Group 47 - Hair Salon Database Management System

Members: Monica Nguyen, Ahmed Farazi, Bhavan Pahuja

Abstract

As a hairstylist's clientele grows, it becomes increasingly difficult for stylists to remember the specific details and preferences of each client. By creating a database that logs relevant hair details for each client, hairstylists can be better equipped to understand the needs and wants of clients and deliver optimal service. Remembering small details about each client regarding their personal lives and hair promotes better customer service as it lets each client feel valued as services can be personalized for them. In this project, we aimed to implement a database that would allow salons, stylists, and clients to make the process of getting hair services done more smooth and efficient.

Introduction

Many salons today are taking clients on a walk-in basis or are running an appointment-based system without logging important information about their clients from previous visits. As a result, there are stylists acting as if they remember the exact haircut or colour formula for each client when in reality, they do not. After a few thousand clients, hairstylists are bound to give clients the wrong haircut or style without proper documentation. The widely used software, StyleWare Touch (Version 3.76.58.713)[2] has come out with a database system that stores client information. However, this is a limited software because it does not allow clients to be categorized under particular stylists and it stores information on haircuts only. It would be worthwhile to address the limitations of this software to allow for database management for salons that offer colour and other speciality services as well. This way, the database is applicable for more than just one type of salon and allows grouping of clients for particular stylists as necessary. In addition, by having a database that the stylist can access prior to seeing clients, the stylist can prepare necessary equipment in advance to make appointments carry out more smoothly.

Our System

We used MySQL and PHP to implement a system that an admin, employee and customer can access. On the client end, they are able to enter an appointment to book with a stylist. On the stylist end, they are able to manage clients, client info, appointments, services and equipment. The admin is able to manage salon input, employee input and grouping stylists and receptionists.

Project Design

The different users of the HSDM include

a) Admin: From adding to deleting to viewing data stored in the Salons, Employees, Stylists and Receptionist sections. They will be to control the inputs and maintain the overall record of the salons.

b) Employee: In the employee section, the user will be able to add, delete and view current client records. Our system allows us to keep hair characteristics of each client and thus making it easier to know earlier what type of client we are dealing with. Make appointments which can also be deleted and also add the type of service the client is requesting for a precise system of record keeping.

c) Client: Our system can also allow clients to make appointments quickly and thus saving time wasted by waiting in line and also on the phone looking for the next appointment. Our system is easy to use and allows clients to book a particular time, the stylist they want to book and also the type of service they want.

Entity-Relationship Diagram

One of the only changes we made to the EERD was that we condensed services into 1 attribute called Service name for the Service entity.

https://app.diagrams.net/#G1vTF_-KM9HJ5-hvOJDHa62_7tb0Rm54Hh

Please see EntityRelationshipDiagram for a clear PDF

Implementation

Relational Model

https://app.diagrams.net/#G1k1q_E_Bimui5CKIZ2x98xKWrNSoL2T2i

Please see RelationalModel for a clear PDF

Some changes that were made included adding in some attributes to specify the location of the client and salon. Instead of just an address, there should also be state/province, postal code, etc. It made more sense to include these attributes directly under the tables of client and salon, rather than separating it into another table, so we removed the multi-attributed ‘location’ entity that we had in the previous model. The multi-attributed ‘previous appointment’ entity was also changed to be direct attributes to ‘appointment.’ We made the change of not underlining the foreign keys and keeping it to only underlining the primary keys. We realized for equipment to be meaningful, there should be a name associated with the equipment ID along with a potential owner id of the equipment. It also did not make sense to have an employee ID for ‘hair characteristics’ because there would be high amounts of redundant data as each employee would be able to view all client hair characteristics. Instead, it made sense for us to allow employees the access to see the client data rather than include it as an attribute of hair characteristics - of which it is not related to that category of data. For the most part, the relations and the arrows drawn between keys did not have changes. After making the database, we had learned that it is worthwhile to invest a lot of time into planning the tables and attributes because the addition, edit, and deletion of data heavily rely on the framework it was built upon. To simplify the table for services, we grouped together all the services of haircut, styling, perm, etc., into servicename instead. It is also crucial to keep the database concise and uncluttered with redundant data that could be otherwise retrieved by querying through the tables on certain keys.

