Driveways

Mohit Sonwane, Dat Le, Michael Moore

Team 1

Table of Contents

Section 1. Project Requirements

1.1. Project Description

This project is a database application that is based around the idea of renting out user-owned driveways / parking spots. The project's main goal is to provide hosts with renters who are interested in parking at driveways on a daily basis. Hosts will have the ability to list their driveways for rent and modify their listings at any time. Renters will have the ability to search for the nearest location to park their cars. The project is trying to solve the problem of overcrowded parking lots in the Bay Area. Renters will have the opportunity to park in a neighborhood near their work-places. This could promote the use of public transport and create a more sustainable environment for working commuters. The stakeholders for this project would be the hosts and renters who are interested in either renting out their driveways or looking for a convenient parking spot with a price that is less than traditional parking lots. Through this project, we aim to provide a service where people can find a parking spot at their convenience. The application domain will be a business domain as we would generate profit through being a middle-man to help match the needs of both parties in the service.

1.2. Functional Requirements

This application provides functionality for different types of users, including hosts, renters, and admins. Hosts and renters will be the primary user. The primary user will be able to create an account and be able to list a driveway rental or rent a parking space from the rentees. The behaviors will change depending on the selection of roles when the user interacts with the application. Renters will have to input their basic information, car identification, and payment methods. The application will provide them with the list of closest parking spots when they search based on their location. The hosts will have the option to list their driveways on the application. The application will ask for the space description, address to approve a listing. They also have the authority to modify and update their listings at any time. Admins have access to remove, edit and add any listing at any time. This allows admins to maintain and test the application.

User Login / Sign-up

- The system will prompt a “guest” user to either login or signup, and will prompt the user with a form to do so.
- The first time users should sign-up inorder to access the application.
- A “Signed-in” user will be able to search and reserve listings, or list their own listing on the system.

Account creation

- There will be user accounts and admin accounts. Admin accounts will have additional features on top of the standard user account.
- There will be a form provided to the user by the system to input user details such as name, email, payment information, and car details.

Display Parking Information

- The user will be able to search an area for available parking spots (limited to the Bay Area in this project). They will be able to see price data, size of parking spots, available dates/times, and location.
- The system will display the parking spots on a map (Google Maps API), and the user will be able to click each spot to reveal a detailed view of that specific spot.

Host Parking Spot

- A user account will be able to list a parking spot that they own for “rent”.
- The user will need to provide details of the parking spot such as size of vehicle permitted, dates/times available.

Search for Parking Spot

- Users will be able to enter an address (within a valid area) and the system will display all the nearest parking spots. Also the available parking spots are up on the map GUI (See: “Display Parking Information”).

Update / Edit Listing

- Admin will be able to update and edit the current rental postings (add/remove place for rent, and modify)

User classification (host / guest)

- Non-admin users will be separated into hosts and renters based on their selection when accessing the web application.

Vehicle Identification

- Users will be able to provide car identification (ID) such as VIN, Color, Make, Model and license plate number.
- Users can add multiple vehicles into their accounts.

Parking Duration

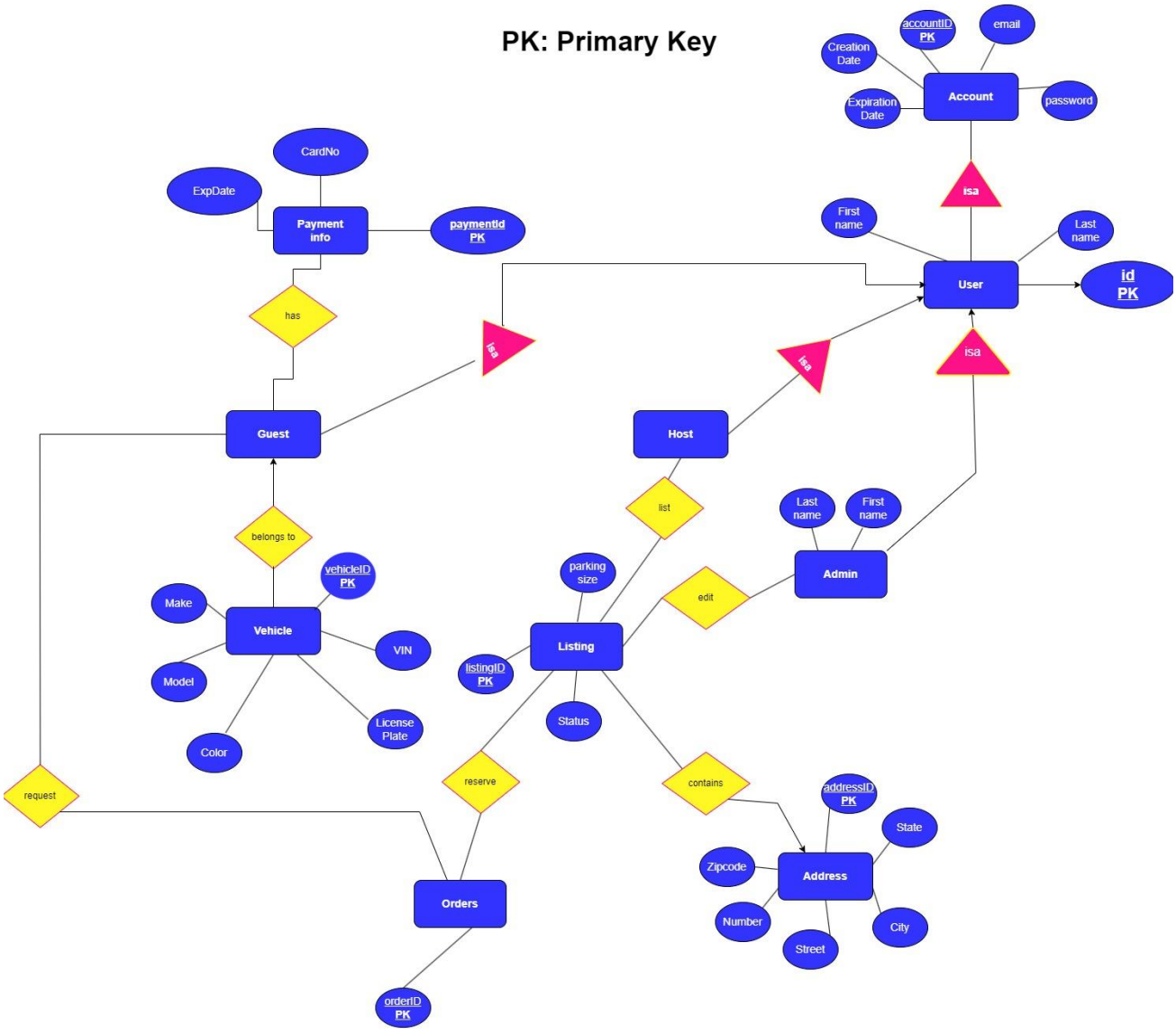
- The User’s while booking a parking spot can enter the date and time from the time of drop-off to pick-up.

Payments

- The user can enter their card details in order to book the parking on our website.

Section 2. Project Design

2.1. The ERD Diagram



2.2. Entity sets and relationships

PK = Primary Key

FK = Foreign Key

ERD = Entity Relationship Diagram

1. **Account Entity Set**

   The Account Entity set is the root entity set in our ERD. The PK is accountId.

   This table is responsible for creating(Sign-Up) and managing(Login) accounts in our application.

   Account Entity Relationship:

   - The Account handles the User and its PK is accountId. The User is a subclass of account and inherits its PK.

2. **User Entity Set**

   The User entity set is the parent of Guest, Host, Admin Entity Set. The PK is inherited from the Account called Id.

   This table manages different types of User's in our application.

   User Entity Relationship

   - The User handles Guest and its PK is accountId. The Guest is a subclass of User and inherits its PK.

- The User handles Host and its PK is accountId. The Host is a subclass of User and inherits its PK.
- The User handles Admin and its PK is accountId. The Admin is a subclass of User and inherits its PK.

3. **Guest Entity Set**

   The Guest Entity set checks the vehicle that belongs to the Guest, and requests orders for the Guest.

   The PK is accountId. The FK is vehicleId.

Guest Entity Relationship

- The Vehicle has a many-to-one relationship with the Guests. Each vehicle belongs to a specific Guest.The PK of Vehicle is vehicleId.
- The Guest has many-to-many relationships with orders. A guest can make multiple orders to book a parking space. The PK of Orders is orderId. The FK is listingId & vehicleId.
- The Guest is a subclass of User.

4. **Host Entity Set**

   The Host list's the parking spots in the listing. The PK is accountId. The FK is listingId.

   Host Entity Relationship

- The Host has many-to-many relationships with Listing. The ListingId is the PK. A host can host multiple parking spaces to the Users.
- The Host is a subclass of User.

5. **Admin Entity Set**

   The Admin edits the list. The PK is accountId.

   Admin Entity Relationship

- The Admin has many-to-many relationships with Listing. There can be multiple Listings modified or accessed by more than one user.
- The Admin is a subclass of User.

6. **Vehicle Entity Set**

   The Vehicle belongs to a specific Guest. The PK is vehicleId.

   Vehicle Entity Relationship

- The vehicle has a many-to-one relationship with the Guest. Each vehicle belongs to a specific guest. The PK of Guest is accountId.

7. **Listing Entity Set**

   Listing is the most important entity as it has relationships with many entity sets. Listing entity set manages available parking space and its fares for both the host and user.The PK is listingId. The FK is addressId.

   Listing Entity Relationship

- The listing is listed by a host that has a many-to-many relationship. A host can host multiple parking spaces in listings. The PK of the host is accountId.

- The listing can be edited by Admin which has many-to-many relationships. An admin can modify numerous listings. The PK of Admin is accountId.
- The listing contains the address of a one-to-many relationship. Thus a listing ID corresponds to at most one address.The PK of Address is addressId
- The listing reserves orders has many-to-many relationships. The PK of orders is orderId.

8. Order Entity Set

The Order Entity Set manages the parking request of the Guest User, and reserves the order of listing for Host users. The PK is orderId. The FK is listingId and vehicleId.

Order Entity Set Relationship

- The Orders requested by Guest is a many-to-many relationship. A guest can request multiple orders for parking cars. The PK of Guest is accountId.
- The order reserved by listing has a many-to-many relationship. A listing can be ordered by different users at different times.The PK of Listing is listingId.

9. Address Entity Set

The Address Entity set manages the address of the specific listing. The PK is addressId.

Address Entity Set Relationship

- The address is contained by Listing is a one-to-many relationship. Every listing where a car will be parked will have only a single address.

10. Payment Info Entity Set

The Payment Info Details manages the payment details for guests. The PK is paymentId.

Payment Info Entity Set Relationship

- Every guest has payment information. It has many-to-many relation with guests.

2.3. ERD Schemas

Account(**accountId**, email, password, creationDate, expirationDate)

user(**id,** first_name, last_name, **accountID**)

guest(**guestId, vehicleId**)

PaymentInfo(cardNo, ExpDate, **paymentId**);

host(**hostId, listingId**)

admin(**adminId,** lastName, firstName)

Vehicle(**vehicleId,** Make, Model, Color, VIN, license_plate)

Listing(**ListingId,** parking_size, status)

order(**orderId, listingId**)

Address(**addressId,** Country, State, City, ZipCode, street, number)

belongTo(**vehicleId**, **id**)

edit(**id**, **listingId**)

list(**id**, **listingId**)

requests(**guestId, orderId**)

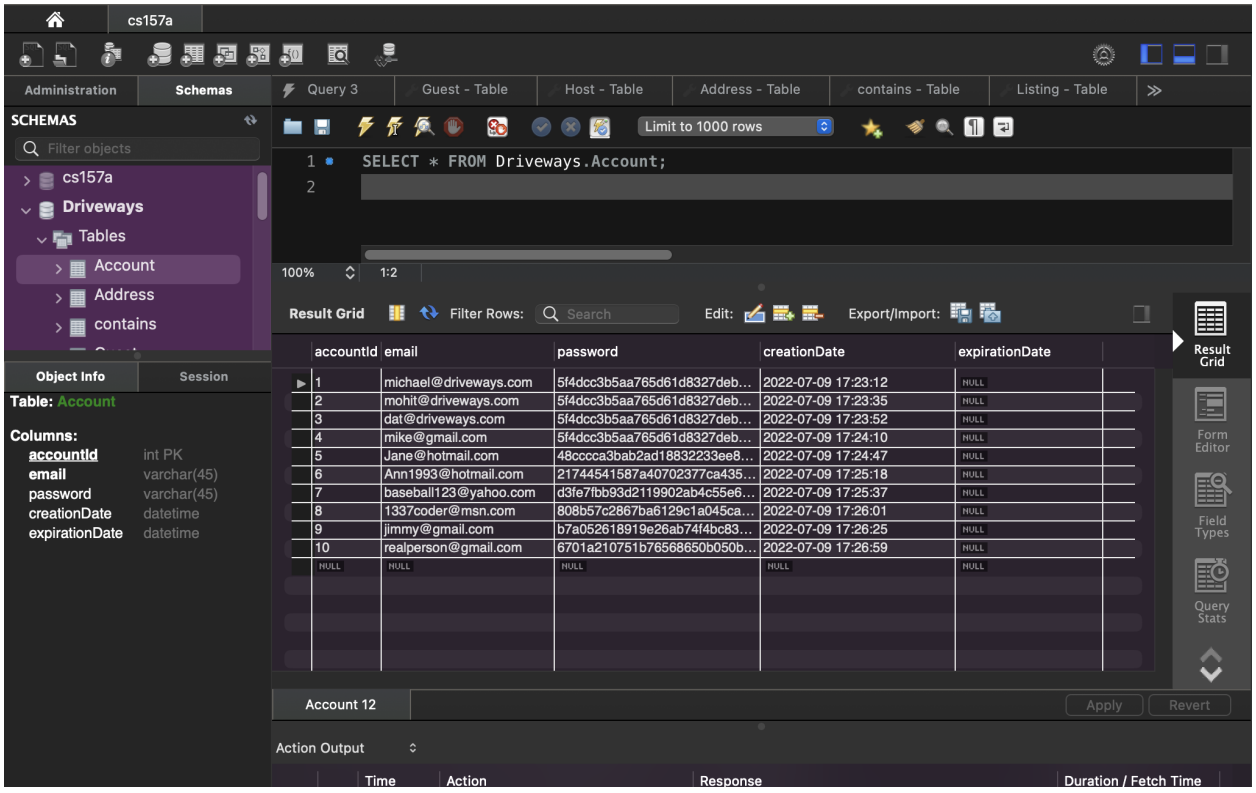
reserves(**orderId**, **listingId**)

contains(**listingId**, **addressId**)

