Studio Cascadia
Pacific Northwest Art Gallery Chain
Database Management System

Group 7 - Lauren Graham & Kevin Hayden
CS 340
Project Step 6 Final
6/6/2022

URL to Index Page: http://flip1.engr.oregonstate.edu:1177/

Executive Summary

The initial project steps for the Studio Cascadia database did not display the purpose of each entity in the Database Outline. As part of the outline, the purposes of each of the entities within Studio Cascadia's Entity Relationship Diagram were added to clarify the purpose for the variety of entities present. Naming conventions were an early design dilemma. PascalCase, camelCase and snake_case had all been implemented at different times. To satisfy readability as well as common naming conventions in the course, our team utilized snake_case with capital first letters for entities and all lowercase for attributes. The result was a more clearly readable database structure with easier to implement references for syntax in SQL queries and in flask coding. It was noted that our original design of having a 1:1 relationship between the Sales and Shipments entity could present a problem where the Shipment could be considered as an attribute. This brought our attention to understand that practically for Studio Cascadia's business nature, that there could be large Pieces sold by the stores and that many shipments may be necessary for a single sale. The entity relationship between these two were then modified to be M:1. Our early design featured a circular relationship between Sales, Customers and Shipments that was in some ways confusing to reviewers and it was removed. It was deemed that the relationship between Customers and Shipments was actually unnecessary as the Sales Entity held FKs for both of these other entities within it. The intent was to remove redundant data by changing this. The last note taken into consideration for the initial step was to include how relationships are connected through FKs in our Database Outline. This was promptly added to increase understanding and how relationships connected within the outline as they were displayed on our Schema.

Delving further into normalization, attributes were altered to create more reasonable detail when necessary in the database. Longer naming conventions that detailed entities within each attribute were removed in favor of keeping things simpler. For example, 'pieces_id' became simply 'id'. This was judged as effective by the fact that the id for a piece within the Pieces_Artists Entity was then named as pieces_id when related to as a FK. It was found that a NULLable relationship was not present between Shipments and Sales, with that possible actuality in mind, that was adjusted. The early web application utilized a Verification system to have the user agree to changes before submitting a CREATE or UPDATE operation. This was removed as it was considered confusing by users.

The User Interface of the web application was steadily improved through development. The initial dark mode aesthetic was altered in favor of more neutral light colors to display the database in less contrasting nature. Action buttons such as Submit, Add New, Edit and Delete were kept changed to a light blue color to attract the attention of the system administrator to see where interactions with the database can occur at a simple glance.

Functionalities needed to satisfy the project scope were focused on and excessive operations that were originally in place were removed. The end result was providing SELECT and CREATE operations for all entities. UPDATE operation was utilized on the Artists Entity as this was the most desired to have the edit functionality for the artists to have updated contact information at any time for receiving payment. DELETE functionalities are implemented on all Entities but Locations to provide ease of use for mistaken CREATEs. Locations was only implemented with a single CREATE operation as the Locations will hold a living archive of all artists and pieces that were featured and sold at each of Studio Cascadia's present and future locations regardless of them closing down unexpectedly. BROWSE is featured in the CREATE functionalities for the Pieces_Artists entity that dynamically fills the fields to add a piece to belong to an artist within the database's storage system.

Studio Cascadia Overview

Studio Cascadia is an art gallery chain currently with three locations: Portland, Oregon, Spokane, Washington, and Seattle, Washington. Each location holds approximately 24 to 30 art pieces at any given time. Studio Cascadia typically works with local artists who produce art in a variety of styles and mediums. The gallery requires a database to store artist information, customer information, piece information, location information, and transactions.

Retail prices vary depending on the cost of materials for the artist but often fall in the range of $500 for smaller pieces and up to $15,000 for large pieces. Payments can be made via check, card, or cash. Customers can “mix and match” payment types. While the number of sales can be unpredictable, it is important to keep an adequate record of each transaction. Studio Cascadia is selective about the artists it hosts, holding preference for artists who bring in more frequent sales and larger profits.

Additionally, because many large pieces, such as sculptures, require shipment, shipment information must also be accounted for. This information is not mandatory; however, because pieces may also be picked-up.

Although pieces typically have only one artist, several of the larger works, such as large glass pieces and pieces of mixed-media art, have been created by several artists working together. Studio Cascadia frequently hosts the artwork of a spouse-duo as well.

While the location of each piece could be documented with only an ID, Studio Cascadia hopes to expand, opening several more locations across the Pacific Northwest region. Therefore, specific information pertaining to the locations, such as state and zip code, will be stored.

