Gabe Benninghoff

CIS-382

Dr. Ayano

Final Project

The Database

The database itself is not super complicated. I built it using Microsoft Azure. The program has a plus button that generates a blank database. Then I used a notebook to create all the tables at once. These tables are:

Customer(customer\_id, cu\_first\_name, cu\_last\_name, phone\_nu, street, cu\_city,

cu\_state, cu\_zip)

Employee(employee\_id, em\_first\_name, em\_last\_name)

Card(card\_no, card\_fname, card\_lname, exp\_date, cvc, bank, customer\_id)

foreign key: Customer(customer\_id)

ServiceRecord(service\_id, customer\_id, service\_cost) foreign key:

Customer(customer\_id)

Appointements(appt\_id, service\_id, appt\_date, appt\_time, customer\_id) foreign

key: ServiceRecord(service\_id) Customer(customer\_id)

All of these tables are designed for practicality and with the idea of my parents using them for real. They include all the relevant information that a company might need when adding in the different information. For each of the primary keys that are id’s I used a self generating step function so that there is a unique id. This will produce duplicates if customers are added in through the form but it prevents the user from having to try and guess where the ids are at. I did misspell appointments the first time and had to rebuild the table as it wasn’t changing. As for views there is a built in view that allows the user to view the current/upcoming appointments.

The Website

To talk about the website side is to give credit where credit is due. First I have to give credit to Tech With Tim and his [video](https://www.youtube.com/watch?v=dam0GPOAvVI&t=5341s) which was a basic database consisting of adding and subtracting notes. While I did a lot of editing I cannot claim the basic design as my own and it laid a lot of the groundwork. I also have to cite ChatGPT as a huge help. I have never received any formal course in frontend web development and as such I used ChatGPT to generate a lot of the HTML code. This could have been done by hand but would have resulted in many extra hours of work on my part learning how just buttons work. Now, onto the actual implementation. The basic website is a Flask based frontend connected to Azure/Docker through pyodbc. I took several hours figuring out pyodbc, the problem was as simple as calling version 17 instead of 18 in my connection code. Once I figured out this I (ChatGPT) was able to create html views that have field forms for all of the information that is passed onto the database. These forms pass along all of the information received to the views.py script that specifies what each page does. There is a base html template that all edit on top of so that there is the nav-bar at the top of the page. Tech With Tim’s application provided the implementation for a login, logout, and registration page. However, this page used a different database created within a models.py using SQLite. I kept this for security reasons, so that users can’t access the login information. There is an auth.py that provides the necessary code for login, logout, and sign-up page that does a lot of the same as the views.py but with more of the security side. The \_\_init\_\_.py makes this able to be ran and a lot of that implementation is from the Flask documentation itself. As for the homepage I (ChatGPT) generated a basic HTML script that was able to display and image of my companies logo.