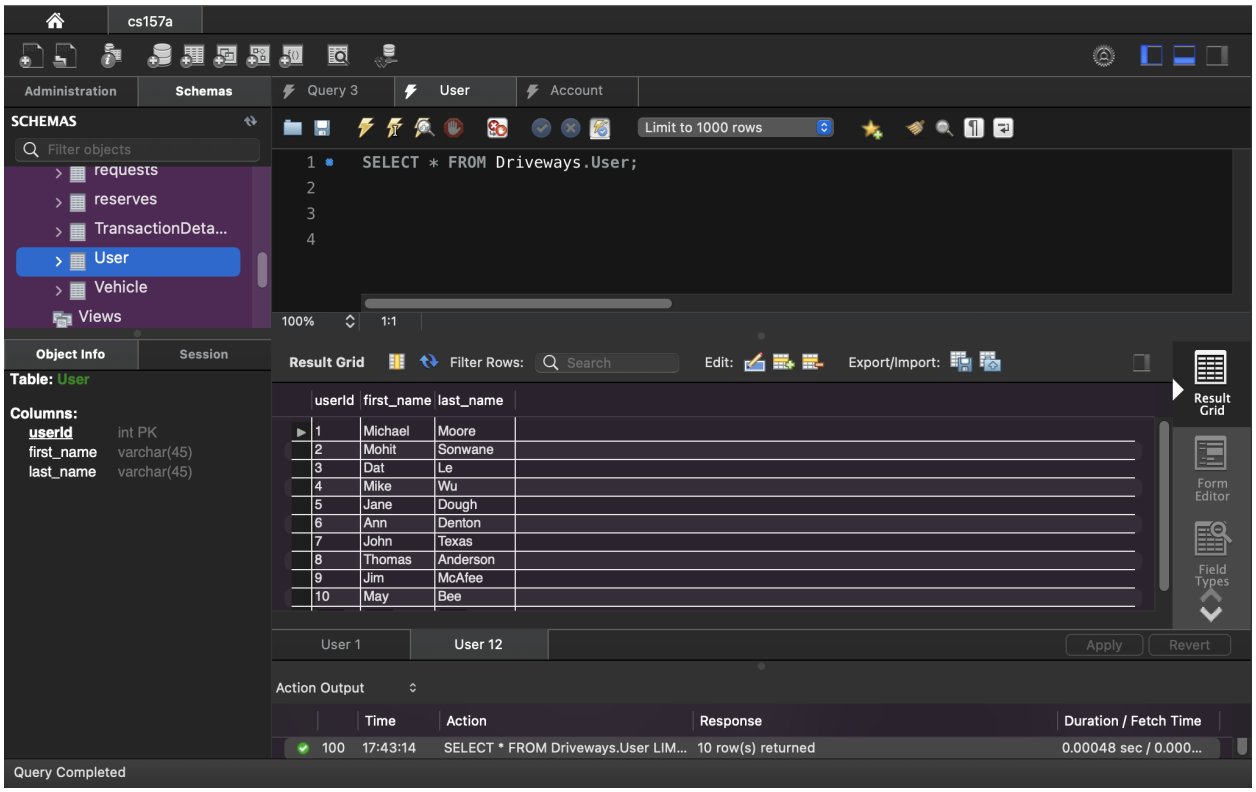
has(**paymentId**, **id**)

2.4. MySQL Workbench contents

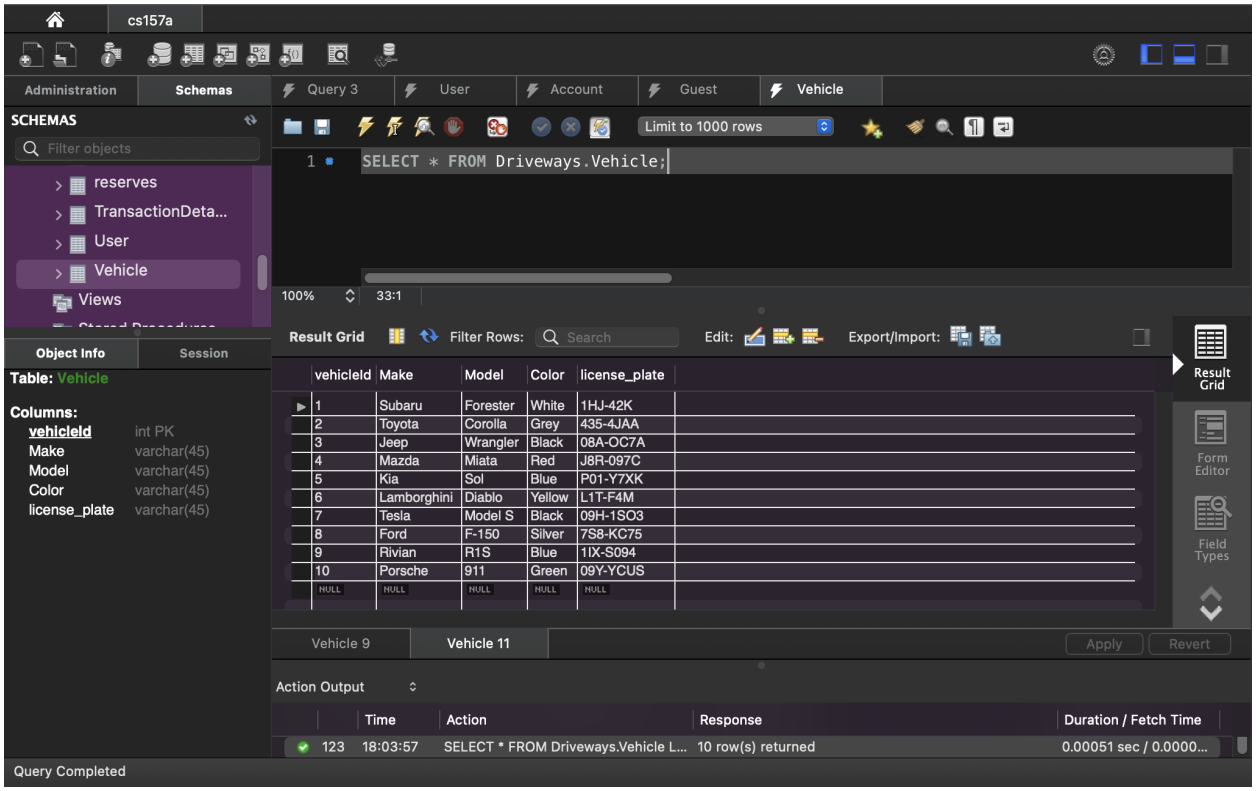
Account:



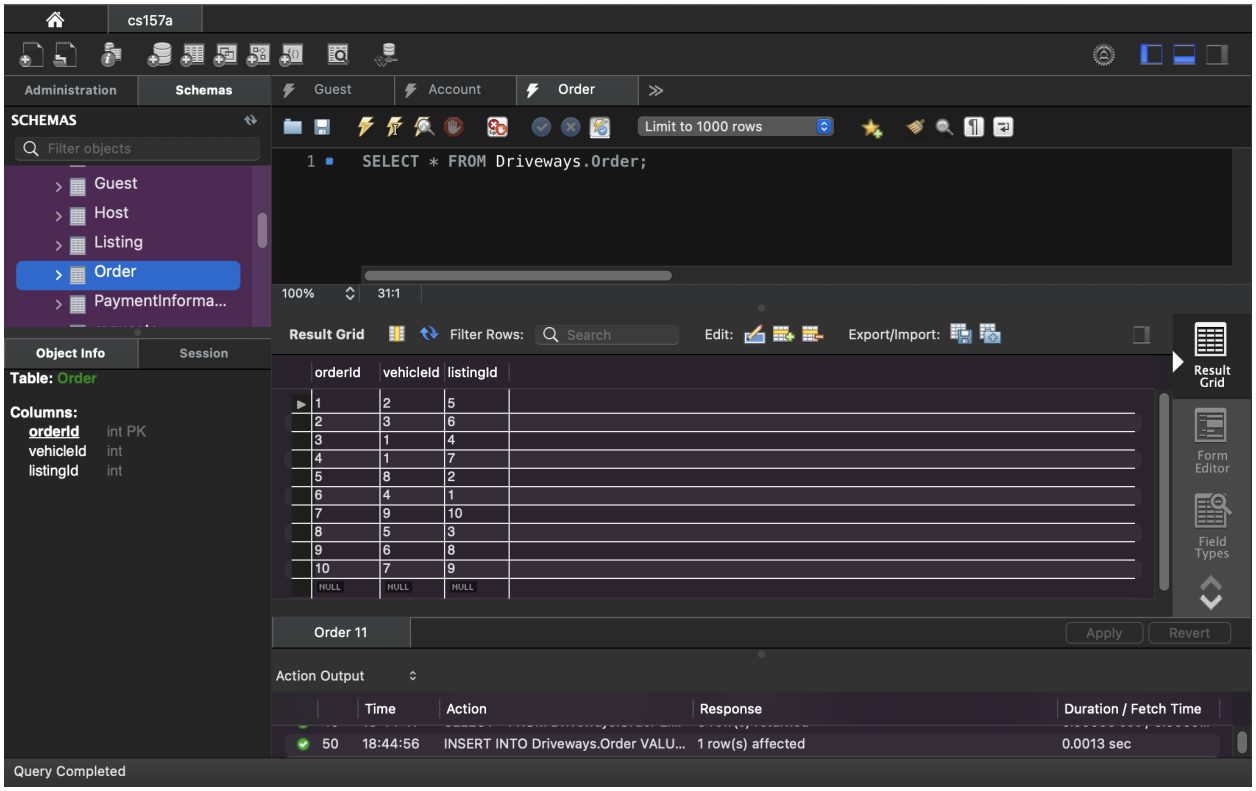
User:


cs157a
Administration
Schemas
SCHEMAS
Filter objects
requests
reserves
TransactionDeta...
User
Vehicle
Views
Object Info
Session
Table: User
Columns:
userId int PK
first_name varchar(45)
last_name varchar(45)
Query 3
User
Account
Limit to 1000 rows
1 SELECT * FROM Driveways.User;
2
3
4
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
userId first_name last_name
1 Michael Moore
2 Mohit Sonwane
3 Dat Le
4 Mike Wu
5 Jane Dough
6 Ann Denton
7 John Texas
8 Thomas Anderson
9 Jim McAfee
10 May Bee
Result Grid
Form Editor
Field Types
User 1
User 12
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
100 17:43:14 SELECT * FROM Driveways.User LIM... 10 row(s) returned 0.00048 sec / 0.000...
Query Completed

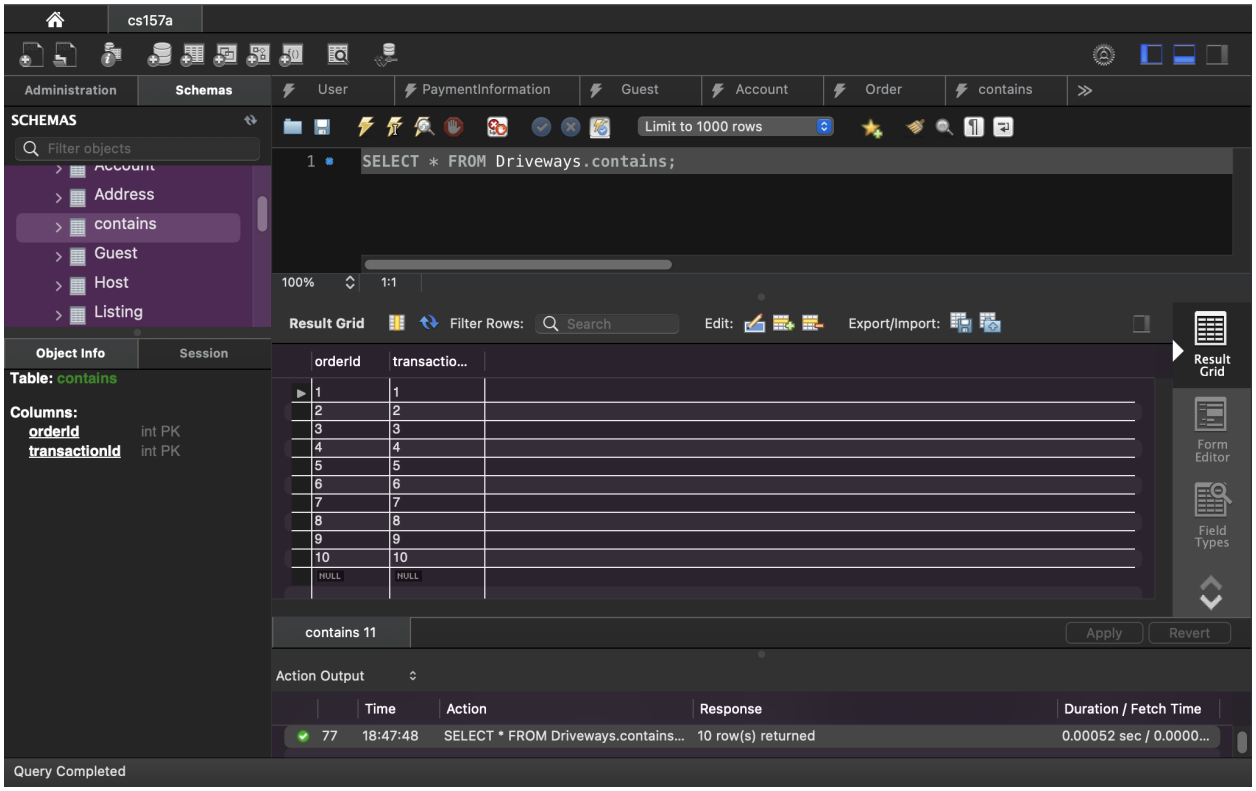
Vehicle:


cs157a
Administration
Schemas
Query 3
User
Account
Guest
Vehicle
SCHEMAS
Filter objects
reserves
TransactionDeta...
User
Vehicle
Views
Limit to 1000 rows
1 SELECT * FROM Driveways.Vehicle;
100%
33:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
Object Info
Session
Table: Vehicle
Columns:
vehicleId int PK
Make varchar(45)
Model varchar(45)
Color varchar(45)
license_plate varchar(45)
vehicleId Make Model Color license_plate
1 Subaru Forester White 1HJ-42K
2 Toyota Corolla Grey 435-4JAA
3 Jeep Wrangler Black 08A-OC7A
4 Mazda Miata Red J8R-097C
5 Kia Sol Blue P01-Y7XK
6 Lamborghini Diablo Yellow L1T-F4M
7 Tesla Model S Black 09H-1SO3
8 Ford F-150 Silver 7S8-KC75
9 Rivian R1S Blue 1IX-S094
10 Porsche 911 Green 09Y-YCUS
Result Grid
Form Editor
Field Types
Vehicle 9
Vehicle 11
Apply
Revert
Action Output
Time Action Response Duration / Fetch Time
123 18:03:57 SELECT * FROM Driveways.Vehicle L... 10 row(s) returned 0.00051 sec / 0.0000...
Query Completed