Database Outline

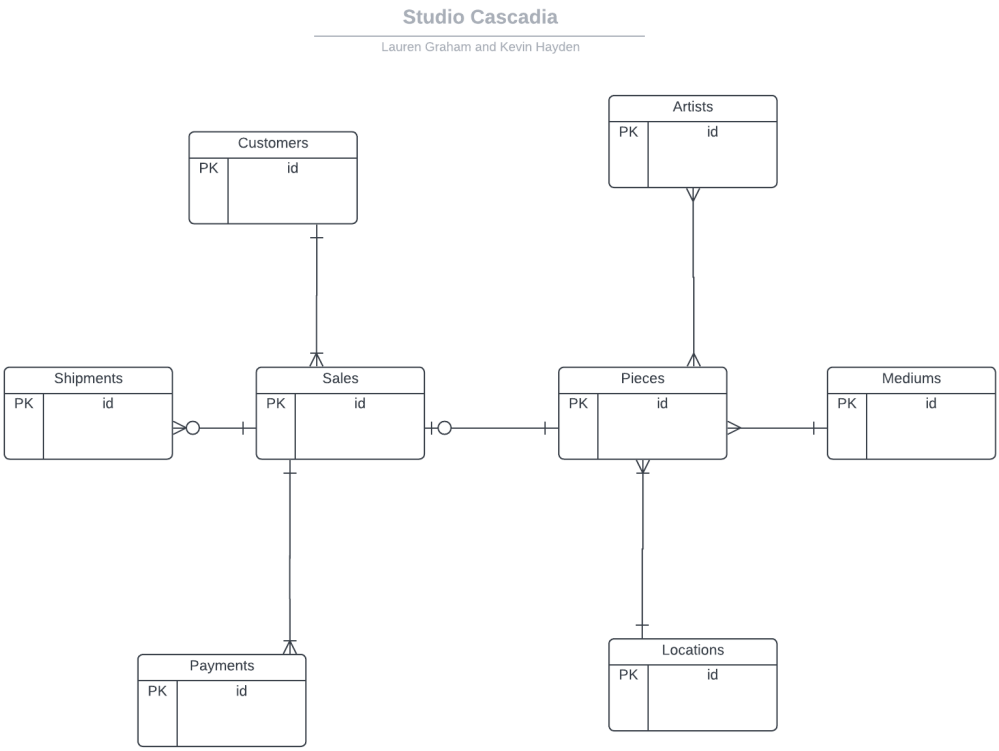
- **Sales**: records information regarding each sale of a single piece
    - id: PK, int, auto_increment, unique, not NULL
    - customer_id: FK, int, not NULL
    - piece_id: FK, int, not NULL
    - date: date, not NULL
    - amount: decimal, not NULL
    - ship: tinyint, not NULL
    - Relationships:
        - There is a M:1 relationship between Sales and Customers because customers can be involved in multiple transactions. Customer_id is included as a foreign key.
        - There is a 1:1 relationship between Sales and Pieces. Because of the high price and one-of-a-kind nature of many of the art pieces, Studio Cascadia prefers to sell each piece separately rather than putting more than one piece on a single ticket. Piece_id is included as a foreign key.
        - There is a 1:M relationship between Sales and Shipments. Because some art pieces can be quite large, they may need to be shipped in multiple shipments. As depicted in the ER diagram, shipments are not mandatory in this relationship.
        - There is a 1:M relationship between Sales and Payments. Payments can be made with multiple methods for a single sale. For example, a customer may choose to pay with some cash, with the rest of the balance charged on a card.
- **Customers**: records information regarding Studio Cascadia's customers
    - id: PK, int, auto_increment, unique, not NULL
    - last_name: varchar, not NULL
    - first_name: varchar, not NULL
    - address: varchar, not NULL
    - city: varchar, not NULL
    - state: varchar, not NULL
    - zip: char(5), not NULL
    - phone: varchar, not NULL
    - Relationships:
        - There is a 1:M relationship between Customers and Sales. Customers can make one or many purchases.
        - There is a 1:M relationship between Customers and Shipments. Since a customer can make many purchases, and because some pieces require multiple shipments, it is possible for customers to have a relationship between many shipments.
- **Shipments**: records information regarding the shipments of pieces. The gallery wants to keep track of whether or not the piece has been shipped and whether or not it has been delivered. It is also important to record tracking information if it is available.
    - id: PK, int, auto_increment, unique, not NULL

    - sale_id: FK, int, not NULL
    - shipped: tinyint, not NULL
    - delivered: tinyint, not NULL
    - carrier: varchar, not NULL
    - tracking: varchar
    - Relationships:
        - There is a M:1 relationship between Shipments and Customers. As mentioned previously, customers can make many purchases requiring shipments. Some individual purchases will even require multiple shipments. Customer_id is included as a foreign key.
        - There is a M:1 relationship between Shipments and Sales. A shipment is not considered mandatory since a customer can choose to pick up the item directly. Pieces will not be shipped together. Sale_id is included as a foreign key.
- **Payments**: records payment information. This is a separate entity from Sales because multiple payments could contribute to a single sale.
    - id: PK, int, auto_increment, unique, not NULL
    - sale_id: FK, not NULL
    - date: datetime, not NULL
    - card: tinyint, not NULL
    - cash: tinyint, not NULL
    - check: tinyint, not NULL
    - card_number: varchar
    - exp_date: date
    - amount: varchar, not NULL
    - Relationships
        - There is a M:1 relationship between Payments and Sales because multiple payments or payment types may be used for a single sale. sale_id is included as a foreign key.
- **Pieces**: records information regarding each piece of art present at any of the Studio Cascadia locations. The gallery would like to store information regarding the location of the piece, title, year published, style, and medium (glass, acrylic, tempera, oil, charcoal, marble, etc.). Sometimes the medium will be recorded as mixed-media. Other pieces of information which are important for sale are price, whether or not the piece is available for purchase, whether it's been commissioned by a customer, and whether or not the piece is "on hold" for a customer.
    - id: PK, int, auto_increment, unique, not NULL
    - location_id: FK, not NULL
    - medium_id: FK, not NULL
    - title: varchar, not NULL
    - year: varchar, not NULL
    - price: decimal, not NULL
    - available: tinyint, not NULL
    - hold: tinyint, not NULL

