Ski Resort Database

(Final Report)

Group Name: Jino
Audrey Pino & Harvey Ji

I. Introduction

The goal of this project is to create a comprehensive database that can be used to track ski resort activity, manage employees, and track skier information and ticketing. Further, this project provides a conceptual representation of the database through the design of an Entity Relationship Diagram (ERD) and Logical Design. The ERD conveys the relationships between the entities, while the Logical Design provides a detailed representation of the data structure and design. The illustrated database provides the ski resort with vital information and gives a better understanding of the day-to-day operations.

In addition to the conceptual design, the project will also include a physical implementation, allowing a user-friendly interface for ski resort customers and employees to easily input and retrieve information from the database. The frontend will be designed using HTML and CSS, with a modern and intuitive interface to facilitate easy navigation and data input. The backend will be built using PHP and MySQL, with PHP serving as the programming language to handle database interactions and MySQL serving as the database management system.

II. Requirements Analysis

The designed ski resort database includes the entities Ski Resort, Employee, Skier, Tickets, and Lifts. The Ski Resort entity will contain information about each resort, such as its name, address, and phone number. The Employee entity will store data on the resort's staff, including their employeeID, names, salary, and phone number. The Skier entity will hold information on each individual skier, such as their SSN, name, DOB, and phone number. The Tickets entity will contain data on the ticketID, skier name, and expiration date. Further, the Ticket entity is classified as a weak entity set that is dependent on the Ski Resort entity as tickets across different resorts may have the same information. The Lifts entity will store information about each lift at the resort, including the lift name, and total capacity. The Lifts entity is also a weak entity set since lifts across different resorts may have the same name. Further, the Lift entity is broken into two subclasses of Chairlift and Gondola that store data on the lift name, and seat capacity or gondola capacity.

Constraints on the database may include limits on the number of lifts, employees, and skiers that can be accommodated at the resort, as well as restrictions on the number of tickets that

can be sold for each lift. Operations that will need to be supported by the database include ticket sales and tracking, lift usage and capacity, employee payroll, and skier demographic analysis. Additionally, the database may need to support reporting and analytics functions to provide insights into resort operations and trends.

Entities:

- Ski Resort
- Employee
- Skier
- Tickets
- Lifts
    - Chair Lift → Subclass
    - Gondola → Subclass

Relationships:

- Many Employees work for a Ski Resort: Many to One
- Many Skiers may ski at many different Ski Resorts: Many to Many
- The Ski Resort sells many Tickets: Many to Many
- The Ski Resort operates many Lifts: Many to One

Attributes:

- Ski Resort
    - Resort_Name -> Primary Key
    - Phone_Number
    - Address
- Skier
    - SSN -> Primary Key
    - Name
    - DOB
    - Email
- Employee
    - EmployeeID -> Primary Key
    - Employee_Name
    - Email
    - Salary
    - Resort_Name -> Foreign Key
- Tickets
    - TicketID -> Primary Key
    - Skier_Name
    - Expiration_Date
    - Resort_Name -> Foreign Key
- Lifts
    - Lift_Name -> Primary Key

- Seat_Capacity
- Lift_Type
- Resort_Name -> Foreign Key

III. Entity Relationship Diagram (ERD)

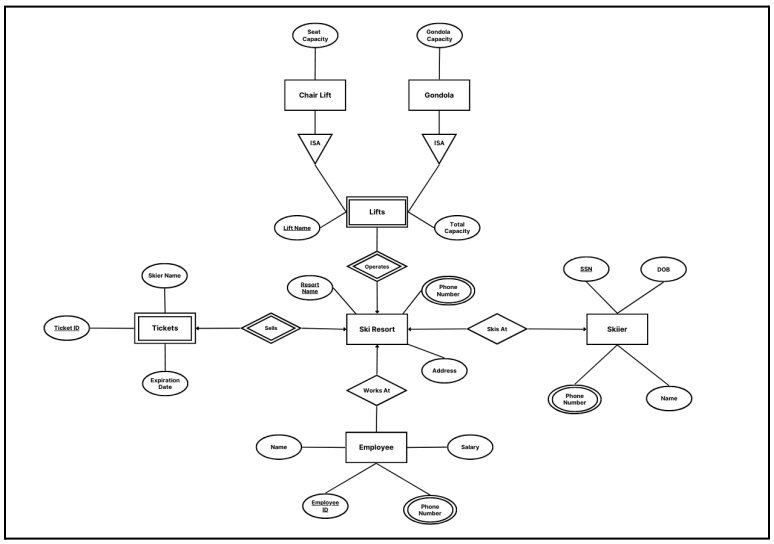

Figure 1: *Original Entity Relationship Diagram (ERD)*

IV. Logical Database Design

IV.1. Relational Schemas

SkiResort(Resort_Name, Phone_Number, Address)
Skier(SSN, Name, DOB, Phone_Number, Resort_Name)
Employee(EmployeeID, Employee_Name, Phone_Number, Salary, Resort_Name)
Tickets(TicketID, Skier_Name, Expiration_Date, Resort_Name)
Lift(Lift_Name, Total_Capacity, Resort_Name)
CharLift(Lift_Name, Seat_Capacity)
Gondola(Lift_Name, Gondola_Capacity)

IV.2. Normalized Relations

1. **SkiResort Relation:**
   SkiResort(Resort_Name, Phone_Number, Address)
   Resort_Name-> Phone_Number, Address

   SkiResort(A, B, C)
   A -> BC

The SkiResort relation has no violation so therefore is in BCNF.
The SkiResort relation has no trivial FD's so therefore is in 3rd Normal Form.