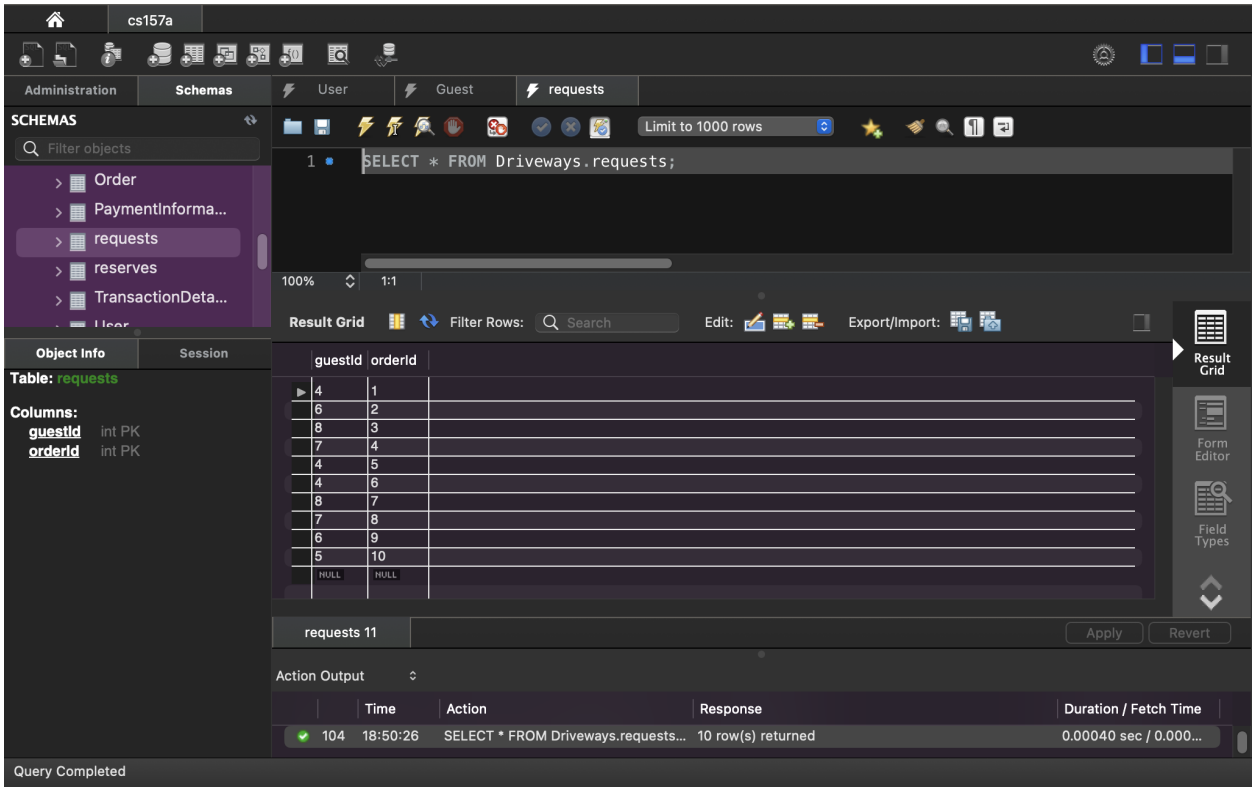
Order:


cs157a
Administration
Schemas
Guest
Account
Order
SCHEMAS
Filter objects
Guest
Host
Listing
Order
PaymentInforma...
Limit to 1000 rows
1 SELECT * FROM Driveways.Order;
100%
31:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
orderId vehicleId listingId
1 2 5
2 3 6
3 1 4
4 1 7
5 8 2
6 4 1
7 9 10
8 5 3
9 6 8
10 7 9
NULL NULL NULL
Result Grid
Form Editor
Field Types
Object Info
Session
Table: Order
Columns:
orderId int PK
vehicleId int
listingId int
Order 11
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
50 18:44:56 INSERT INTO Driveways.Order VALU... 1 row(s) affected 0.0013 sec
Query Completed

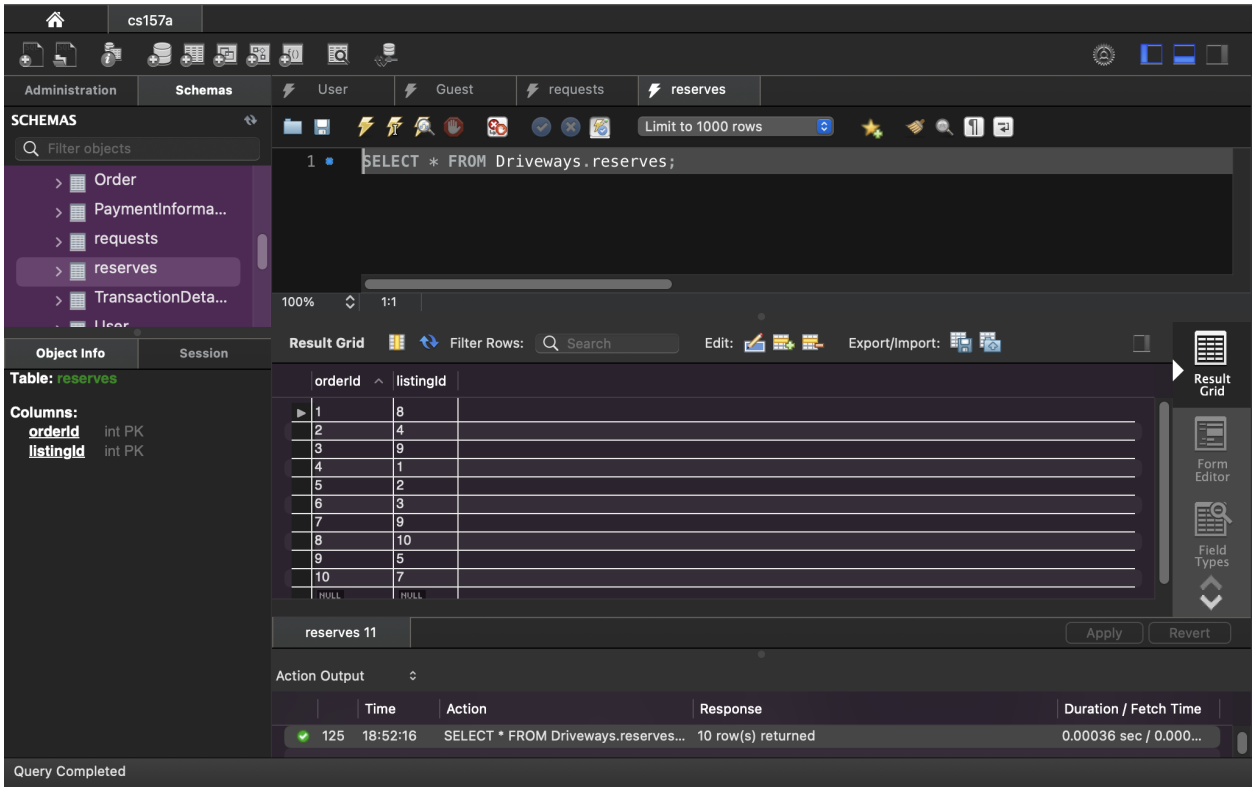
Contains:


cs157a
Administration
Schemas
SCHEMAS
Filter objects
Address
contains
Guest
Host
Listing
Object Info
Session
Table: contains
Columns:
orderId int PK
transactionId int PK
User
PaymentInformation
Guest
Account
Order
contains
Limit to 1000 rows
1 SELECT * FROM Driveways.contains;
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
orderId
transactio...
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
9 9
10 10
NULL NULL
Result Grid
Form Editor
Field Types
contains 11
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
77 18:47:48 SELECT * FROM Driveways.contains... 10 row(s) returned 0.00052 sec / 0.0000...
Query Completed

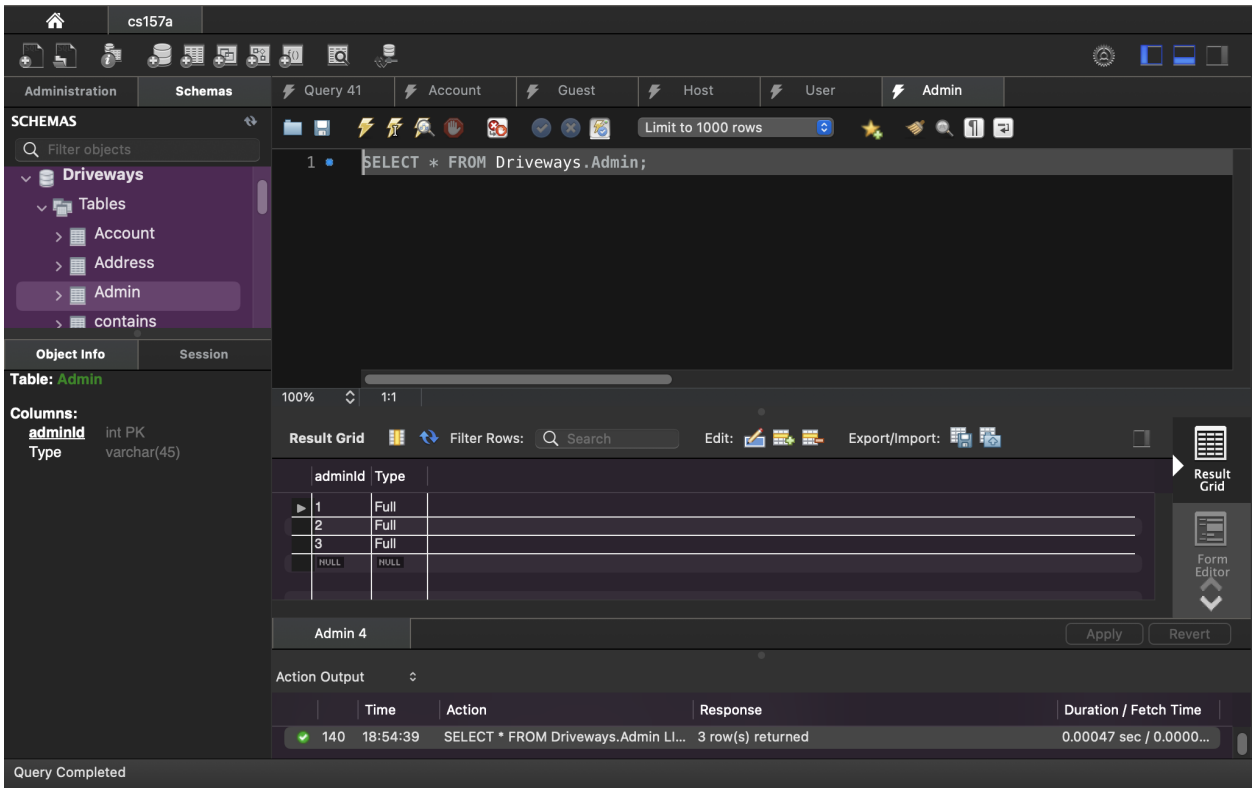
requests:


cs157a
Administration
Schemas
SCHEMAS
Filter objects
Order
PaymentInforma...
requests
reserves
TransactionDeta...
Object Info
Session
Table: requests
Columns:
guestId int PK
orderId int PK
User
Guest
requests
Limit to 1000 rows
1 SELECT * FROM Driveways.requests;
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
guestId orderId
4 1
6 2
8 3
7 4
4 5
4 6
8 7
7 8
6 9
5 10
NULL NULL
Result Grid
Form Editor
Field Types
requests 11
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
104 18:50:26 SELECT * FROM Driveways.requests... 10 row(s) returned 0.00040 sec / 0.000...
Query Completed

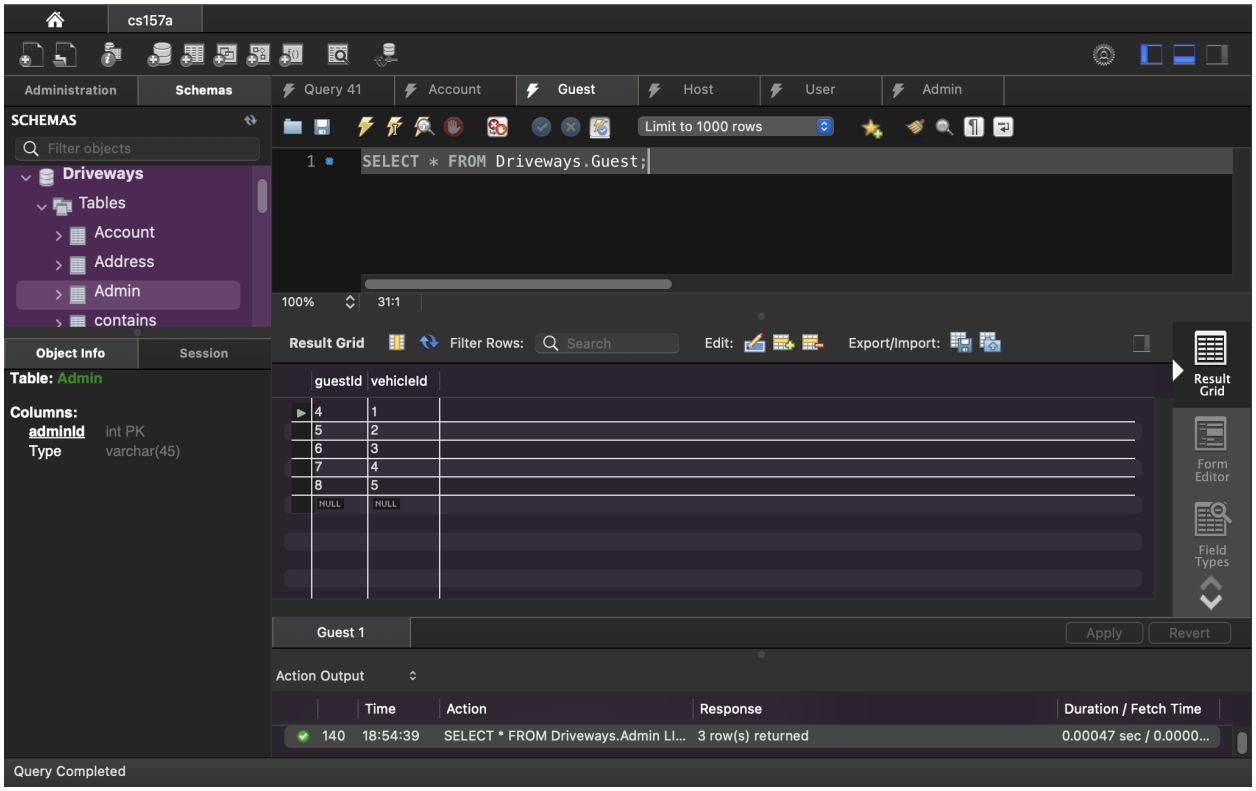
Reserves:


cs157a
Administration
Schemas
SCHEMAS
Filter objects
Order
PaymentInforma...
requests
reserves
TransactionDeta...
Object Info
Session
Table: reserves
Columns:
orderId int PK
listingId int PK
User
Guest
requests
reserves
Limit to 1000 rows
1 SELECT * FROM Driveways.reserves;
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
orderId
listingId
1 8
2 4
3 9
4 1
5 2
6 3
7 9
8 10
9 5
10 7
Result Grid
Form Editor
Field Types
reserves 11
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
125 18:52:16 SELECT * FROM Driveways.reserves... 10 row(s) returned 0.00036 sec / 0.000...
Query Completed

