information from. Users should be able to view what all a dining hall has to offer in a variety of ways:
  + When a user clicks on a dining hall directly from the dining hall dropdown box, screen, clicking the search button will bring up the menus and prices for that location (while leaving the other dropdown box empty). Above the dining hall menus, the dining hall’s information (hours of operation, address, and whether or not it is buffet or a la carte) will be displayed as static, non-interactive text.
  + Users can choose a filter from the second drop down box at the top of the screen. To clarify, there is one drop down box that contains a few separate filters as individual choices that can be chosen (multiple filters CANNOT be chosen). These filters will include what is currently open for business, price range, cuisine type, restaurant type, meal plans, discounts, and specific food items. Filters will be used to help define which portion of the database to retrieve the results from, and therefore may have certain restraints that determine if user input is accepted or if more predefined search options are displayed.
* Below the two dropdown boxes and the search bar mentioned above, there will be one more dropdown box and a “go” button on the main screen of the GUI. This will contain the three categories that a customer can be categorized in: on campus resident, off campus resident, and Heritage Commons Resident. By selecting one of these options and clicking go, the user will be able to find out which meal plan their selection qualifies for. On campus residents qualify for Diplomat, and the other two options qualify for Express.
* Admins will be able to update their dining hall’s menu items, prices, and discounts. This requires the admins to login using their credentials.
* The purpose of straying from a free-form query entry style to a restricted query entry system (dropdown boxes) is to create a more efficient user experience. Only valid choices will be present, cutting down on erroneous queries and exception-handling code.

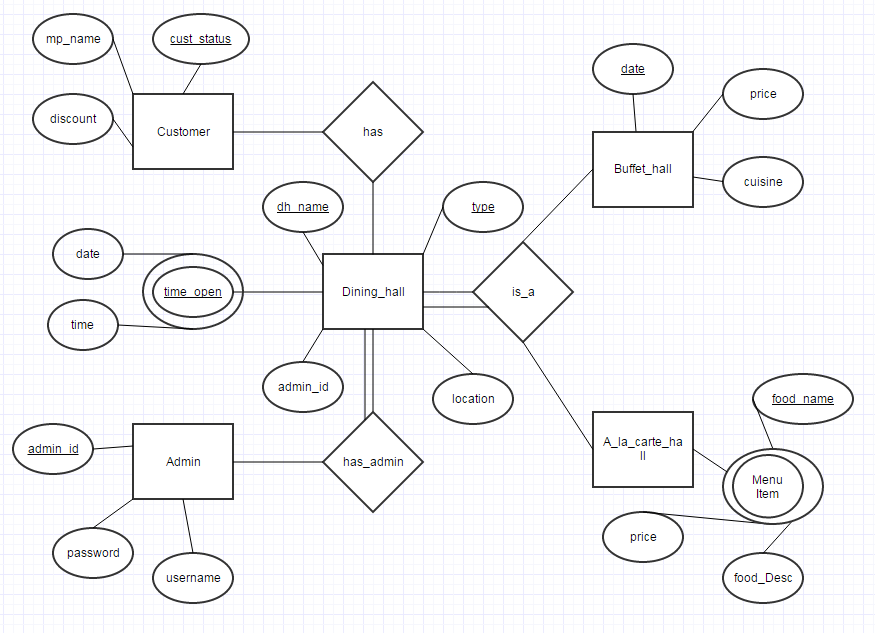
*List as well the optional features that you would like to include if you have the time but do not promise to include in your final deliverable.*

Optional features to add:

* Customer accounts
  + Customer favorite foods
  + Customer favorite locations
  + Customer class schedule
  + Food allergies
* Customer ratings for dining halls
* Customer ratings for food items
* Customer reviews for dining halls
* Food allergy information
* Nutritional information
* Additional Locations: Markets
* Admins can manage menu specials

Optional search filters to add:

* + Filter by ratings. (User login req.)
  + Filter by favorites. (User login req.)
  + Filter by food allergies.
  + Filter by nutritional info.
  + Filter by markets.

*A detailed ER diagram for your database with an explanation of what each entity and relationship represents.*

*A relational database schema: the result of mapping your ER diagram into a database schema.*

Final Schema:

**Dining\_Hall** (Dh\_name, Type, location, adminId, customer\_status)

FK: adminId, customer\_status

**Buffet\_menu**(Dh\_Name, Date, Price, Cuisine)

FK: Dh\_name

**A\_la\_carte\_menu**(Dh\_name, Food\_name, food\_desc, Price)

FK: Dh\_name

**Operation\_hours**(Dh\_name, Day, Time)

FK: Dh\_name

**Payment\_Type** (Customer Status, Meal\_plan\_name, Discount Amount)

**Admin**(AdminId, Password, Username)

1. **SQL Statement Collection**

*The SQL statements to change your database schema into actual database tables.*

CREATE TABLE dining\_hall (

dh\_name varchar(30) not null,

dh\_type varchar(30) not null,

location varchar(30),

adm\_id int not null,

customer\_status varchar(30),

Primary Key (dh\_name, dh\_type),

Foreign Key (adm\_id )references admin\_detail(adm\_id)

Foreign Key (customer\_status) references payment\_type(customer\_status)

);

CREATE TABLE operational\_hours(

dh\_name varchar(30) not null,

time\_open varchar(10),

time\_close varchar(10),

wk\_day varchar(15) not null,

Primary Key (dh\_name, wk\_day, time\_open),

foreign key (dh\_name) references dining\_hall(dh\_name)

);

CREATE TABLE admin\_detail (

username varchar(20),

adm\_id int not null,

password varchar(20),

Primary Key (adm\_id)

);

CREATE TABLE a\_la\_carte\_menu(

dh\_name varchar(30) not null,

food\_name varchar(30) not null,

food\_desc varchar(100),

price int,

Primary Key (dh\_name, crt\_date, food\_name),

foreign key (dh\_name) references dining\_hall(dh\_name)

);

CREATE TABLE buffet\_menu(

dh\_name varchar(30) not null,

cuisine varchar(30) not null,

price int,

Primary Key (dh\_name),

foreign key (dh\_name) references dining\_hall(dh\_name)

);

CREATE TABLE payment\_type (

customer\_status varchar(30) not null,

mp\_name varchar(40),

discount int,

Primary Key (customer\_status, mp\_name)

);

**IV. Specific SQL Statements**

* Aggregate Function:

Select \*, AVG(price2)

From dining\_hall, operational\_hours, payment\_type, admin\_detail, buffet

Where dh\_name = dh\_name1 AND dh\_name3 = dh\_name AND customer\_status1 = customer\_status

Group By dh\_name

* Insert Statement:

d\_name, d\_type, loc, d\_admin, c\_stat are all strings entered by admin

Insert INTO dining\_hall Values ('" + d\_name + "', '" + d\_type + "', '" + loc + "', " + d\_admin + ", '" + c\_stat + "')

* Multiple Table Selection:

Select \*

From dining\_hall, operational\_hours, payment\_type, admin\_detail, buffet

Where dh\_name = dh\_name1 AND dh\_name3 = dh\_name AND customer\_status1 = customer\_status

Order By dh\_name