2. **Skier Relation:**
   Skier(SSN, Name, DOB, Phone_Number)
   SSN -> Name, DOB, Phone_Number

   Skier(A, B, C, D)
   A -> BCD
   A (SSN) -> Primary Key
   The Skier relation has no violation so therefore is in BCNF.
   The Skier relation has no trivial FD's so therefore is in 3rd Normal Form.

3. **Employee Relation:**
   Employee(EmployeeID, Employee_Name, Phone_Number, Salary, Resort_Name)
   EmployeeID -> Employee_Name, Phone_Number, Salary, Resort_Name

   Employee(A, B, C, D, E)
   A -> BCDE
   A (EmployeeID) -> Primary Key
   E (Resort_Name) -> Foreign Key
   The Employee relation has no violation so therefore is in BCNF.
   The Employee relation has no trivial FD's so therefore is in 3rd Normal Form.

4. **Tickets Relation:**
   Tickets(TicketID, Skier_Name, Expiration_Date, Resort_Name)
   TicketID -> Skier_Name, Expiration_Date, Resort_Name
   Tickets(A, B, C, D)

   A -> BCD
   A (TicketID) -> Primary Key
   D (Resort_Name) -> Foreign Key
   The Tickets relation has no violation so therefore is in BCNF.
   The Tickets relation has no trivial FD's so therefore is in 3rd Normal Form.

5. **Lift Relation:**
   Lift(Lift_Name, Total_Capacity, Resort_Name)
   Lift_Name -> Total_Capacity, Resort_Name

   Lift(A, B, C)
   A -> BC
   A (Lift_Name) -> Primary Key
   C (Resort_Name) -> Foreign Key
   The Lift relation has no violation so therefore is in BCNF.

The Lift relation has no trivial FD's so therefore is in 3rd Normal Form.

6. **CharLift Relation:**
   CharLift(Lift_Name, Seat_Capacity)
   Lift_Name -> Seat_Capacity

   CharLift(A,B)
   A -> B
   A (Lift_Name) -> Primary Key
   The ChairLift relation has no violation so therefore is in BCNF.
   The ChairLift relation has no trivial FD's so therefore is in 3rd Normal Form.

7. **Gondola Relation:**
   Gondola(Lift_Name, Gondola_Capacity)
   Lift_Name -> Gondola_Capacity

   Gondola(A,B)
   A -> B
   A (Lift_Name) -> Primary Key
   The Gondola relation has no violation so therefore is in BCNF.
   The Gondola relation has no trivial FD's so therefore is in 3rd Normal Form.

The Employee, Ticket and Lifts tables have a foreign key column for Resort_Name, which links them to the SkiResort table. This allows us to identify which employees, skiers, tickets and lifts belong to the resort. This ER diagram is in BCNF, as every non-trivial functional dependency in the schema has a candidate key as its determinant. Specifically, the candidate keys for each entity are:

SkiResort: Resort_Name
Skier: SSN
Employee: EmployeeID
Tickets: TicketID
Lifts: Lift_Name

Further, every attribute in the schema is dependent only on the candidate keys and not on any other non-key attribute, which satisfies the requirements for BCNF.

V. Tables

1. Ski Resort

RESORTNAME	PHONENUM	ADDRESS
Big Sky Resort	8005484486	50 Big Sky Resort Rd, Big Sky, MT 59716
Breckenridge Ski Resort	9704535000	1599 Ski Hill Rd, Breckenridge, CO 80424
Crested Butte Mountain Resort	9702517021	12 Snowmass Rd, Crested Butte, CO 81225
Jackson Hole Mountain Resort	3077332292	3395 Cody Ln, Teton Village, WY 83025
Park City Mountain Resort	4356498111	1345 Lowell Ave. Park City, UT 84060
Winter Park Resort	9707265514	85 Parsenn Rd, Winter Park, CO 80482