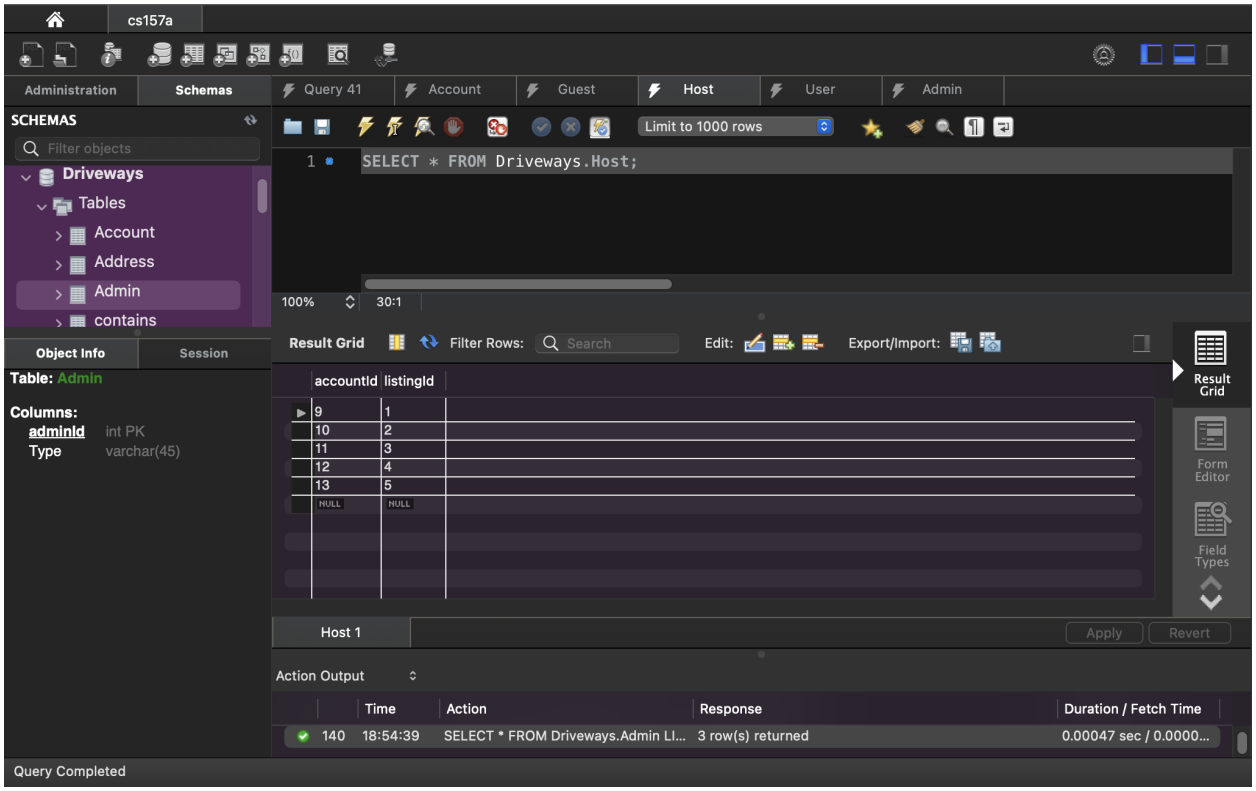
Admin:


cs157a
Administration
Schemas
Query 41
Account
Guest
Host
User
Admin
SCHEMAS
Filter objects
Driveways
Tables
Account
Address
Admin
contains
Limit to 1000 rows
1 SELECT * FROM Driveways.Admin;
Object Info
Session
Table: Admin
Columns:
adminId int PK
Type varchar(45)
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
adminId
Type
1
Full
2
Full
3
Full
NULL
NULL
Result Grid
Form Editor
Admin 4
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
140
18:54:39
SELECT * FROM Driveways.Admin LI...
3 row(s) returned
0.00047 sec / 0.0000...
Query Completed

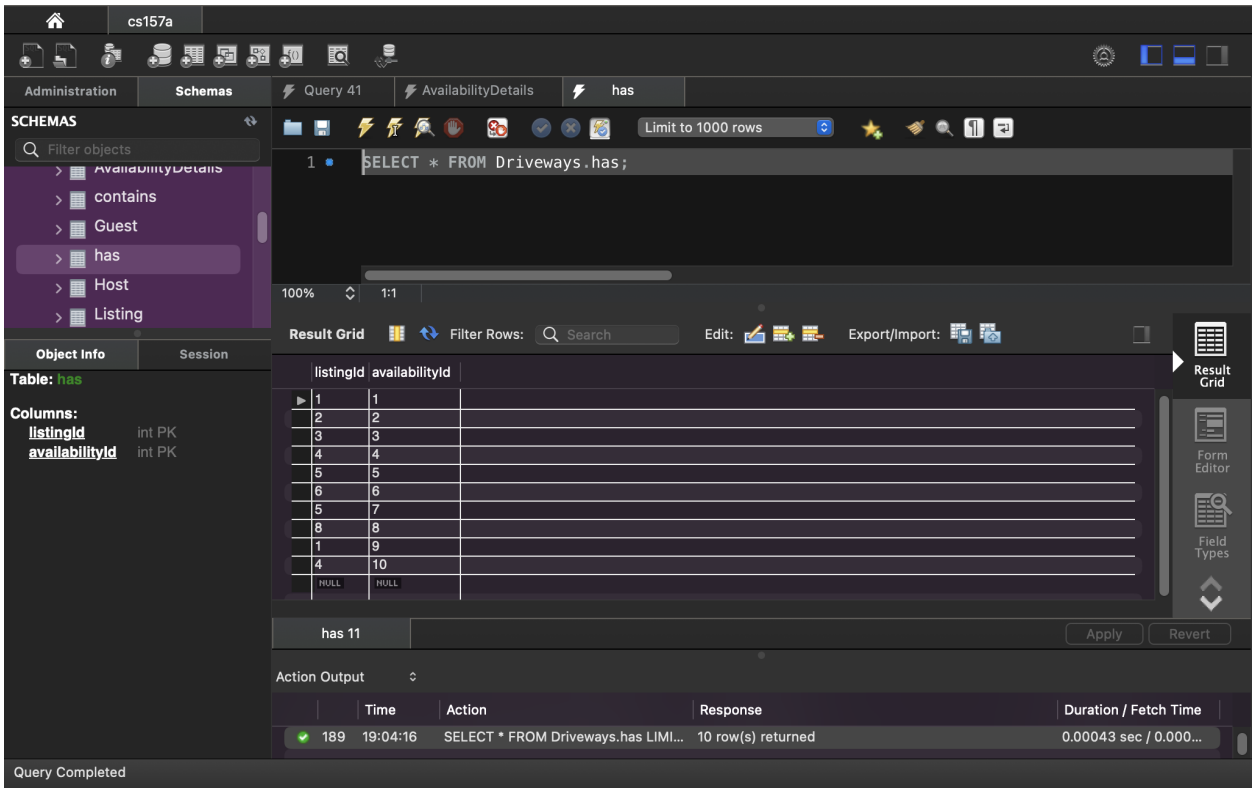
Guest:


cs157a
Administration
Schemas
Query 41
Account
Guest
Host
User
Admin
SCHEMAS
Filter objects
Driveways
Tables
Account
Address
Admin
contains
Limit to 1000 rows
1
SELECT * FROM Driveways.Guest;
100%
31:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
Object Info
Session
Table: Admin
Columns:
adminId int PK
Type varchar(45)
guestId
vehicleId
4 1
5 2
6 3
7 4
8 5
NULL NULL
Result Grid
Form Editor
Field Types
Guest 1
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
140 18:54:39 SELECT * FROM Driveways.Admin LI... 3 row(s) returned 0.00047 sec / 0.0000...
Query Completed

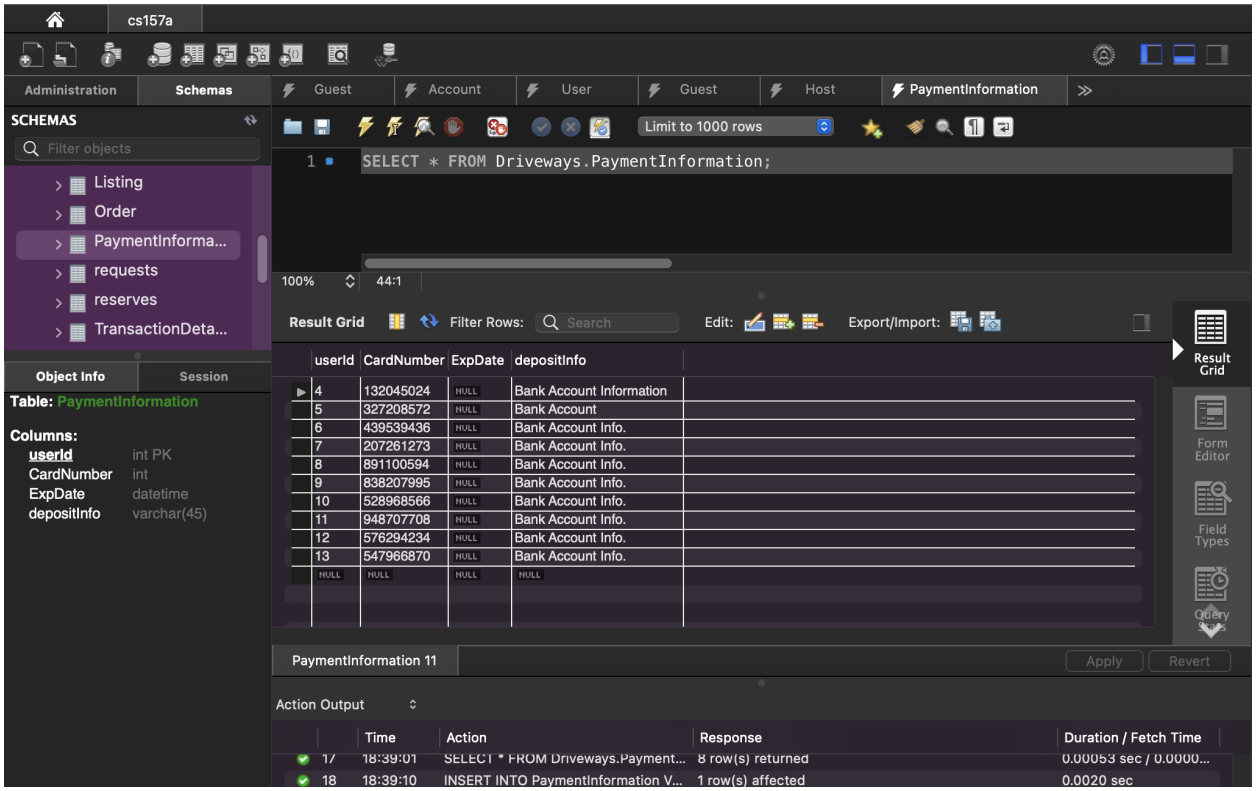
Host:


cs157a
Administration
Schemas
Query 41
Account
Guest
Host
User
Admin
SCHEMAS
Filter objects
Driveways
Tables
Account
Address
Admin
contains
Limit to 1000 rows
1
SELECT * FROM Driveways.Host;
100%
30:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
Object Info
Session
Table: Admin
Columns:
adminId int PK
Type varchar(45)
accountId listingId
9 1
10 2
11 3
12 4
13 5
NULL NULL
Result Grid
Form Editor
Field Types
Host 1
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
140 18:54:39 SELECT * FROM Driveways.Admin LI... 3 row(s) returned 0.00047 sec / 0.0000...
Query Completed

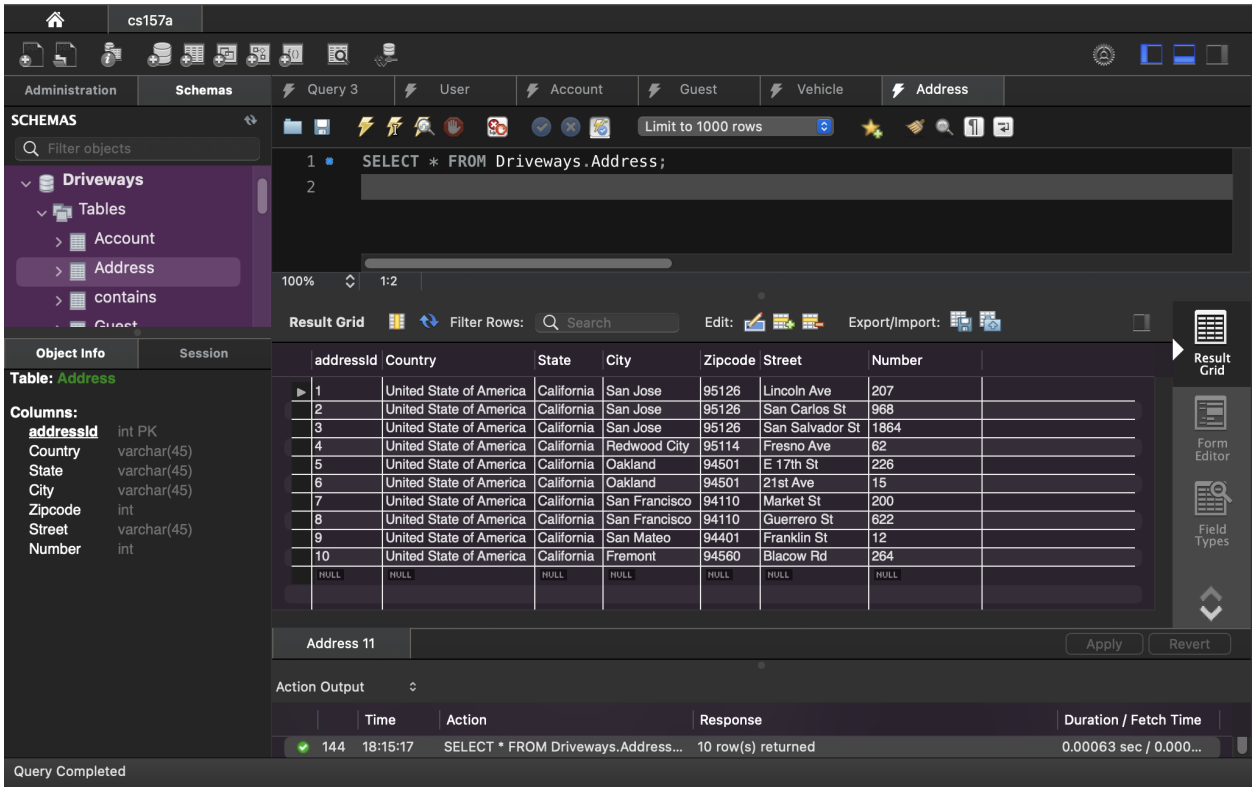
Has:


cs157a
Administration
Schemas
Query 41
AvailabilityDetails
has
SCHEMAS
Filter objects
contains
Guest
has
Host
Listing
Limit to 1000 rows
1 SELECT * FROM Driveways.has;
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
listingId availabilityId
1 1
2 2
3 3
4 4
5 5
6 6
5 7
8 8
1 9
4 10
NULL NULL
Result Grid
Form Editor
Field Types
Object Info
Session
Table: has
Columns:
listingId int PK
availabilityId int PK
has 11
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
189 19:04:16 SELECT * FROM Driveways.has LIMI... 10 row(s) returned 0.00043 sec / 0.000...
Query Completed