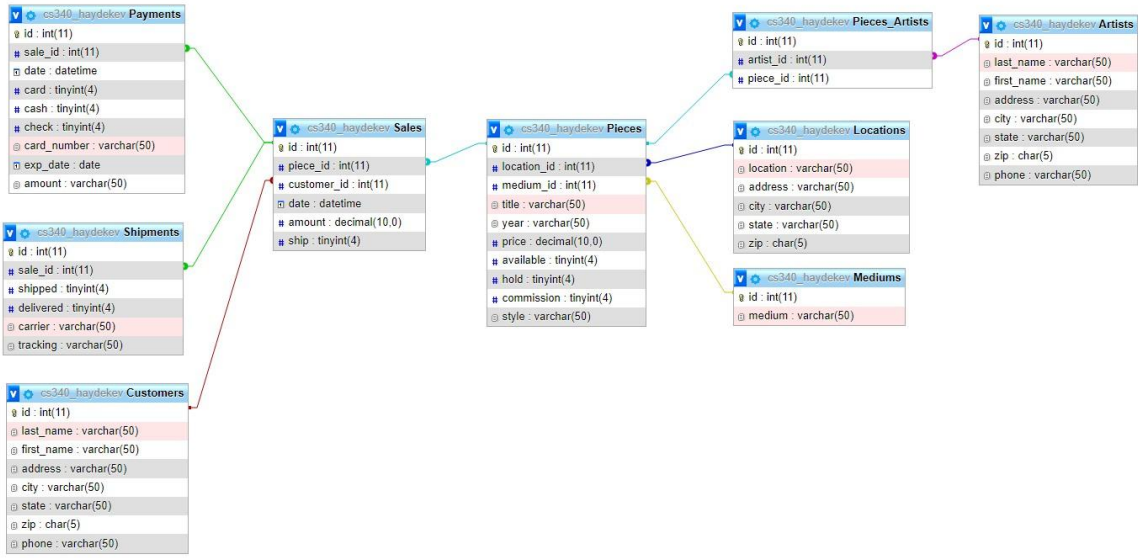
    - commission: tinyint, not NULL
    - style: varchar, not NULL
    - Relationships:
        - There is a M:M relationship between Pieces and Artists. Artists typically create more than one piece. Although pieces typically have only one artist, several of the larger works, such as large glass pieces and pieces of mixed-media art, have been created by several artists working together. Because this relationship is M:M, an intersection table, Pieces_Artists, is included. This appears on the ER diagram as a 1:M relationship between Pieces and Pieces_Artists.
        - There is a M:1 relationship between Pieces and Locations because many pieces are hosted at each location. Location_id is included as a foreign key.
        - There is a M:1 relationship between Pieces and Mediums since each piece will have one medium stored in the database but there may be more than one piece made with a medium. Medium_id is included as a foreign key.
        - There is a 1:1 relationship between Pieces and Sales because pieces are each sold individually, never with multiple pieces on a single ticket.
- **Pieces_Artists**: This an intersection table which exists due to the M:M relationship between Pieces and Artists
    - artist_id: FK, int, not NULL
    - piece_id: FK, int, not NULL
    - Relationships:
        - Due to the nature of an intersection table, there is a M:1 relationship between Pieces_Artists and Pieces, as well as a M:1 relationship between Pieces_Artists and Artists. artist_id and piece_id are both included as primary keys.
- **Artists**: records information regarding the artists responsible for art pieces hosted in the gallery
    - id: PK, int, auto_increment, unique, not NULL
    - last_name: varchar, not NULL
    - first_name: varchar, not NULL
    - address: varchar, not NULL
    - city: varchar, not NULL
    - state: varchar, not NULL
    - zip: char(5), not NULL
    - phone: varchar, not NULL
    - Relationships:
        - There is a M:M relationship between Artists and Pieces. See “Pieces” for more information.
        - As mentioned above in “Pieces” and “Pieces_Artists”, there is a 1:M relationship between Artists and Pieces_Artists due to the nature of an intersection table.

- **Mediums**: records information regarding media used for art pieces such as acrylic, oil, glass, charcoal, and mixed-media
    - id: PK, int, auto_increment, unique, not NULL
    - class: varchar, not NULL
    - Relationships:
        - There is a 1:M relationship between Mediums and Pieces. Each medium could have many pieces within said category.
- **Locations**: records basic location information for each gallery location
    - id: PK, int, auto_increment, unique, not NULL
    - location: varchar, not NULL
    - address: varchar, not NULL
    - city: varchar, not NULL
    - state: varchar, not NULL
    - zip: char(5), not NULL
    - Relationships:
        - There is a 1:M relationship between Locations and Pieces. Each piece can only be present at one location. The studio sells one-of-a-kind pieces only.
- **Foreign Keys:**
    - 1:M relationship between Sales and Payments implemented via sale_id as a FK within Payments.
    - 1:M relationship between Customers and Sales implemented via customer_id as a FK within Sales.
    - 1:M relationship between Sales and Shipments implemented via sale_id as a FK within Shipments.
    - 1:1 relationship between Pieces and Sales implemented via piece_id as a FK within Sales.
    - 1:1 relationship between Pieces and Sales implemented via piece_id as a FK within Sales.
    - 1:M relationship between Mediums and Pieces implemented via medium_id as a FK within Pieces.
    - 1:M relationship between Locations and Pieces implemented via location_id as a FK within Pieces.
    - M:M relationship between Pieces and Artists implemented via artist_id and piece_id as FKs within Pieces_Artists.