Figure X: *RESORT Table in MySQL*

2. Skier

SKIERID	NAME	DOB	EMAIL	PASSWORD
162930576	Ansley Powell	1984-06-16	apowell5432@outlook.com	greeneyes2
207518473	Kassandra Norman	2005-09-23	knorman5987@gmail.com	sunny123
293756861	Kaylynn Peterson	1977-03-18	kaylynn.petersen@email.com	iloveyou22
320661190	Jeffery Padilla	1999-11-02	jpadilla6879@yahoo.com	coffee4me
399347918	Keira Kelley	1972-06-05	keirak2376@gmail.com	starrynight6
462082133	Dayton Rubio	1983-05-06	drubio5421@outlook.com	sunshine5
524799808	Madeline Lin	1996-07-12	madelinel4312@yahoo.com	1234abcd
537166321	Nayeli Santiago	1965-10-31	nayelis4523@yahoo.com	kitten123
559504774	Jino Jino	2000-05-02	jino@gmail.com	12345
675739159	Kristina Lutz	1975-12-23	kristinal2987@gmail.com	musiclover3
678237710	Semaj Farrell	1963-09-12	sfarrell2451@yahoo.com	carpediem7
701245392	Lukas Mendoza	1971-02-08	lmendoza8674@hotmail.com	doglover1
710408578	Ezequiel Ruiz	1989-02-15	eruiz5634@hotmail.com	bluebird8
730831056	Rowan Barber	1990-09-07	rowanb2309@email.com	mountain7
812698174	Leilani James	1991-04-19	leilanij6534@outlook.com	happydog2
816684312	Alvaro Callahan	1993-08-04	acallahan7643@hotmail.com	summerfun4
837506380	Harvey Ji	2023-01-11	harvey.ji123@gmail.com	12345
879693238	Hannah Mullins	2000-01-15	hannahm3456@hotmail.com	rainbow99
971548607	Martin York	1987-01-27	martiny9876@gmail.com	football99
984927589	Raven Villa	2002-12-28	ravenv8912@email.com	pizzalover1
985461021	Kylan Valdez	2007-03-29	kylanv4572@yahoo.com	butterfly9
986152987	Ibrahim Wilkins	1978-11-08	ibrahimw3245@outlook.com	beachgirl7

Figure X: *SKIER Table in MySQL*

3. Employee

EMPLOYEEID	EMPLOYEENAME	SALARY	RESORTNAME	EMAIL	PASSWORD
2285	Brennen Curtis	7563.46	Jackson Hole Mountain Resort	brennen.curtis1234@gmail.com	brennen789
2286	Cael Calhoun	10814.29	Crested Butte Mountain Resort	cael.calhoun@email.com	cael1234
2736	Fatima Livingston	17595.05	Big Sky Resort	fatima.livingston@hotmail.com	fatima3456
3195	Brylee Duran	9980.12	Winter Park Resort	bryleed1267@outlook.com	brylee1234
3826	Enzo Keller	11732.67	Big Sky Resort	enzok5421@gmail.com	enzo9012
3890	Alani Kirby	11586.42	Winter Park Resort	alani.kirby@email.com	alani456
3982	Frankie Dixon	5781.22	Winter Park Resort	frankied1234@outlook.com	frankie789
3987	Jonah Johnson	3250.50	Big Sky Resort	jonahj9687@yahoo.com	jonah5678
4298	Mina Campos	8942.35	Crested Butte Mountain Resort	minac3049@yahoo.com	mina123
4798	Jadon Schmidt	15689.24	Breckenridge Ski Resort	jadon.schmidt@email.com	jadon5678
5141	Dayton Patterson	19344.11	Park City Mountain Resort	dayton.patterson@email.com	dayton5678
5182	Aleah Lyons	29766.62	Winter Park Resort	aleah.lyons3456@hotmail.com	aleah5678
6012	Nicolas Higgins	6799.17	Breckenridge Ski Resort	nhiggins7642@outlook.com	nicolas789
6045	Maxim Garner	18670.76	Jackson Hole Mountain Resort	maxim.garner@email.com	maxim9012
6330	Keenan Gamble	8865.99	Jackson Hole Mountain Resort	keenang2938@hotmail.com	keenan3456
8179	Lillian Thompson	2780.79	Park City Mountain Resort	lillian.t5678@yahoo.com	lillian9012
8484	Janae Woods	13578.93	Breckenridge Ski Resort	janaew8754@yahoo.com	janae1234
8991	Ethen Fleming	9432.88	Crested Butte Mountain Resort	ethenf2345@hotmail.com	ethen3456
9214	Darwin Marquez	22569.83	Crested Butte Mountain Resort	darwinm4389@yahoo.com	darwin789
9954	Shawn Swanson	22379.99	Park City Mountain Resort	shawnswanson1234@gmail.com	shawn1234