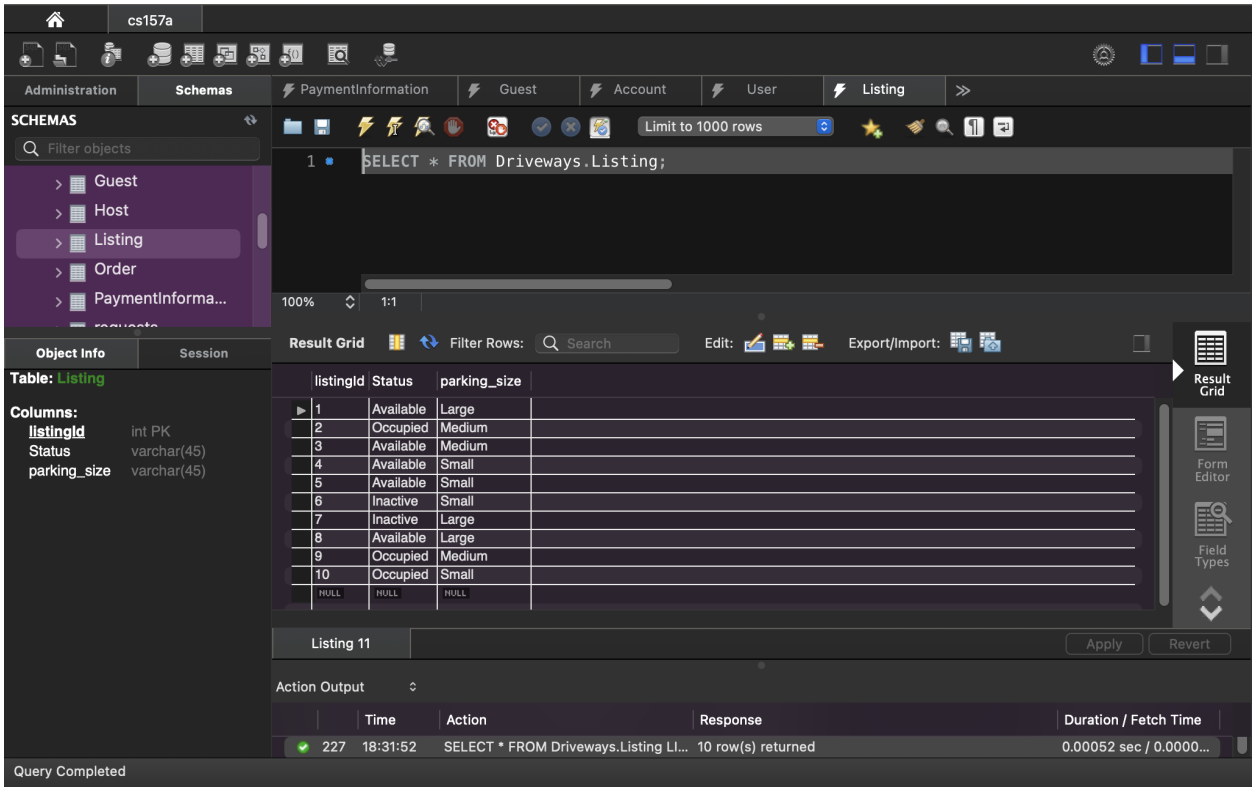
Payment Information:



Address:


cs157a
Administration
Schemas
Query 3
User
Account
Guest
Vehicle
Address
SCHEMAS
Filter objects
Driveways
Tables
Account
Address
contains
Limit to 1000 rows
1 SELECT * FROM Driveways.Address;
2
100%
1:2
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
addressId Country State City Zipcode Street Number
1 United State of America California San Jose 95126 Lincoln Ave 207
2 United State of America California San Jose 95126 San Carlos St 968
3 United State of America California San Jose 95126 San Salvador St 1864
4 United State of America California Redwood City 95114 Fresno Ave 62
5 United State of America California Oakland 94501 E 17th St 226
6 United State of America California Oakland 94501 21st Ave 15
7 United State of America California San Francisco 94110 Market St 200
8 United State of America California San Francisco 94110 Guerrero St 622
9 United State of America California San Mateo 94401 Franklin St 12
10 United State of America California Fremont 94560 Blacow Rd 264
Result Grid
Form Editor
Field Types
Object Info
Session
Table: Address
Columns:
addressId int PK
Country varchar(45)
State varchar(45)
City varchar(45)
Zipcode int
Street varchar(45)
Number int
Address 11
Apply
Revert
Action Output
Time Action Response Duration / Fetch Time
144 18:15:17 SELECT * FROM Driveways.Address... 10 row(s) returned 0.00063 sec / 0.000...
Query Completed

Listing:

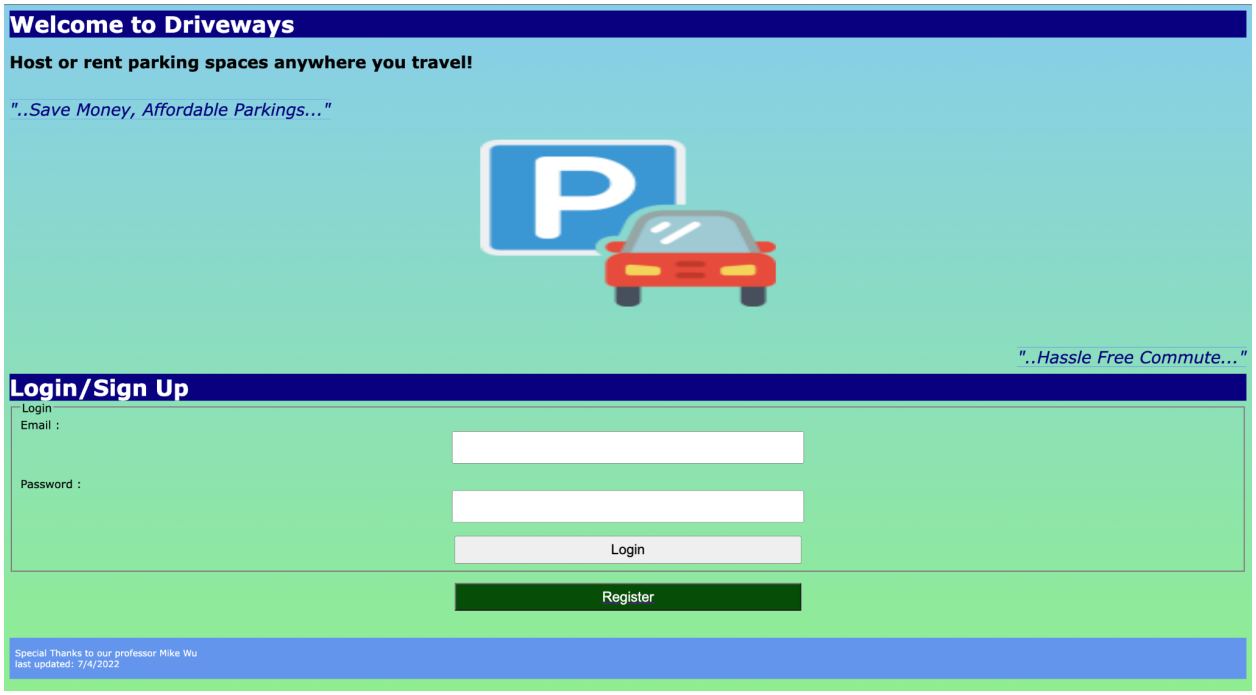

cs157a
Administration
Schemas
SCHEMAS
Filter objects
Guest
Host
Listing
Order
PaymentInforma...
Object Info
Session
Table: Listing
Columns:
listingId int PK
Status varchar(45)
parking_size varchar(45)
PaymentInformation
Guest
Account
User
Listing
Limit to 1000 rows
1 SELECT * FROM Driveways.Listing;
100%
1:1
Result Grid
Filter Rows:
Search
Edit:
Export/Import:
listingId Status parking_size
1 Available Large
2 Occupied Medium
3 Available Medium
4 Available Small
5 Available Small
6 Inactive Small
7 Inactive Large
8 Available Large
9 Occupied Medium
10 Occupied Small
Result Grid
Form Editor
Field Types
Listing 11
Apply
Revert
Action Output
Time Action Response Duration / Fetch Time
227 18:31:52 SELECT * FROM Driveways.Listing LI... 10 row(s) returned 0.00052 sec / 0.0000...
Query Completed

Section 3. Project Implementation

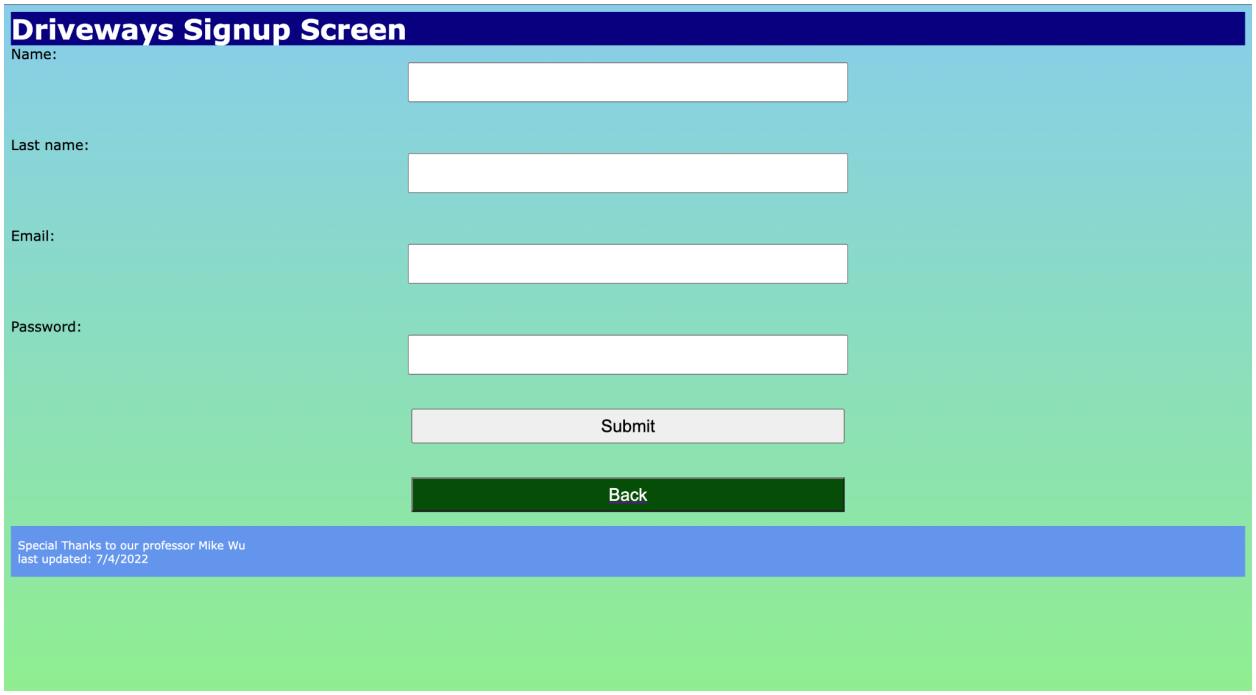
3.1. Real-time running of functional requirements

3.1.1. User Login / Sign-up

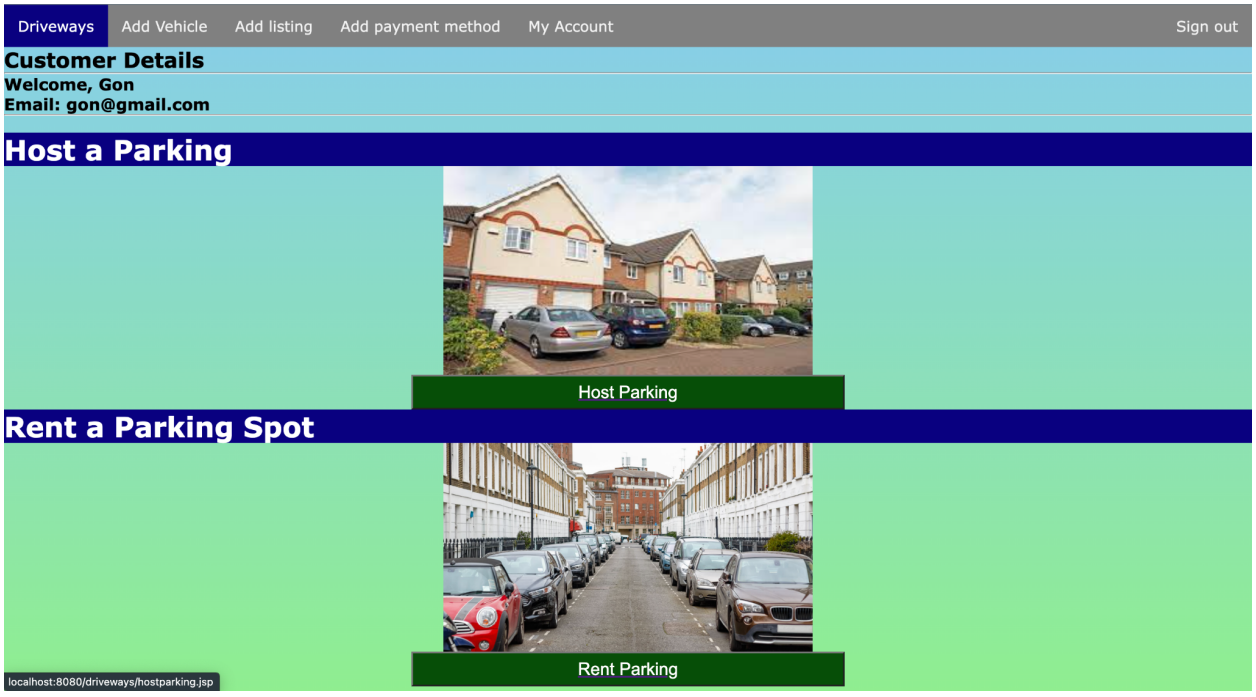
The following picture is the main page of the web application:



The following picture shows the registration page if the user has not registered yet:

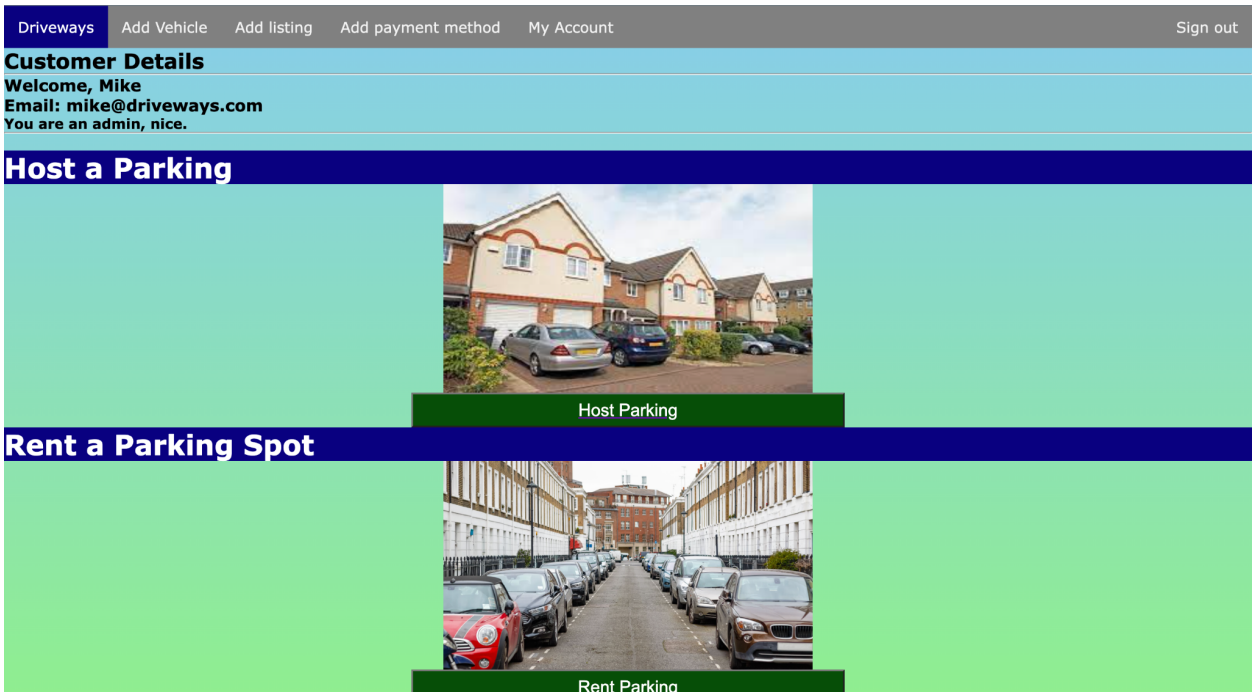


The following snapshot shows how a user successfully logged into the dashboard:



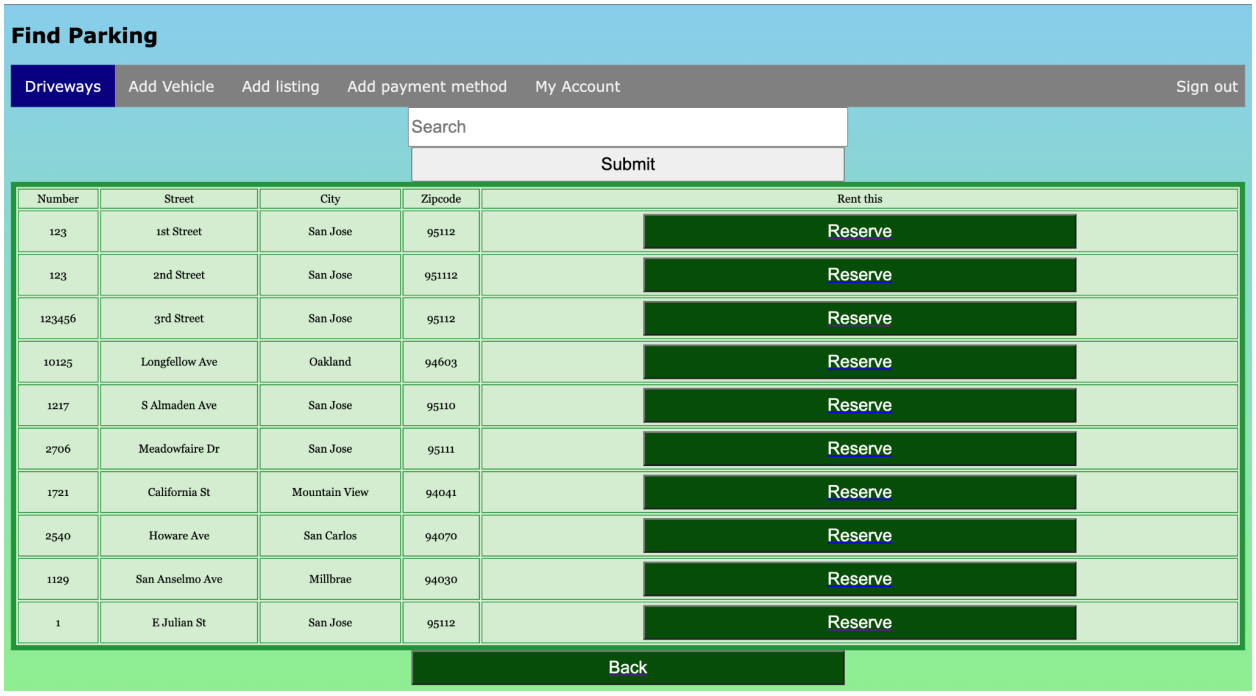
3.1.2. Account creation

The snapshot shows what it looks like to log in as an admin:

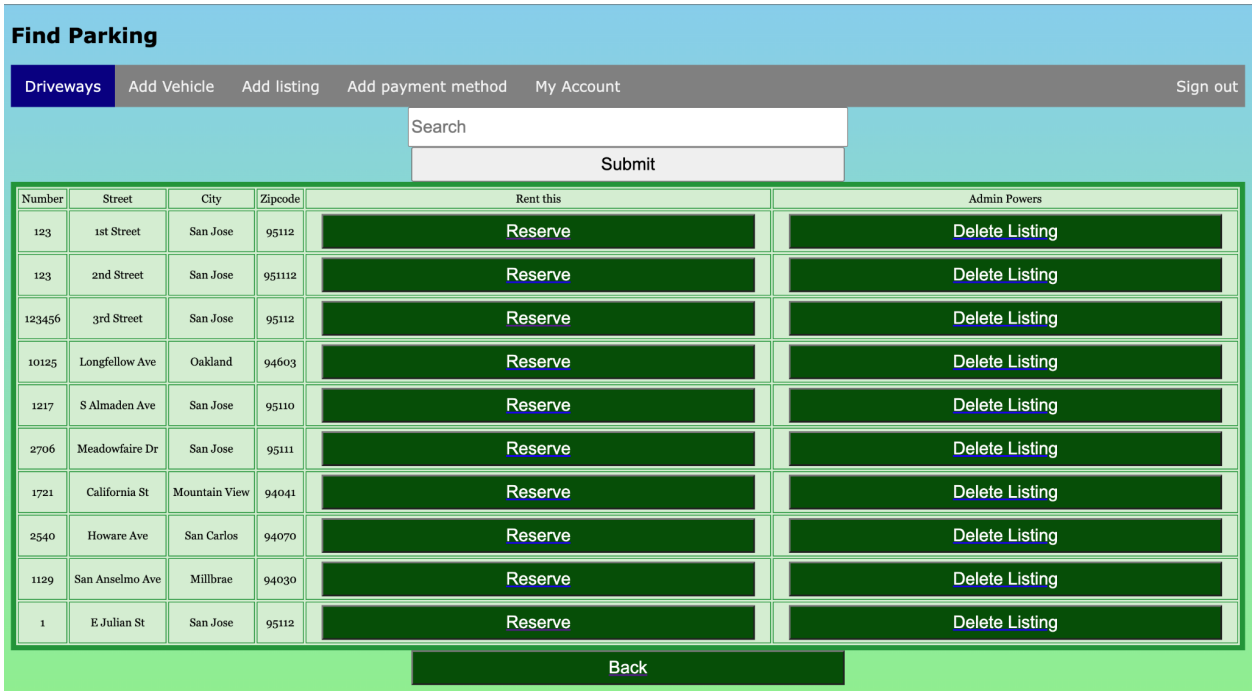


The snapshot shows what admin can do and user cannot to a listing on listing page:

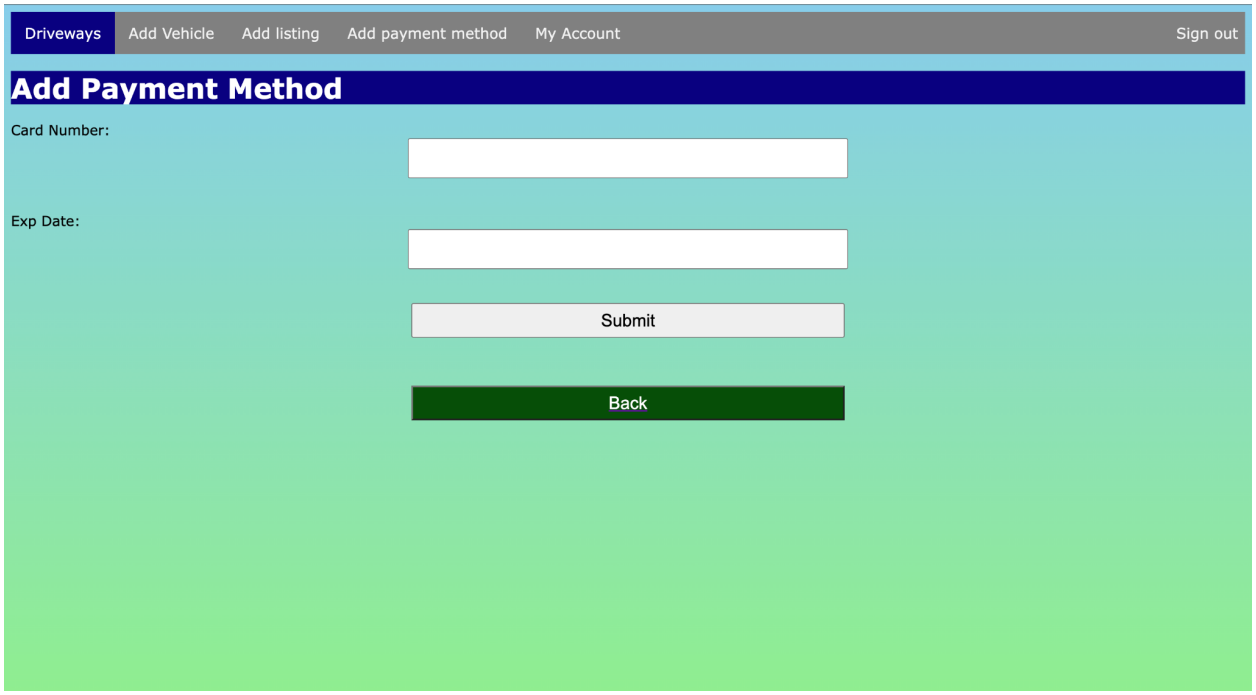
User Perspective:

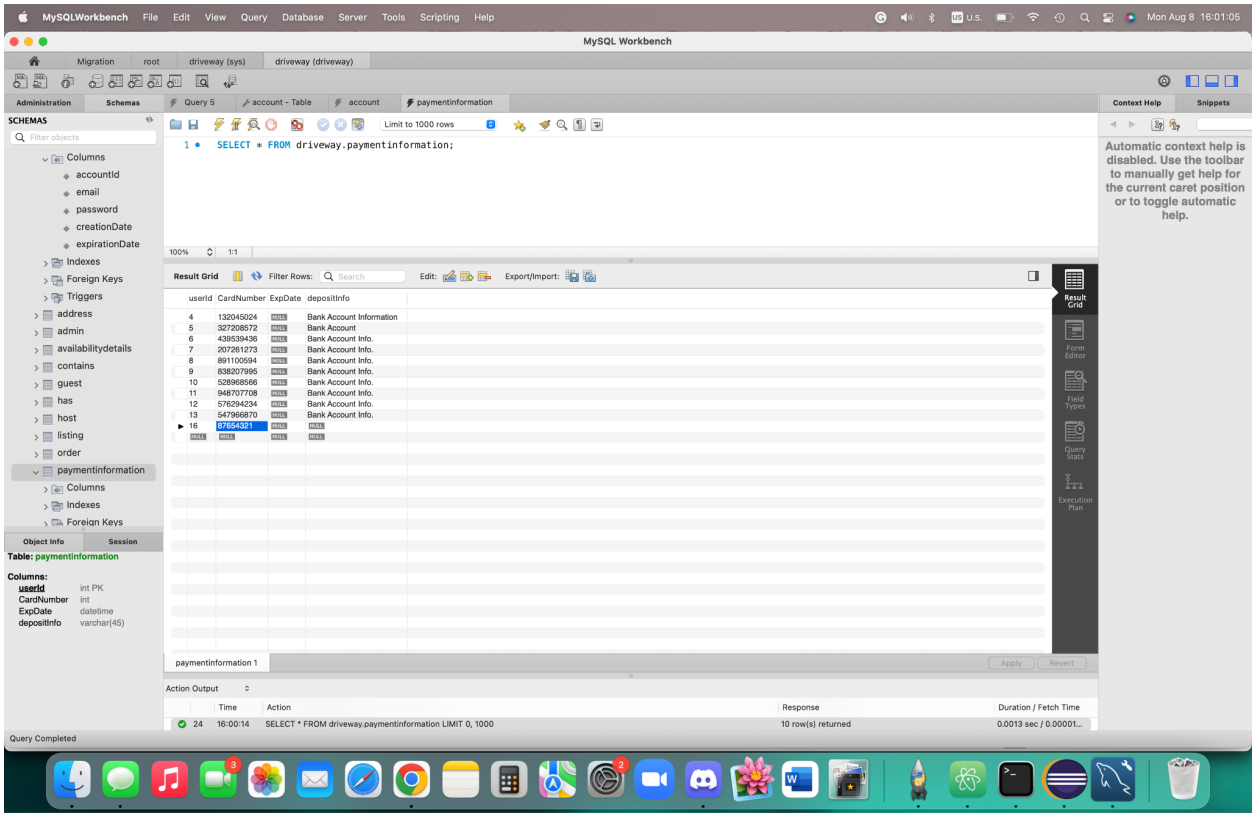


Admin Perspective:



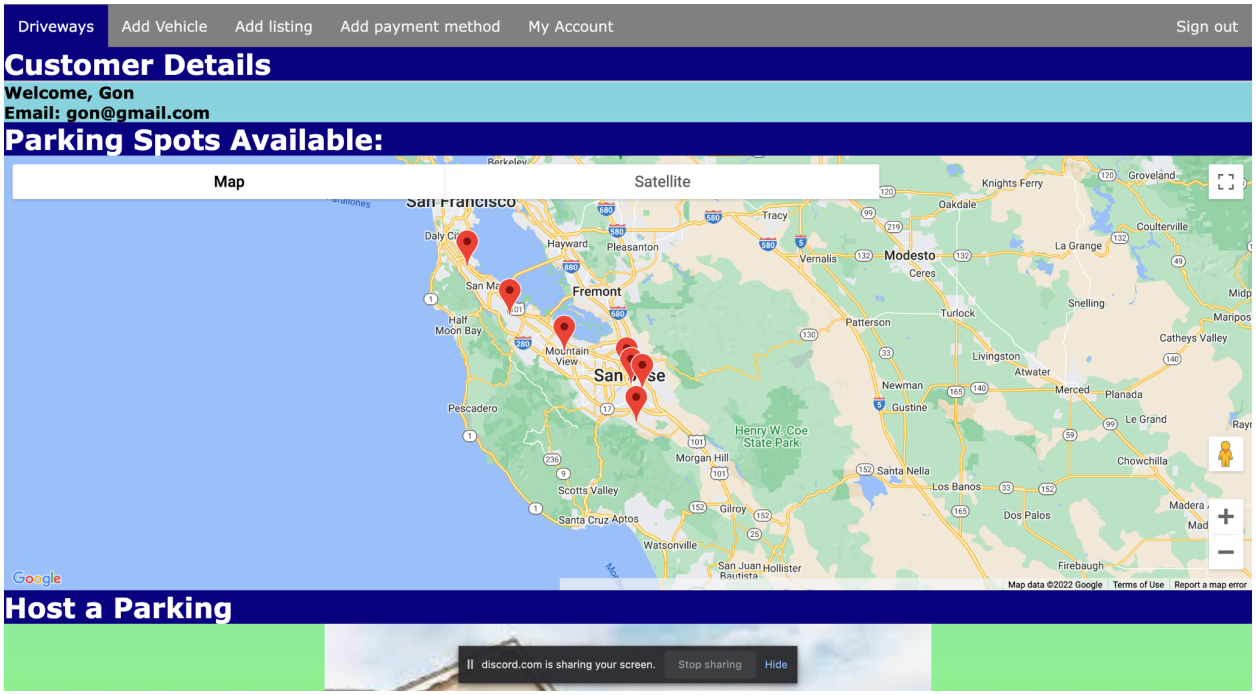
The snapshots show payment information requested and contained in the database:





3.1.3. Display parking information

The snapshot shows the Google API implementation on the dashboard page:

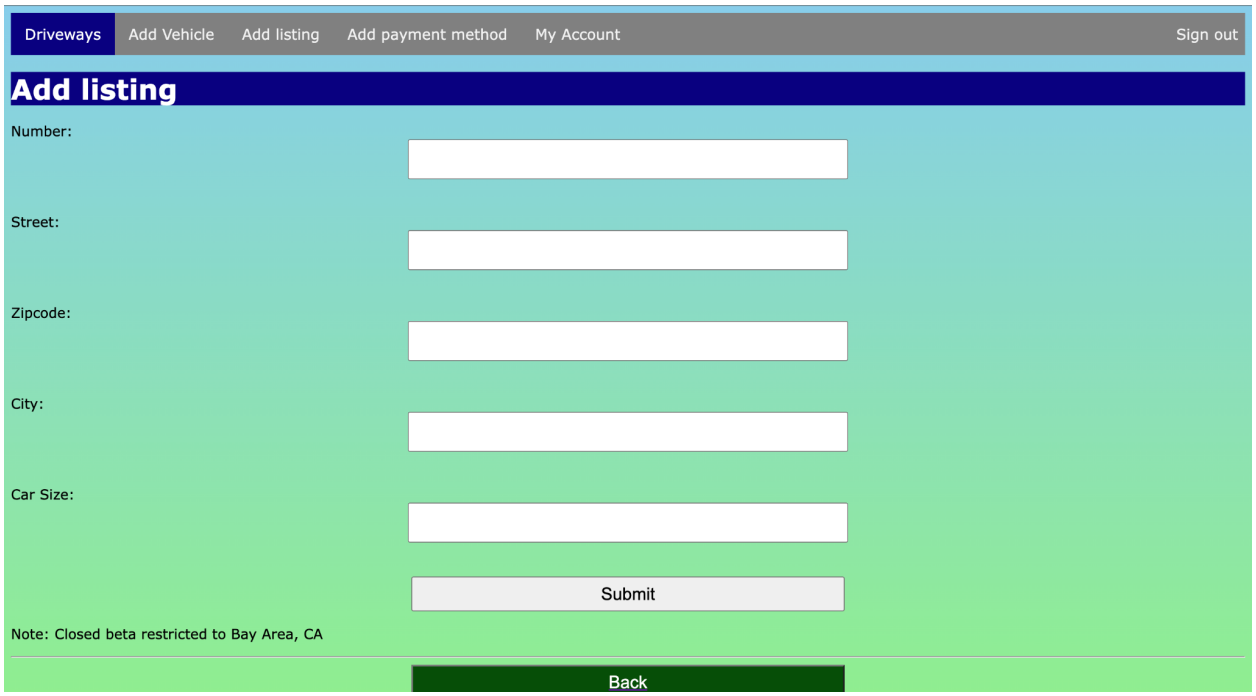


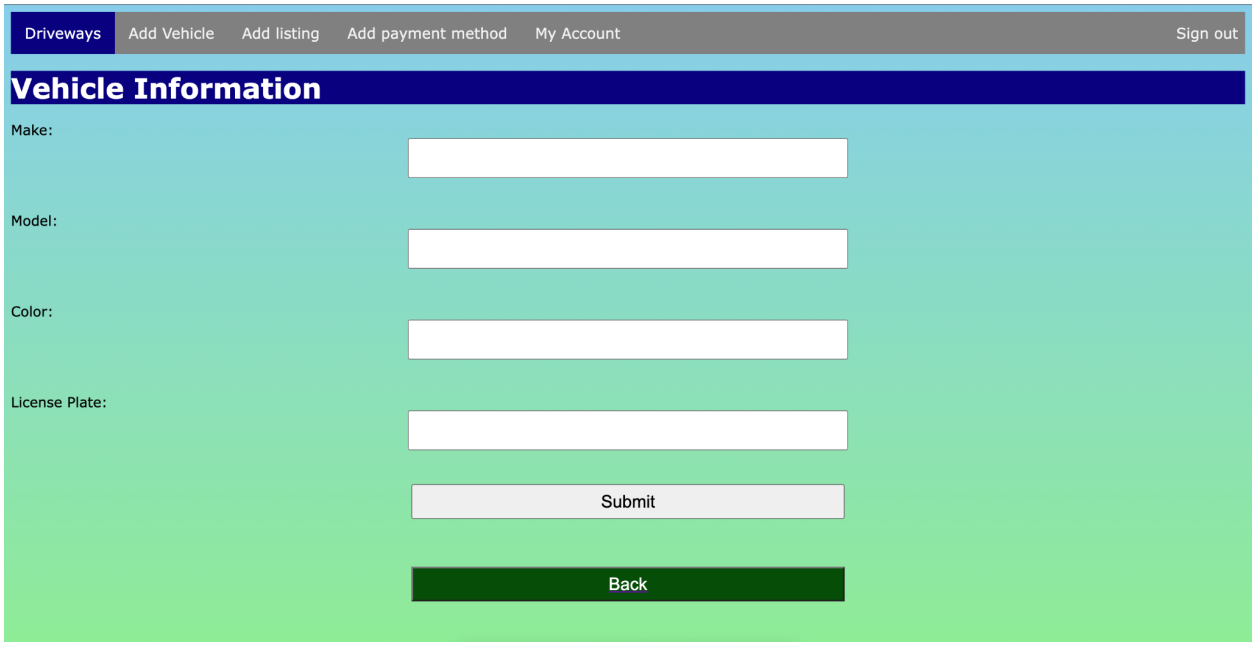
The snapshot shows the listing page:

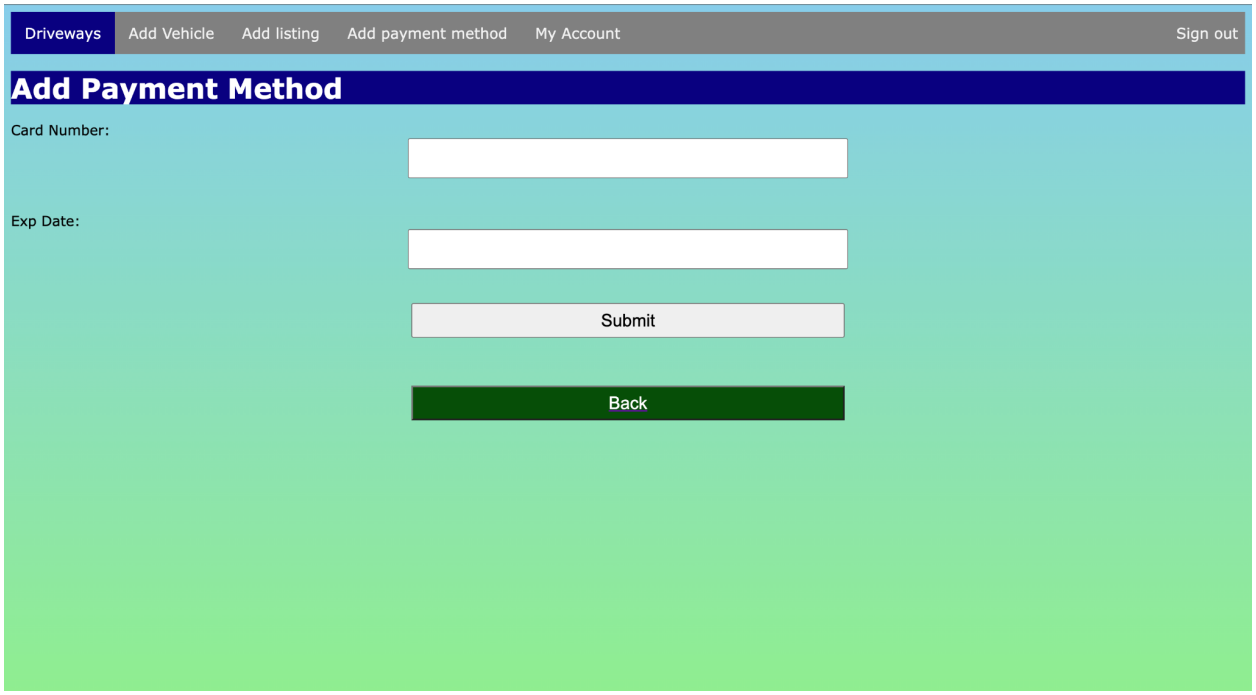
This snapshot shows the parking information when clicking “Reserve”:

3.1.4. Add a listing, vehicle information and rent a listing

The snapshots shows the add listing page, add vehicle page and add payment info page:

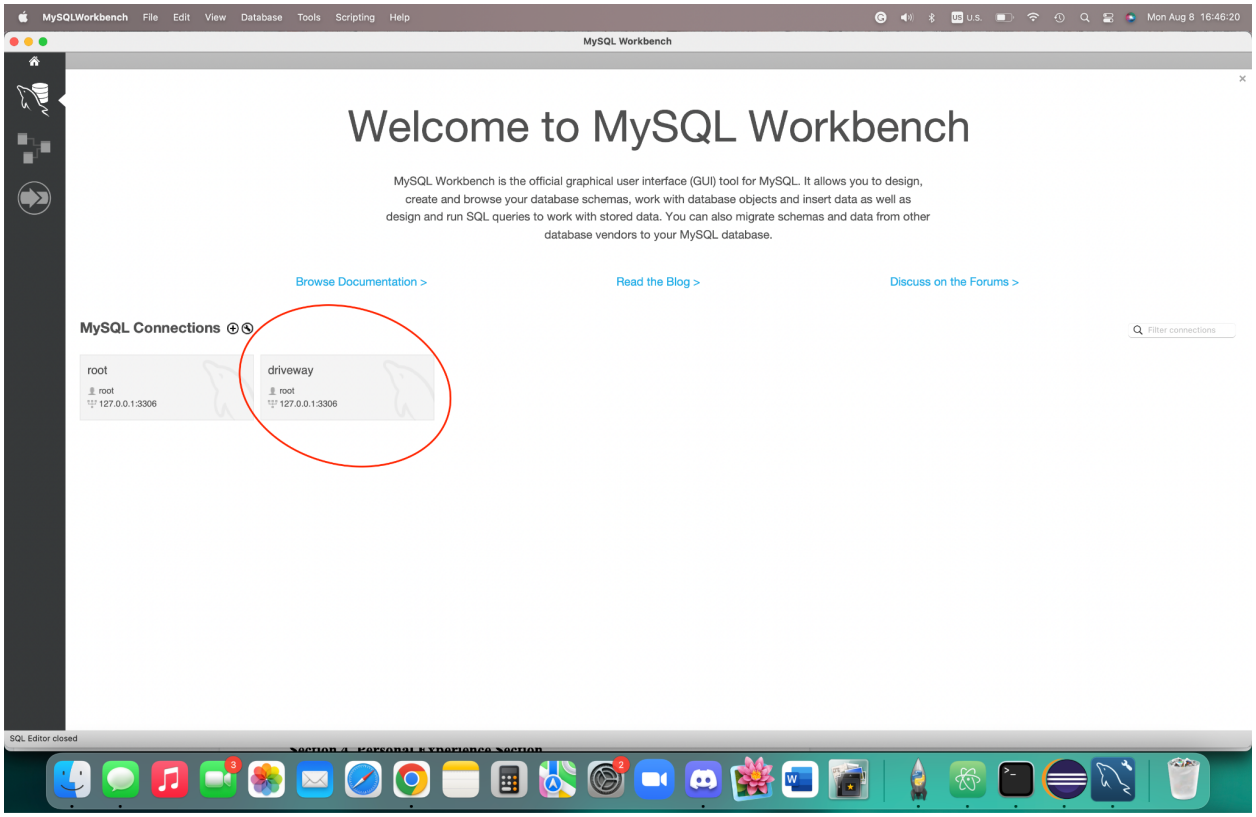



Driveways
Add Vehicle
Add listing
Add payment method
My Account
Sign out
Vehicle Information
Make:
Model:
Color:
License Plate:
Submit
Back