Entity-Relationship Diagram



Schema


cs340_haydekev Payments
id : int(11)
sale_id : int(11)
date : datetime
card : tinyint(4)
cash : tinyint(4)
check : tinyint(4)
card_number : varchar(50)
exp_date : date
amount : varchar(50)
cs340_haydekev Shipments
id : int(11)
sale_id : int(11)
shipped : tinyint(4)
delivered : tinyint(4)
carrier : varchar(50)
tracking : varchar(50)
cs340_haydekev Customers
id : int(11)
last_name : varchar(50)
first_name : varchar(50)
address : varchar(50)
city : varchar(50)
state : varchar(50)
zip : char(5)
phone : varchar(50)
cs340_haydekev Sales
id : int(11)
piece_id : int(11)
customer_id : int(11)
date : datetime
amount : decimal(10,0)
ship : tinyint(4)
cs340_haydekev Pieces
id : int(11)
location_id : int(11)
medium_id : int(11)
title : varchar(50)
year : varchar(50)
price : decimal(10,0)
available : tinyint(4)
hold : tinyint(4)
commission : tinyint(4)
style : varchar(50)
cs340_haydekev Pieces_Artists
id : int(11)
artist_id : int(11)
piece_id : int(11)
cs340_haydekev Locations
id : int(11)
location : varchar(50)
address : varchar(50)
city : varchar(50)
state : varchar(50)
zip : char(5)
cs340_haydekev Mediums
id : int(11)
medium : varchar(50)
cs340_haydekev Artists
id : int(11)
last_name : varchar(50)
first_name : varchar(50)
address : varchar(50)
city : varchar(50)
state : varchar(50)
zip : char(5)
phone : varchar(50)

Sample Data

Pieces

id	location_id	medium_id	title	year
1	1	4	Event Horizon	2019
2	2	3	Flamingo Macchia	2019
3	2	3	Seagrass Seaform	2021
4	2	2	David Lynch	2022

Pieces continued

price	available	hold	commission	style
1300.00	0	0	0	Futurism
6000.00	0	0	0	Abstract
8000.00	0	0	0	Abstract
4500.00	1	0	0	Portrait

—

Artists

id	last_name	first_name	address	city	state	zip	phone
1	Rindler	Wolfgang	800 W Campbell Rd	Richardson	TX	75080	8008892443
2	Chihuly	Dale	509 NE Northlake Way	Seattle	WA	98105	2067818707
3	Jackson	Leslie	509 NE Northlake Way	Seattle	WA	98105	2067818707
4	Domont	Jordan	33 NW Park Ave	Portland	OR	97209	5034674909

—

Pieces_Artists

artist_id	piece_id
1	1
2	2
3	2
2	3

—

Mediums

id	class
1	Oil
2	Mixed
3	Glass
4	Digital

—

Locations

id	location	address	city	state	zip
1	Studio Cascadia Seattle	519 E Pine St	Seattle	WA	98122
2	Studio Cascadia Spokane	1326 E Sprague Ave	Spokane	WA	99202
3	Studio Cascadia Portland	22 SW 3rd Ave	Portland	OR	97204

—

Customers

id	last_name	first_name	address	city	state	zip	phone
1	Graham	Lauren	206 Julian Ave	Honolulu	HI	96818	555-960-8888
2	Brownstein	Carrie	3244 NE 46th Ave	Portland	OR	97213	555-438-9999
3	McCready	Mike	2708 65th Place SE	Mercer Island	WA	98040	555-545-2222

—

Sales

id	customer_id	piece_id	date	amount	ship
1	1	1	2021-10-31 11:05	1300.00	1
2	2	3	2022-01-23 16:33	8000.00	1
3	3	2	2022-01-23 10:10	6000.00	1

—

Payments

id	sale_id	date	card	cash	check	card_number	exp_date	amount
1	1	2021-10-30 14:30	0	1	0	NULL	NULL	1300.00
2	2	2022-01-23 16:30	0	0	1	NULL	NULL	8000.00
3	3	2022-01-23 10:05	0	0	1	NULL	NULL	3000.00
4	3	2022-01-23 10:08	0	1	0	NULL	NULL	3000.00

—

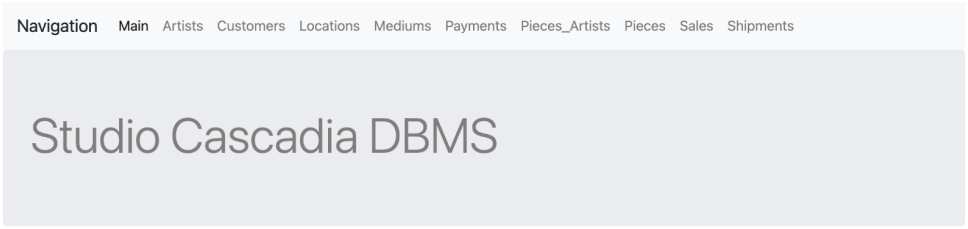
Shipments

id	sale_id	shipped	delivered	carrier	tracking
1	1	1	1	Prime Hawaiian Movers	NULL
2	2	1	1	FedEx	05218962519602
3	3	1	1	FedEx	05227994632790