Figure X: *EMPLOYEE Table in MySQL*

4. Tickets

TICKETID	SKIERID	PURCHASEDDATE	EXPIRATIONDATE	RESORTNAME
3501	162930576	2023-04-06 14:08:38	2023-08-16	Big Sky Resort
6969	162930576	2023-04-14 14:22:00	2023-08-20	Winter Park Resort
2075	207518473	2023-02-19 14:26:55	2023-08-10	Crested Butte Mountain Resort
8365	293756861	2023-04-10 18:27:05	2023-11-13	Big Sky Resort
7913	320661190	2023-03-19 11:56:55	2023-11-02	Breckenridge Ski Resort
4268	399347918	2023-03-24 20:06:11	2023-10-31	Copper Mountain Resort
9031	462082133	2023-02-23 22:02:47	2023-09-02	Crested Butte Mountain Resort
6021	524799808	2023-04-08 17:39:21	2023-07-25	Breckenridge Ski Resort
9016	537166321	2023-04-13 11:49:17	2023-12-01	Winter Park Resort
5289	559504774	2023-05-02 13:43:28	2024-04-02	Crested Butte Mountain Resort
5874	675739159	2023-02-15 19:43:01	2023-12-10	Park City Mountain Resort
2609	678237710	2023-03-29 16:50:01	2023-09-25	Jackson Hole Mountain Resort
6184	701245392	2023-03-11 08:51:24	2023-08-04	Jackson Hole Mountain Resort
9287	710408578	2023-02-12 09:18:47	2023-08-28	Park City Mountain Resort
9172	730831056	2023-04-14 12:27:09	2023-07-15	Winter Park Resort
3750	812698174	2023-02-27 08:22:08	2023-07-31	Copper Mountain Resort
4928	816684312	2023-02-16 12:14:59	2023-12-23	Park City Mountain Resort
3612	837506380	2023-05-01 21:02:41	2023-05-26	Park City Mountain Resort
1456	879693238	2023-04-03 09:33:12	2023-12-18	Winter Park Resort
6543	971548607	2023-03-04 15:37:42	2023-10-29	Breckenridge Ski Resort
1728	984927589	2023-04-01 13:55:29	2023-07-17	Big Sky Resort
2634	985461021	2023-02-20 17:15:53	2023-09-30	Crested Butte Mountain Resort
9861	986152987	2023-03-07 10:45:33	2023-11-20	Winter Park Resort

Figure X: *TICKET Table in MySQL*

5. Lifts

LIFTNAME	RESORTNAME	LIFTTYPE	SEATCAPACITY
Aerial Tram	Jackson Hole Mountain Resort	Gondola	100
Arrow Lift	Winter Park Resort	Chair Lift	4
BreckConnect Gondola	Breckenridge Ski Resort	Gondola	8
Bridger Gondola	Jackson Hole Mountain Resort	Gondola	8
Challenger Lift	Winter Park Resort	Chair Lift	4
Colorado SuperChair	Breckenridge Ski Resort	Chair Lift	4
Crescent	Park City Mountain Resort	Chair Lift	4
Eskimo Express	Winter Park Resort	Chair Lift	6
Imperial Express SuperChair	Breckenridge Ski Resort	Chair Lift	4
Lone Peak Tram	Big Sky Resort	Gondola	15
Painter Boy	Crested Butte Mountain Resort	Chair Lift	3
Payday	Park City Mountain Resort	Chair Lift	6
Quicksilver Gondola	Park City Mountain Resort	Gondola	8
Quicksilver SuperChair	Breckenridge Ski Resort	Chair Lift	6
Ramcharger 8	Big Sky Resort	Chair Lift	8
Red Lady Express	Crested Butte Mountain Resort	Chair Lift	4
Silver Queen Express	Crested Butte Mountain Resort	Chair Lift	4
Swift Current	Big Sky Resort	Chair Lift	4
Teewinot Chairlift	Jackson Hole Mountain Resort	Chair Lift	4
Zephyr Express	Winter Park Resort	Chair Lift	6

Figure X: *LIFT Table in MySQL*

VI. Updated ER Diagram

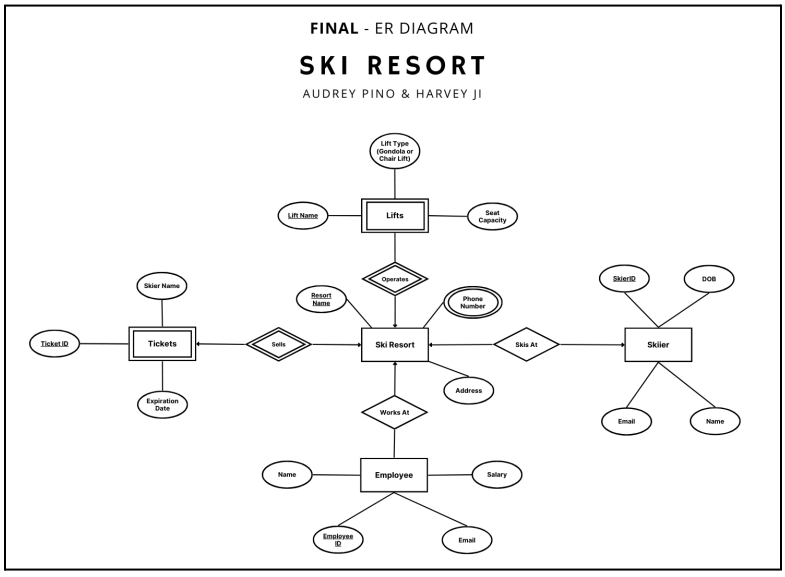

Figure X: Updated Entity Relationship Diagram (ERD)

VII. Web Pages

The Ski Resort database will be accessed through a web page interface in order to offer easy access to information for both the customers and employees.

On the customer side, the webpages are designed to allow customers to *Sign Up* and *Login* to access the system. A query will ensure that a customer does not already exist before signup. A query will create a unique skierID upon account creation. After logging in, the homepage will allow customers to search for specific resorts, lifts, and lift capacity. Customers will also have a *Profile* page to display their information, including their name, skierID, email, and DOB. The customer *Buy Tickets* page will allow customers to purchase tickets for the desired resort, age, and expiration date. Customers can also access their purchased tickets through the *MyTickets* page, which will use a join query to display ticket information for the specific customer. The *MyLifts* page will use a join query to display the lifts available to customers based on their purchased tickets. The *Resort* page will display information about each resort, including the resort address, phone number, and available lifts.