SQL Statements

```
SELECT salonid FROM salon where name='".$salonName."
INSERT INTO employee(idsalon, firstname, lastname, phone, email, address,
postalcode, city, stateorprovince, country) VALUES (?,?,?,?,?,?,?,?,?,?)
```

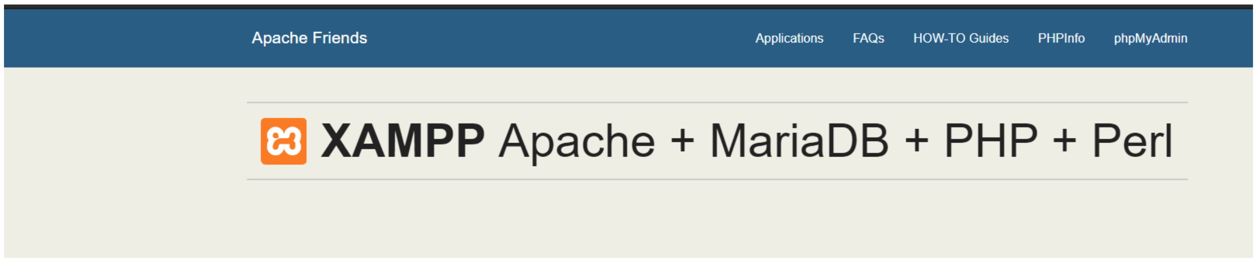
```
INSERT INTO receptionist(employeeid) VALUES (?)
INSERT  INTO  salon(phone,  email,  name,  address,  postalcode,  city,
stateorprovince, country) VALUES (?,?,?,?,?,?,?,?)
SELECT employeeid FROM employee where firstname='".$employeeName."
INSERT INTO stylist(employeenumber) VALUES (?)
DELETE FROM employee WHERE employeeid='" . $_GET["employeeid"]
DELETE FROM salon WHERE salonid='" . $_GET["salonid"] . "
SELECT employeeid FROM employee
SELECT name FROM salon
"SELECT clientid FROM client where firstname='".$clientName."
"SELECT serviceid FROM service where servicename='".$serviceName."'"
"SELECT employeeid FROM employee where firstname='".$employeeName."'"
"INSERT  INTO  appointment(idservice,  idclient,  idstylist,  startdatetime,
enddatetime, previousapt, previousservice) VALUES (?,?,?,?,?,?,?)"
"SELECT clientid FROM client where firstname='".$clientName."'"
"INSERT INTO haircharacteristics(clientno, colorformula, length, texture,
style, notes) VALUES (?,?,?,?,?,?)"
"INSERT INTO client(salonno, phone, firstname, lastname, email, address,
postalcode,    city,    stateorprovince,    country,    discount)    VALUES
(?,?,?,?,?,?,?,?,?,?,?)"
"INSERT   INTO   equipment(employeeno,   name)   VALUES   ('$employeeno',
'$equipmentName')"
"SELECT employeeid FROM employee where firstname='".$employeeName."'"
"SELECT equipmentid FROM equipment where name ='".$equipmentName."'"
"INSERT INTO service(servicename, cost, equipmentno) VALUES (?,?,?)"
"SELECT * FROM appointment"
"DELETE FROM appointment WHERE idservice='" . $_GET["idservice"] . "'"
"SELECT * FROM haircharacteristics"
"DELETE  FROM  haircharacteristics  WHERE  clientno='"  .  $_GET["clientno"]  .
"'"
"SELECT * FROM client"
```

API Documentation

Check see the separated file named PostmanDocumentation.

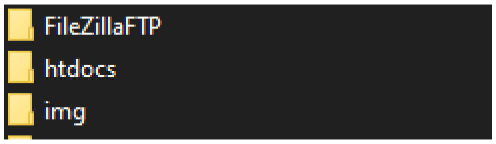
Quick User Guide

1. Download and install XAMPP for the required system.
2. Open any web browser to check if Xampp downloaded properly by going to the link **localhost** and you will see the below image if successful.