UI Screen Captures

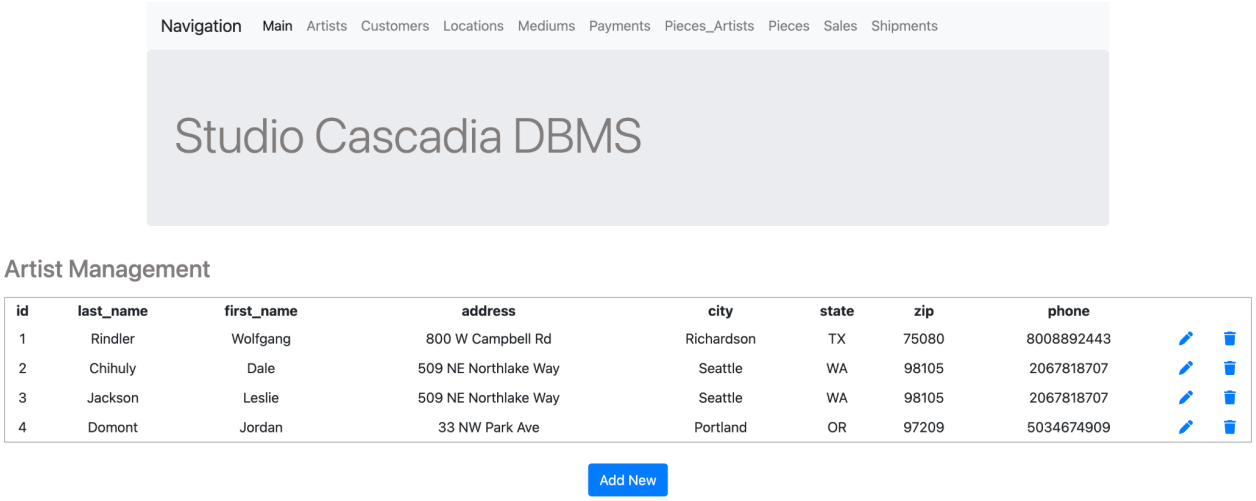
Main

Studio Cascadia DBMS Home Page



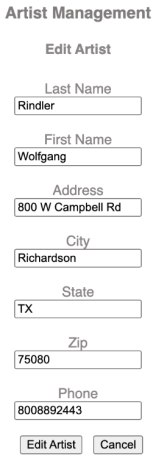
Artists

CREATE/READ/UPDATE/DELETE Artists Page:



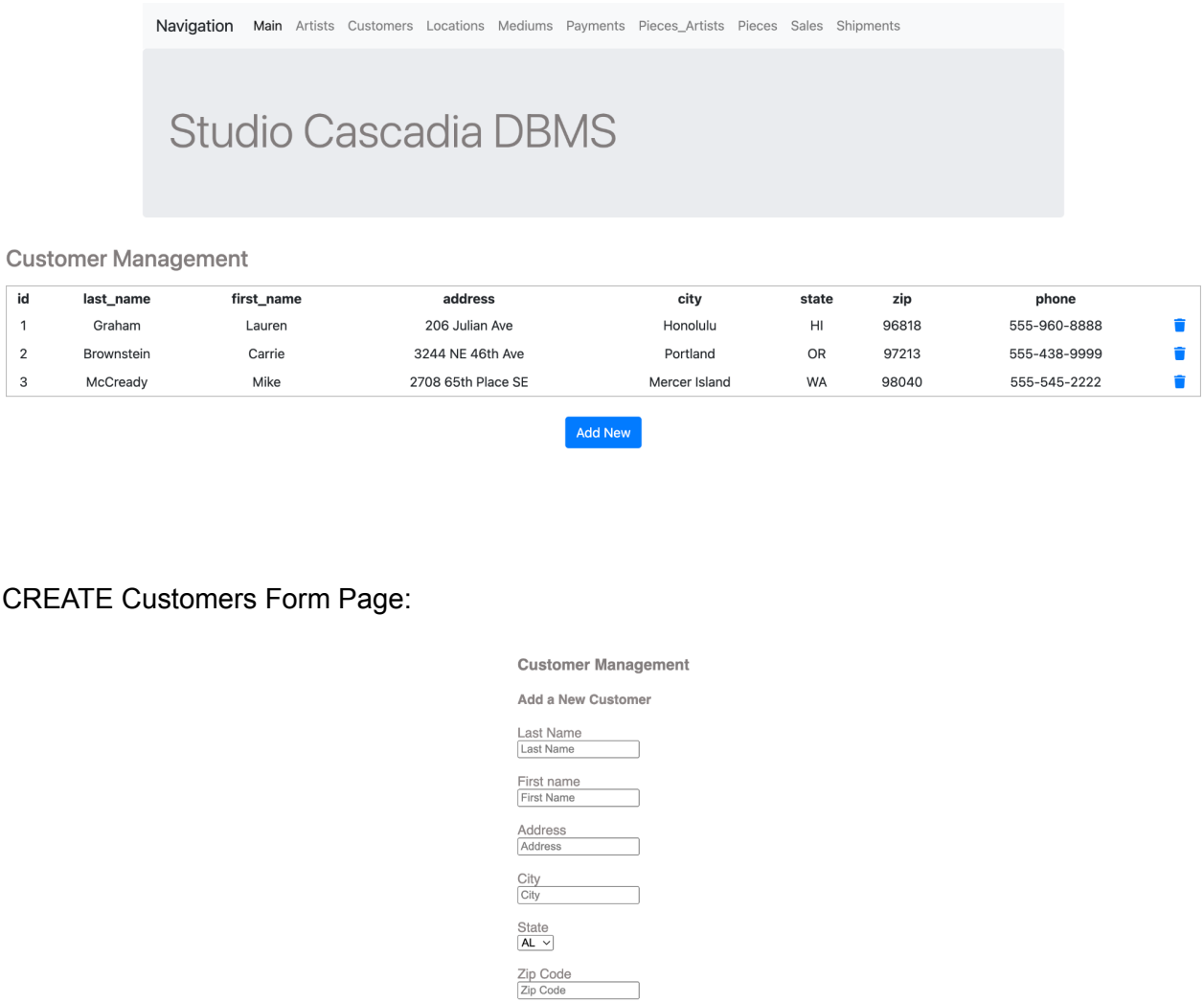
CREATE Artists Form Page:

UPDATE Artists Form Page:



Customers

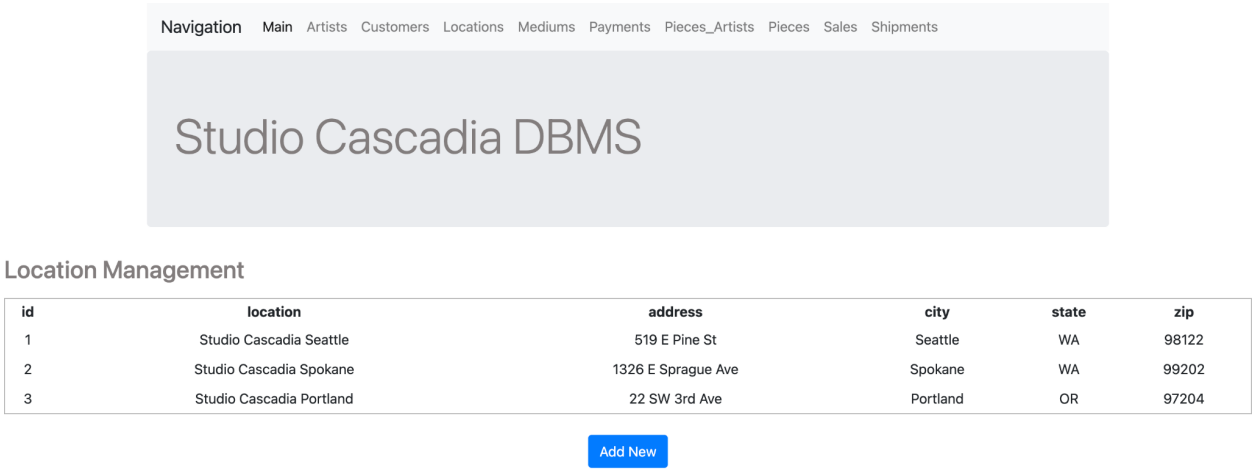
CREATE/READ/DELETE Customers Page:



CREATE Customers Form Page:

Locations

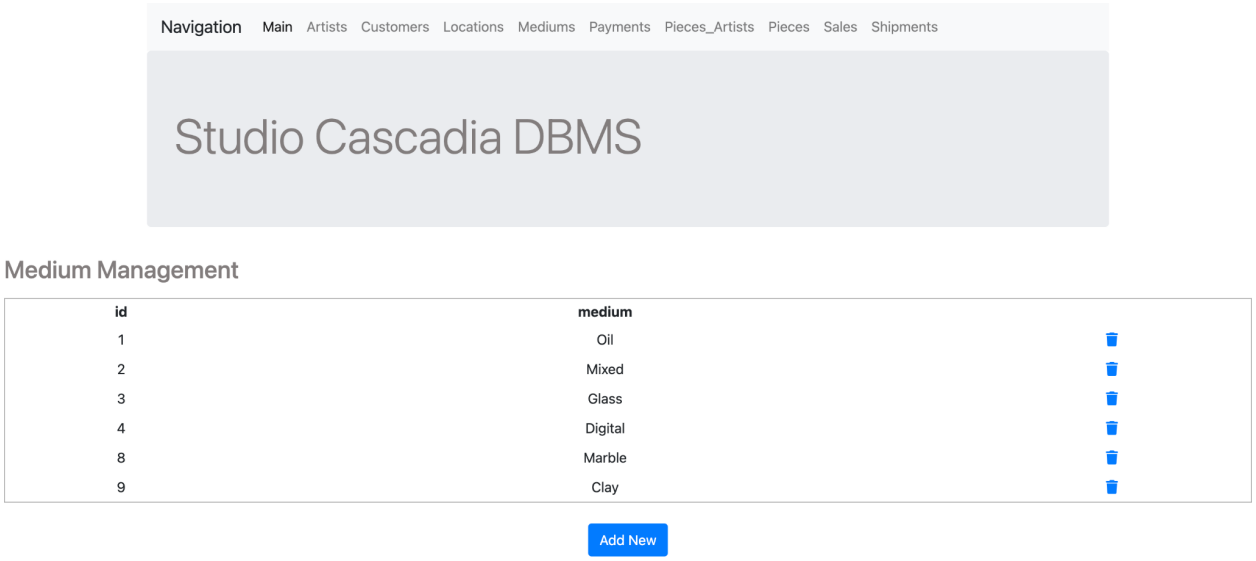
CREATE/READ Locations Page:



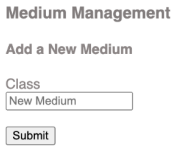
CREATE Locations Page:

Mediums

CREATE/READ/DELETE Mediums Page:

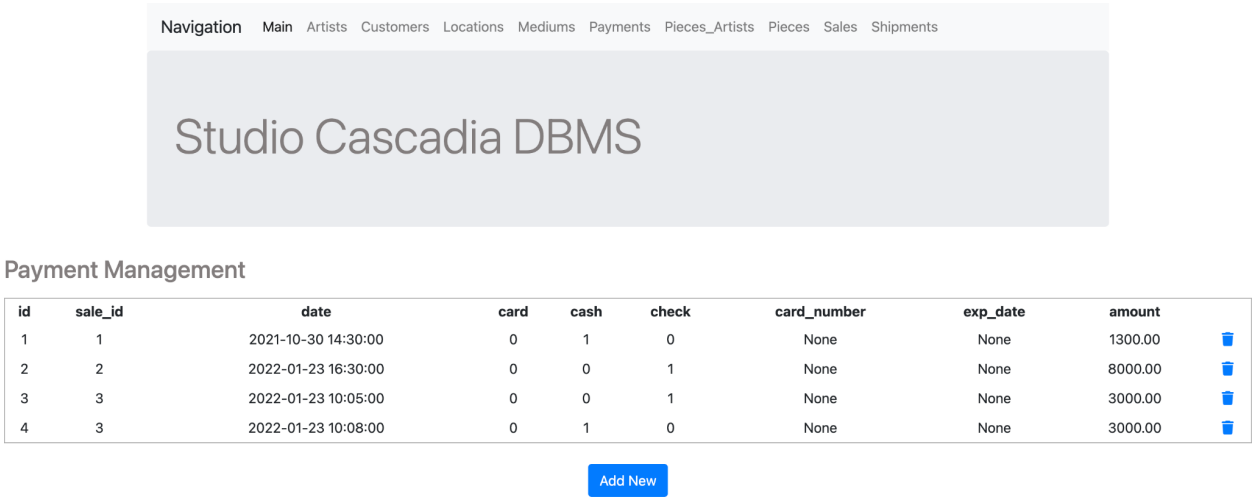


CREATE Mediums Page:



Payments

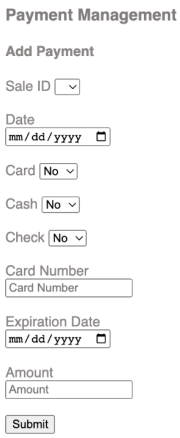
CREATE/READ/DELETE Payments Page:



id	sale_id	date	card	cash	check	card_number	exp_date	amount	
1	1	2021-10-30 14:30:00	0	1	0	None	None	1300.00	
2	2	2022-01-23 16:30:00	0	0	1	None	None	8000.00	
3	3	2022-01-23 10:05:00	0	0	1	None	None	3000.00	
4	3	2022-01-23 10:08:00	0	1	0	None	None	3000.00	

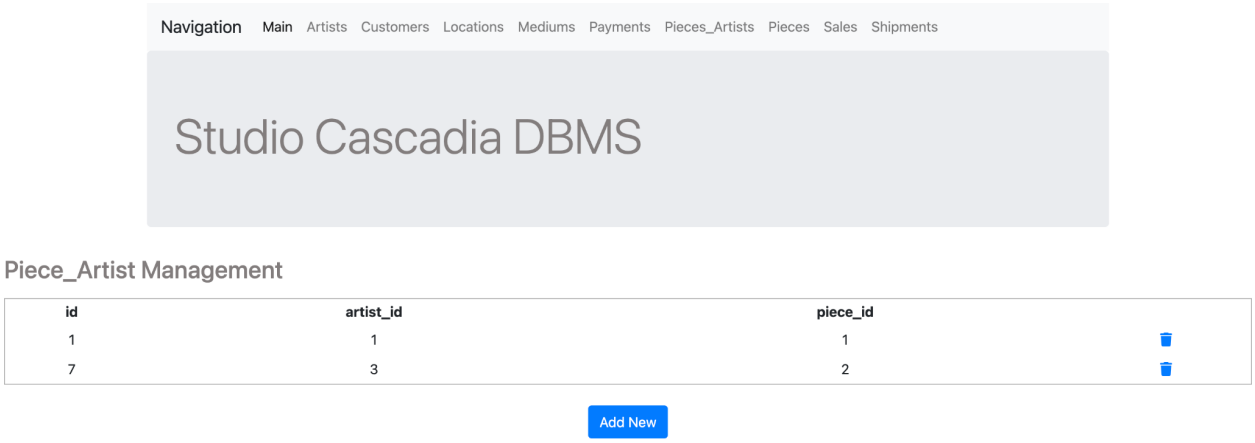
Add New

CREATE Payments Page:

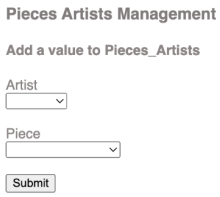


Pieces_Artists

CREATE/READ/DELETE Pieces_Artists Page:

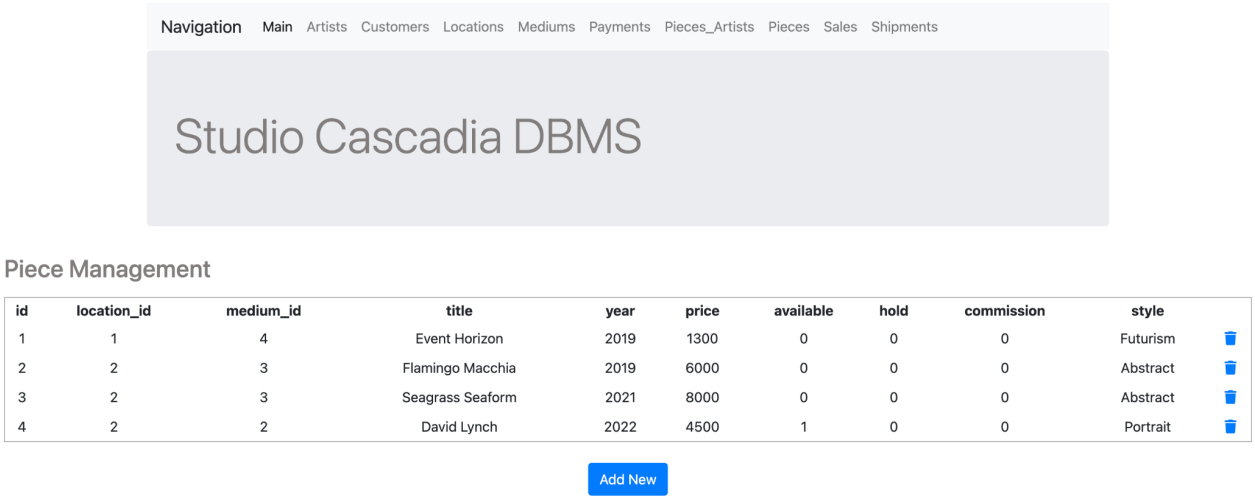


CREATE Pieces_Artists Page:

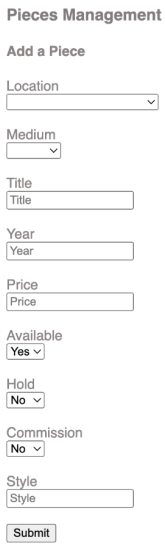


Pieces

CREATE/READ/DELETE Pieces Page:

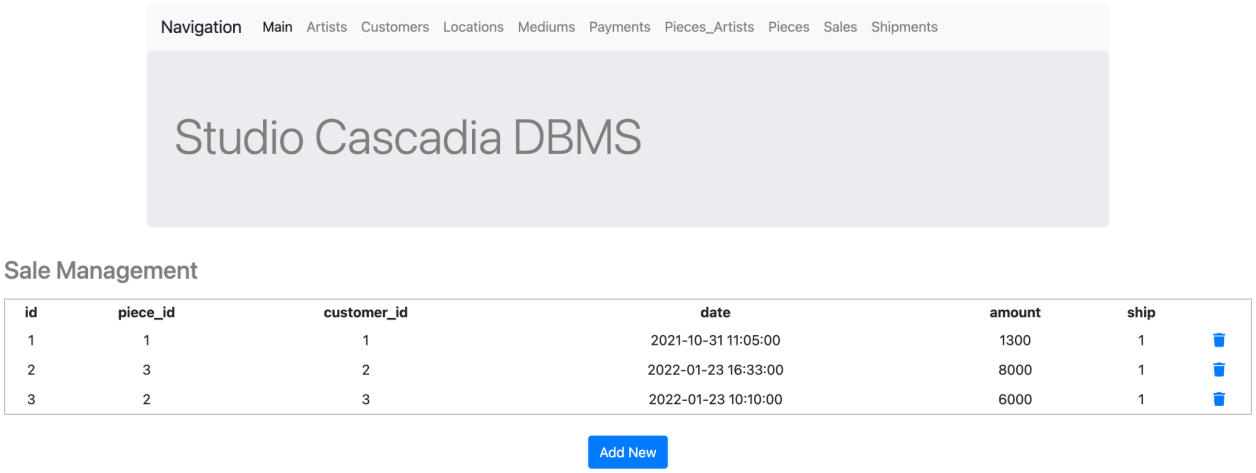


CREATE Pieces Page:



Sales

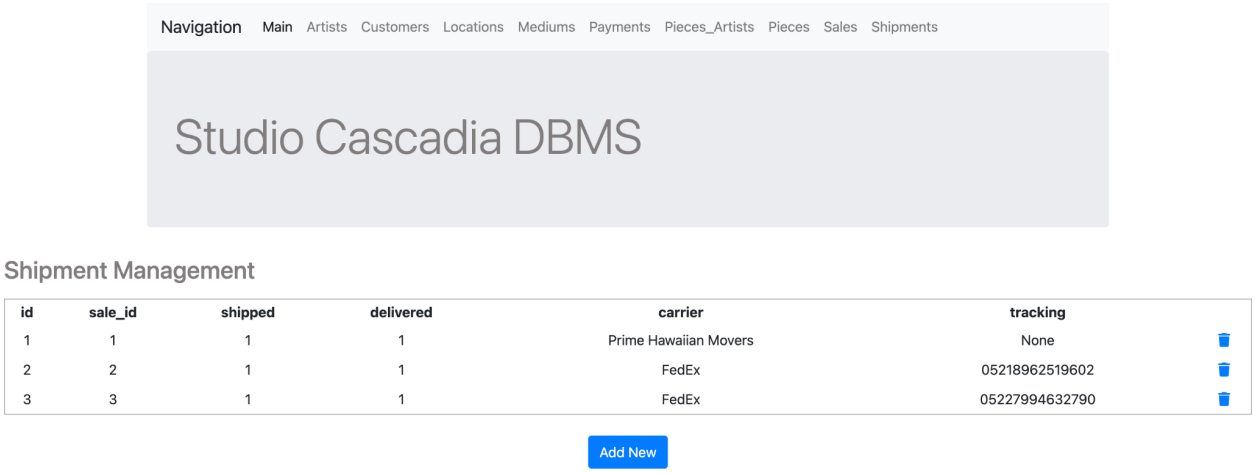
CREATE/READ/DELETE Sales Page:



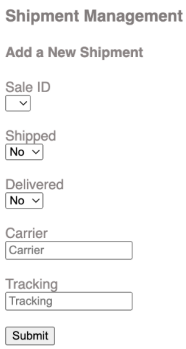
CREATE Sales Page:

Shipments

CREATE/READ/DELETE Shipments Page:



CREATE Shipments Page:



Citations

Navbar was adapted from Shovon, A. R. (2019, October 2). *Example of using Bootstrap's navbar in Flask*. Ahmedur Rahman Shovon. Retrieved May 11, 2022, from https://arshovon.com/snippets/flask-bootstrap-navbar/ as well as https://getbootstrap.com/docs/4.0/components/navbar/

The application was heavily influenced by material present in https://github.com/osu-cs340-ecampus/flask-starter-app hosted by osu-cs340-ecampus on https://github.com. Retrieved between May 5, 2022 and June 6, 2022.