The employee side of the web page will allow employees to access the same *Home* page and *Resort* page as on the customer side. The employees are able to access additional features through the employee *Login* console. Employees will have a *Profile* page displaying their name, employee ID, email, which resort they work at, and their salary. The employee *Buy Ticket* page will allow employees to purchase tickets for customers.

Overall, this ski resort database web page implementation will provide a user-friendly interface for customers to access resort information, purchase tickets, and view their information. Employees can also access the system to make purchases on behalf of customers and manage resort information.

1. Signup

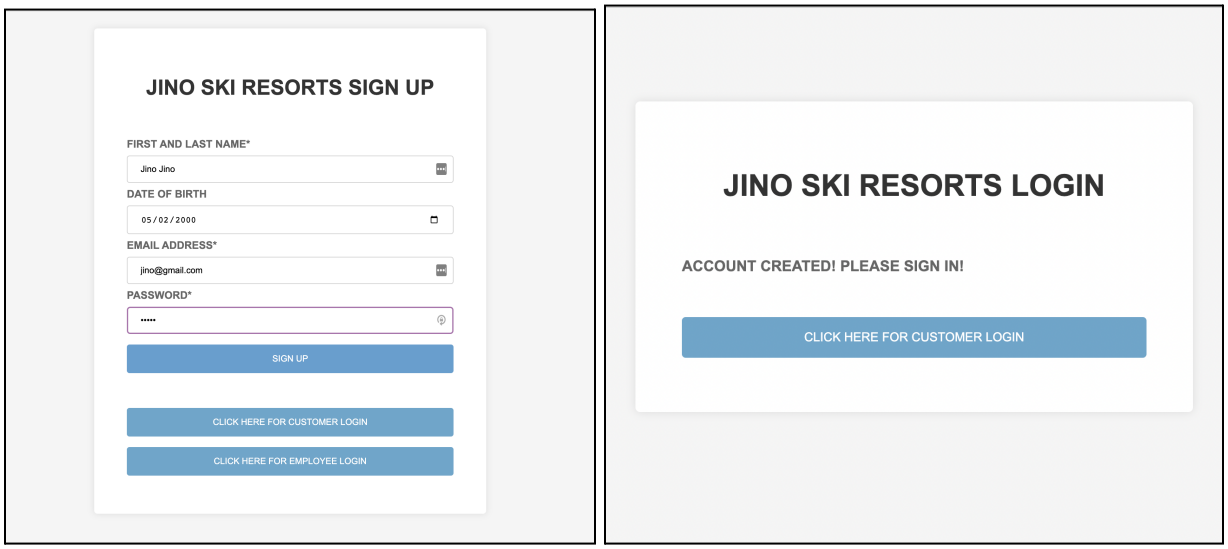

Figure X & X: *Example User Sign-Up & Sign-Up Confirmation/Redirect Screen*

2. Customer Login Page

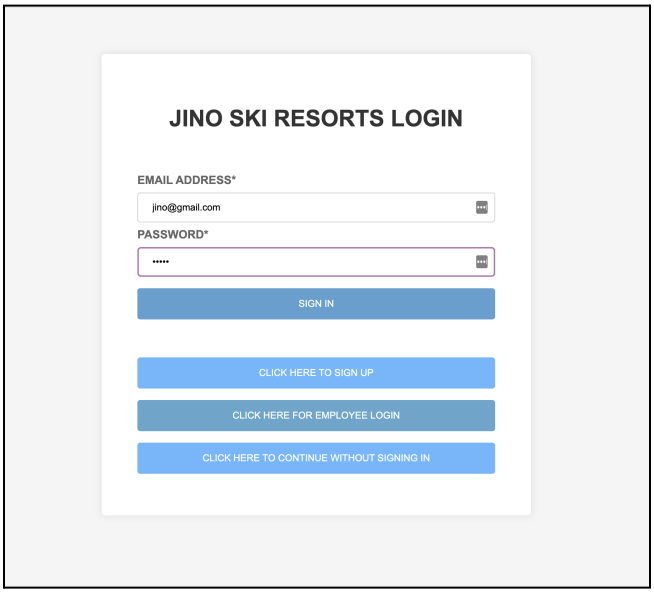

Figure X: *Example Customer Login*

3. Home Page