Driveways
Add Vehicle
Add listing
Add payment method
My Account
Sign out
Add Payment Method
Card Number:
Exp Date:
Submit
Back

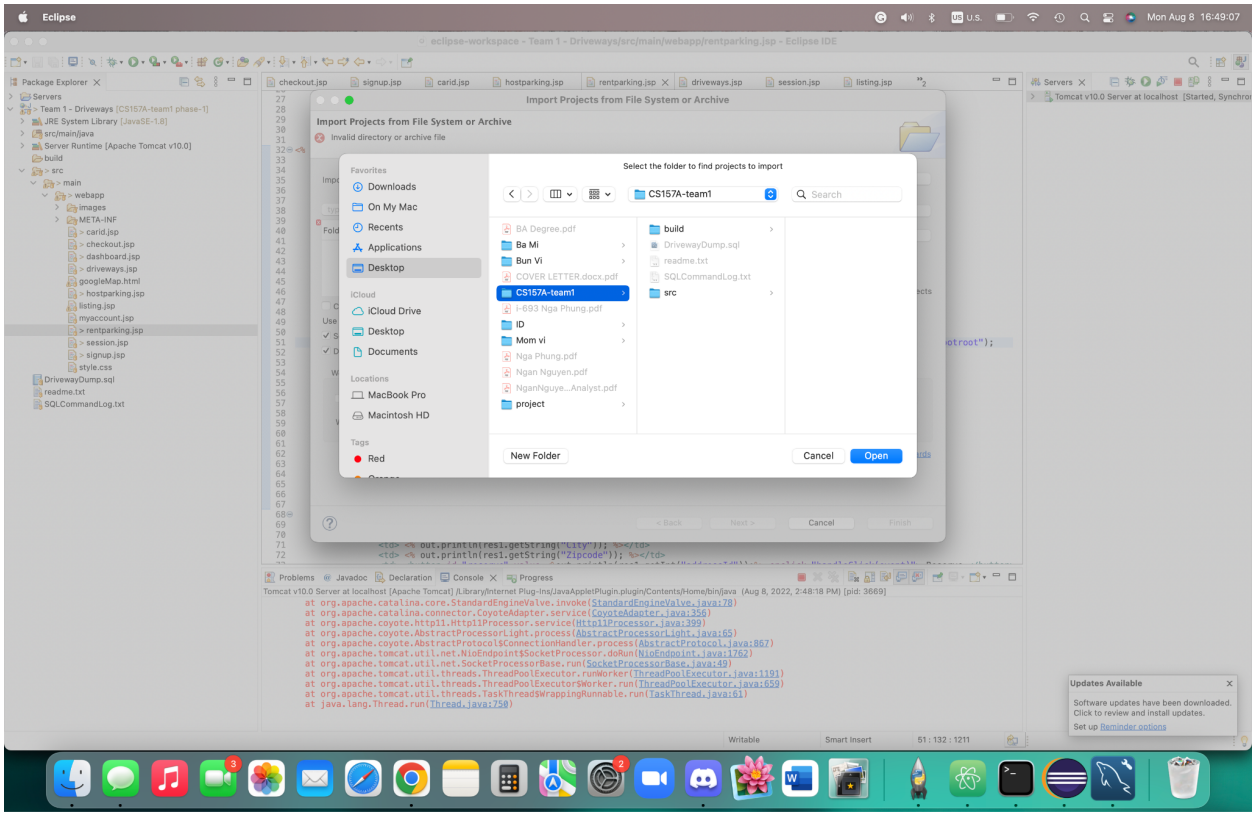
3.2. How to set up and run the system

3.2.1. (In MySQL Workbench) Create a connection named “driveway”



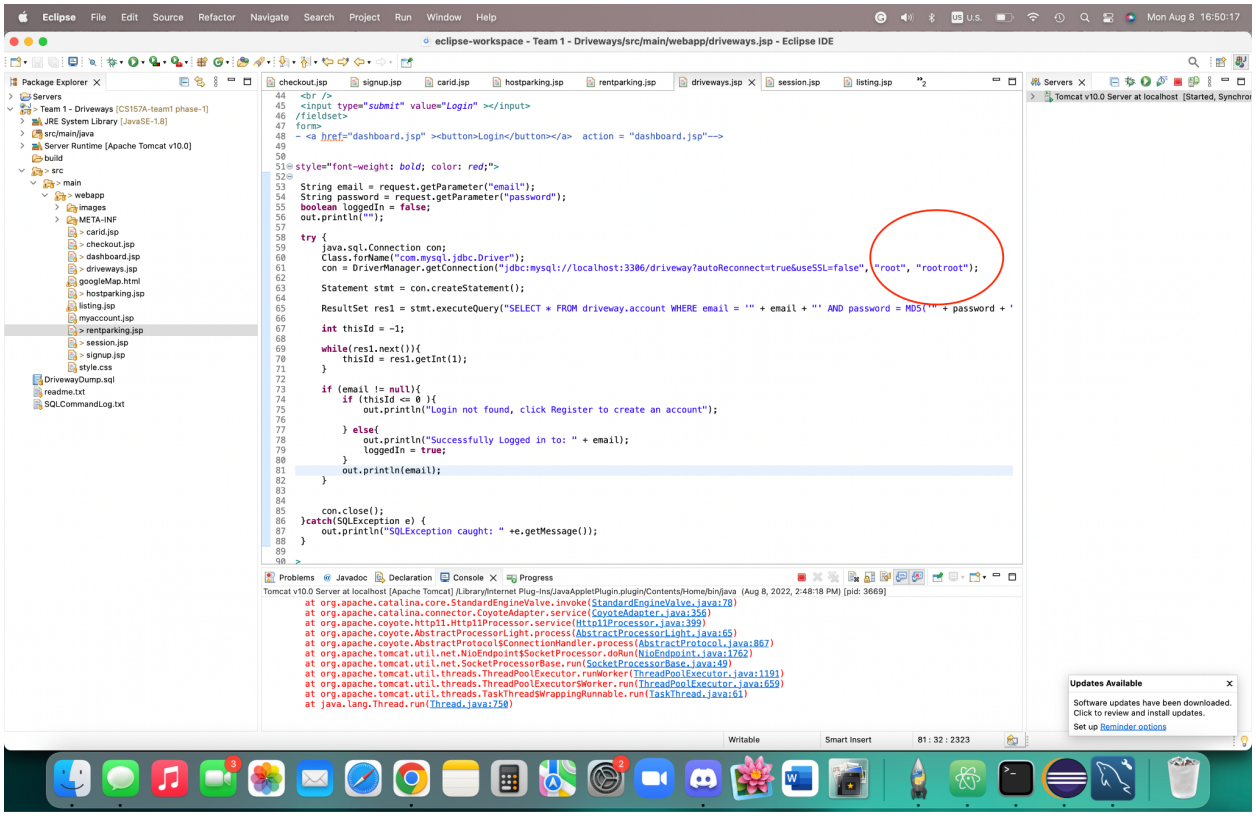
3.2.2. (In MySQL Workbench) Import the DrivewayDump.sql database from the zip file

3.2.3. (In Eclipse) Open project from File System → choose the folder contained in the zip file named “CS157A-team1”

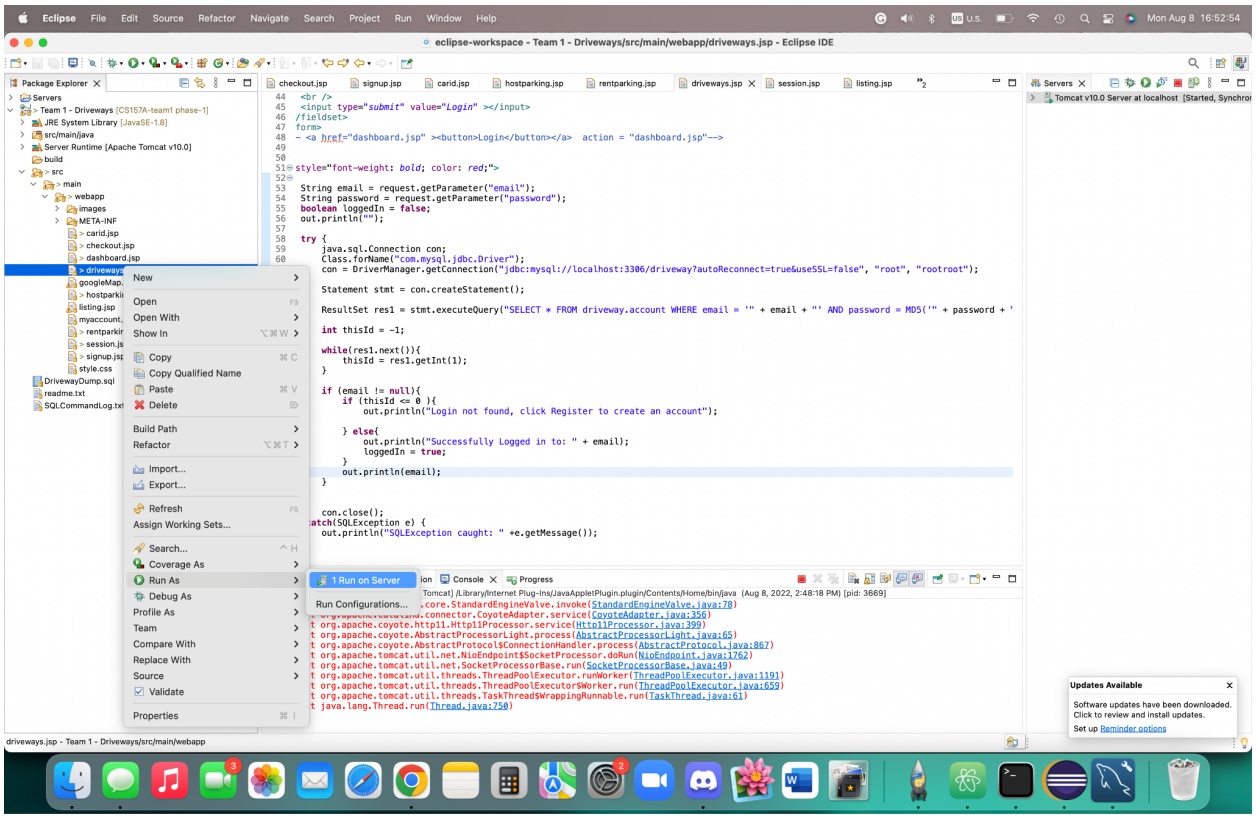


3.2.4. (In Eclipse) Correct the username and password to your own credentials in every file in order for the SQL connection to work ****The default codes have our credentials which are "root" & "Root123!" → Correct it to your own to have the whole thing run properly* OR change your database password to match with our password for a quicker access to the application.***

For example:


eclipse-workspace - Team 1 - Driveways/src/main/webapp/driveways.jsp - Eclipse IDE
Package Explorer
checkout.jsp
signup.jsp
carid.jsp
hostparking.jsp
rentparking.jsp
driveways.jsp
session.jsp
listing.jsp
Servers
Tomcat v10.0 Server at localhost [Started, Synchron
Problems
Javadoc
Declaration
Console
Progress
Updates Available
Software updates have been downloaded.
Click to review and install updates.
Set up Reminder options
Writable
Smart Insert
81 : 32 : 2323

3.2.5. (In Eclipse) Run the driveways.jsp with TomCat server



3.2.6. Follow described steps in Section 3.1. to use the web application

Section 4. Personal Experience

4.1. Dat Le

The project is the first real-world application that I have ever had throughout the entire college career. This is also the first time I learned about databases and how a project from Eclipse can connect to a database. By working on this project, I learned how to design a system structure and an ERD, which are essential tools for any software engineer. Also, after completing the project, I now understand the basics of CSS, HTML, and JavaScript. The Driveway project motivated me to pick up and learn about MySQL and other programming languages in two weeks through several online sources. GitHub was also a new experience for me. I have had an account since 2019 and never had a chance to use it. Through doing the project, I finally had a chance to practice using Git and GitHub so as to create a better and smoother working environment for the team. I never recognized that my limit was this far. The GUI of our project is still simple, but this is no doubt one of the proudest innovations that I have ever made in my life. My team is in perfect chemistry with hardworking and persistent mindsets, we never complained about obstacles and challenges when coming across one while building the web application. There is no regret when I have a chance to be in the team.

4.2. Michael Moore

This was my first time working with any database application, and it was a great learning experience. It was also my first time working with JSP, Tomcat, and MySQL specifically. I found the interaction between JSP and MySQL to be very specific, and getting the

syntax correct was sometimes difficult. I spent much of my time getting familiar with JSP and MySQL interactions on one specific part of the project. However, once I was able to figure out one portion of the project, the code was often applicable in many other sections of the project as well. This project was also a great lesson on how to use (and how not to use) Github. Only once we got to the end of the project was I really aware of how to fully utilize the power of branching, merging, and general version control. My teammates were great and really made the fast-paced nature of this summer semester easier to handle.

4.3. Mohit Sonwane

This was my first interesting and challenging group project in which I used a SQL database, and used up to 4 programming languages including: html, css, javascript, jsp. Thus, this project gave me a fruitful experience in web development using SQL databases, and will add to my skill set for my future career. The part I enjoyed the most was to design the ERD diagram of our database, and to learn on how to interact with the database. However, the challenging part for me was to understand how github works, setting up 3 tier architecture, and writing JSP code to retrieve data from SQL databases. Moreover, I had been using VScode/Atom IDE for web development in the past which had so many features to write code smoothly and quickly. Using Eclipse IDE was not a pleasant experience. However, with the help of my panthers, and browsing on the internet , especially stack overflow, and www3 I was able to overcome the problems I was facing. My teammates were very supportive and were open to other people’s ideas. Hence, we were able to work efficiently based on the system design and functional specs we

proposed prior to implementation. The only thing I felt was that, as everyone was learning some skills and Git for the first time, we used GitHub in a messy way and did redundant work. But, through proper communication and tracking each other's progress, we were able to solve the problems we faced.