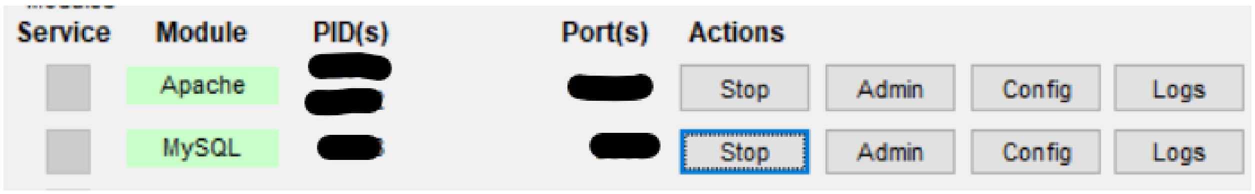


Welcome to XAMPP for Windows 8.0.3

3. Extract the files submitted into a folder called **htdocs** in Xampp folder in C drive.

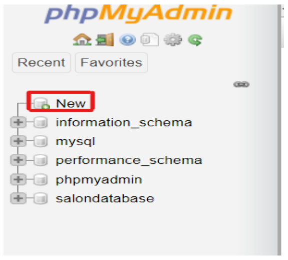


4. Open Xampp and click start button for both Apache Module and MySQL as shown below

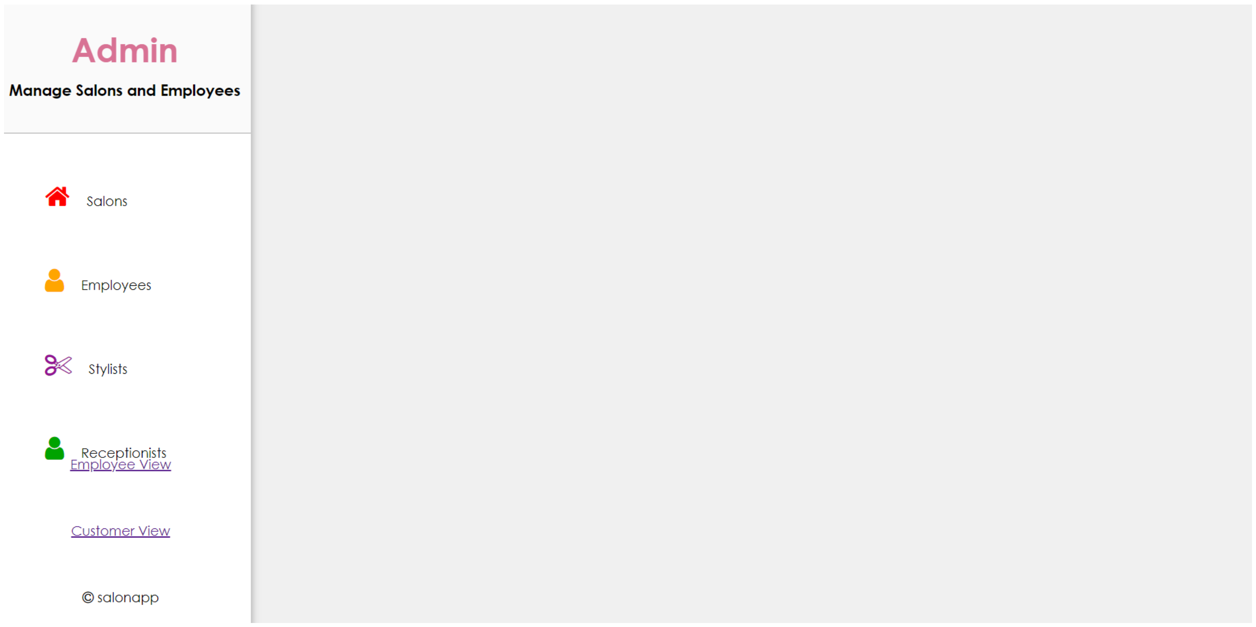


Service	Module	PID(s)	Port(s)	Actions			
	Apache	[redacted]	[redacted]	Stop	Admin	Config	Logs
	MySQL	[redacted]	[redacted]	Stop	Admin	Config	Logs

5. Click on the Admin button for the MySQL, which will take you to the phpMyAdmin page.
6. Click on new for creating a new database.



7. Name the new database as **salondatabase.sql** and after clicking on create, make sure to import the salondatabase.sql from the submitted folder.
8. Now as the database is setup, please copy paste the link to view the php pages - localhost/salonapp-main/frontend/admin/admin.php
It will take you to the admin page.



9. From here you can use the sidebar to navigate around and when required click the back button to go to the previous page.
10. When using the features in the webpage like adding or deleting, you can go to phpMyAdmin to see the changes made in the database.

Reference: https://www.studentstutorial.com/php/php-mysql-data-delete.php

- Please note that a new service has to be added each time for new appointments

- Also please note there are stored procedures but I did not end up using them as I found a separate way to prevent sql injection and did not end up using the stored procedures

- The admin/customer/employee view represents the privileges of what each member can do in terms of making changes to the database.
- The submitted code uses bind_param() and statements to prevent SQL injection for text fields
- Each time you click the nav bar, there are add/delete/view all for each entity