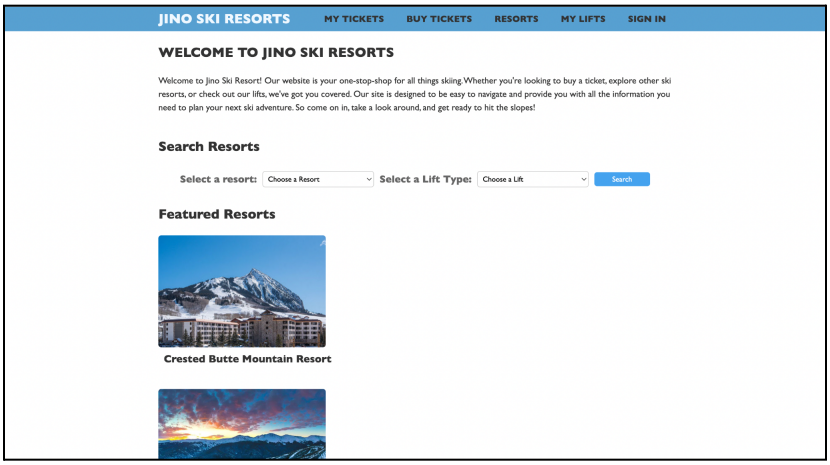

Figure X: *Home Page before User Login*

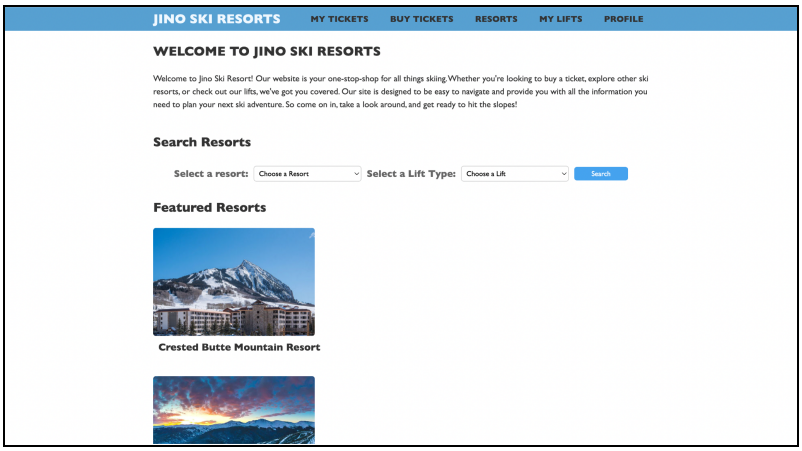

Figure X: *Home Page after User Login*

4. Resort Page

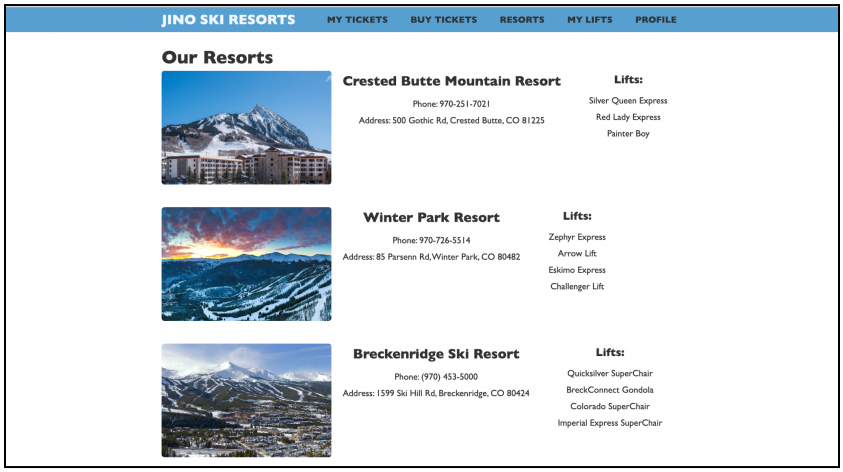

Figure X: *Resort Page*

5. Profile

Figure X: *Example User Profile Page*

6. Buy Tickets

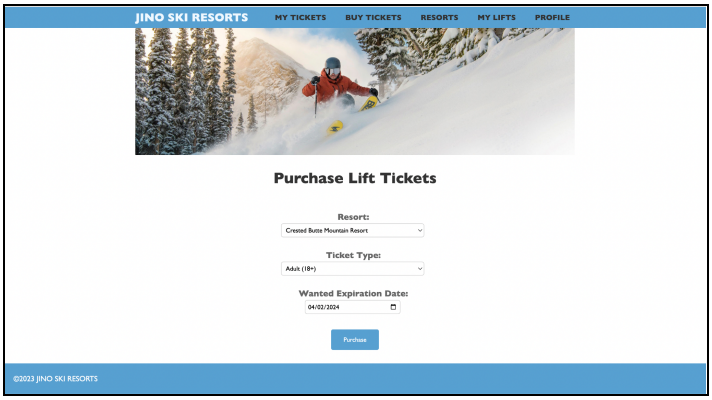

Figure X: *Example Buy Tickets*

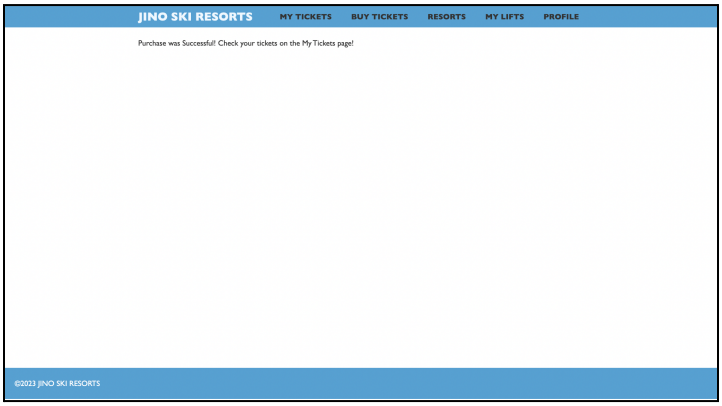

Figure X: *Successful Ticket Purchase*

7. My Tickets

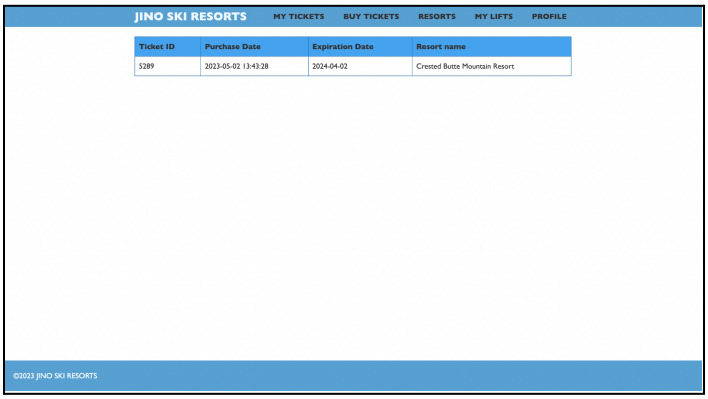

Figure X: *Example of MyTickets Page after Successful Ticket Purchase*

8. My Lifts

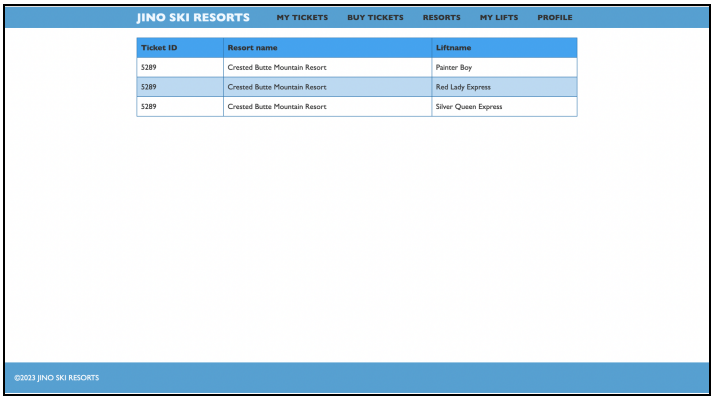

Figure X: *Example of MyLifts Page after Successful Ticket Purchase*

9. Employee Login Page

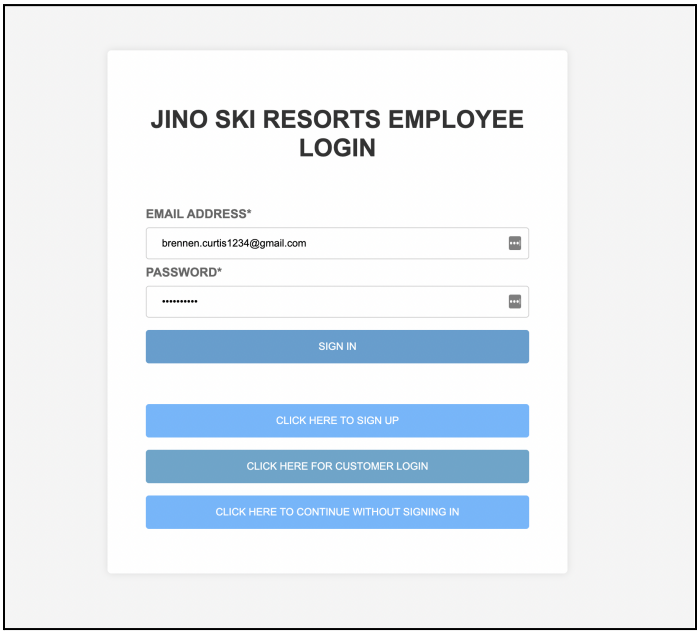

Figure X: *Example Employee Login*

10. Employee Profile

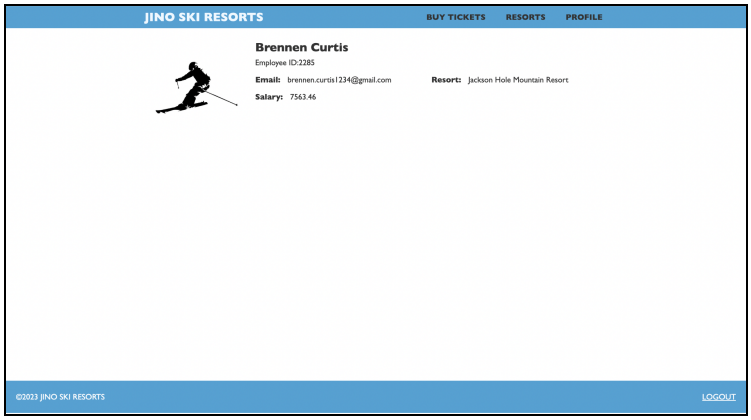

Figure X: *Example Employee Profile*

11. Employee Buy Tickets

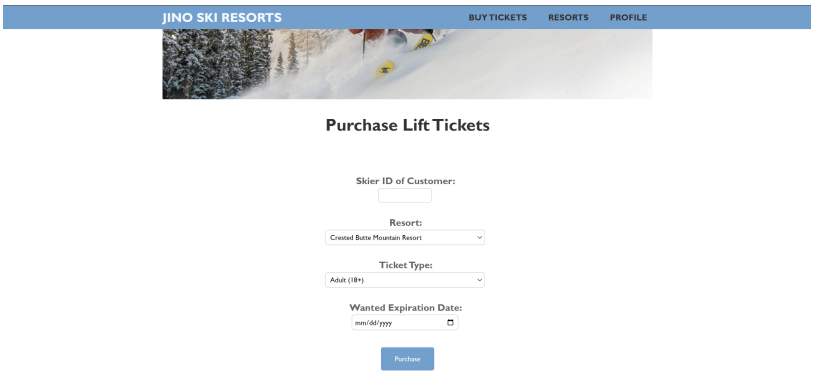

Figure X: *Employee Purchasing Ticket for Customer*

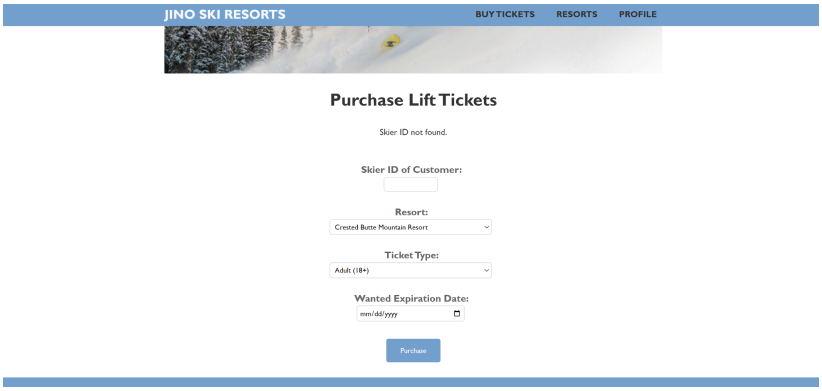

Figure X*: Skier ID isn't valid*

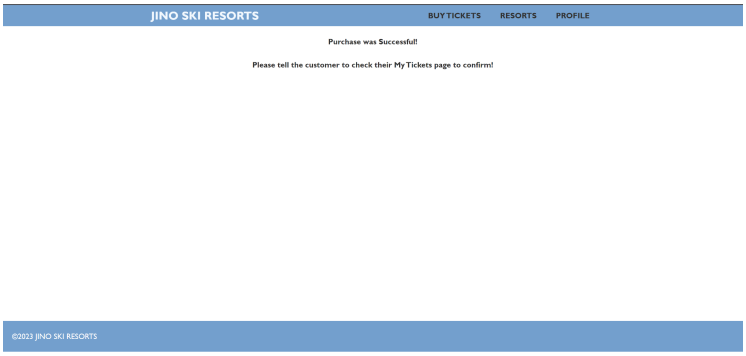

***Figure X**: Purchase Success*

VIII. Database Queries

Listed below are all the dynamic queries used:
Highlighted bullet points indicate Join Query was used.

- Query to Login for Customers:
  SELECT * FROM SKIER WHERE EMAIL='$email' AND PASSWORD='$password'

- Queries used to buy tickets:
  - To check that the randomly generated ticket ID wasn't already in use at that resort:
    SELECT * FROM TICKETS WHERE TICKETID = '$newTicketID' AND RESORTNAME = '$resort'
  - To insert the new ticket into the TICKETS table:
    INSERT INTO TICKETS VALUES('$newTicketID', '$skierID', '$currentDate', '$wantedExpire', '$resort')
  - To check that the insert was successful:
    SELECT * FROM TICKETS WHERE TICKETID = '$newTicketID' and RESORTNAME = '$resort'

- Query used on Customer Side to find their tickets:
  SELECT TICKETID, TICKETS.SKIERID, NAME, PURCHASEDDATE, EXPIRATIONDATE, RESORTNAME FROM TICKETS, SKIER WHERE TICKETS.SKIERID='$skierID' AND SKIER.SKIERID=TICKETS.SKIERID

- Query used to find lifts available to customer based on ticket:
  SELECT TICKETID, LIFTS.RESORTNAME, LIFTNAME FROM LIFTS, TICKETS WHERE TICKETS.SKIERID='$skierID' AND TICKETS.RESORTNAME=LIFTS.RESORTNAME

- Query to Login for Employees:

SELECT * FROM EMPLOYEE WHERE EMAIL='$email' AND PASSWORD='$password'

- Queries to Sign Up:
    - To check if the email was already in use:
      SELECT * FROM SKIER WHERE EMAIL='$email'
    - To check if the randomly generated Skier ID was already in use:
      SELECT SKIERID FROM SKIER WHERE SKIERID = '$newSkierID'
    - To insert the new Skier info into the database:
      INSERT INTO SKIER VALUES('$newSkierID', '$fullName', '$dob', '$email', '$password'
    - To check that the insert was successful:
      SELECT * FROM SKIER WHERE SKIERID = '$newSkierID'