Project Report:

Project Plan & Database Design:

- In terms of our initial design process, we can up with 4 unique entities and attributes for the ***Hotel Reservation System***. They include:
  Entity: Guest
  Attributes: GuestID (PK), ReservationID (FK), Name, Email, PhoneNumber

  Entity: Reservation
  Attributes: ReservationID (PK), TransactionNumber, GuestID (FK), RoomNumber, Check_In, Check_Out

  Entity: Employee
  Attributes: EmployeeID (PK), Position, SSN, Name, Address, BirthDate

  Entity: Transactions
  Attributes: TransactionNumber (PK), GuestID (FK), PaymentMethod, Amount, Date

- We then created 2 relationship tables. The first relationship table is called Bookings and it involves the Guest & Reservation table. The second relationship table is called Charges and it involves Transactions & Guest. For the database we created two triggers that will automatically populate these tables after data is inserted into the Reservation table and Transaction table

- We designated the Employee table as the superclass with 2 subclasses; Managers & Staff. We created the trigger in our database to execute after we insert date into the employee table where depending on the value inside the position attribute it will automatically insert data in the staff or manager table

- In our database, we used the following tables to create a join statement for data retrieval:
- Guest & Transaction
- Reservation & Guest
- Employee & Staff
- For example, for the Guest and Transaction tables, that join statement is going to receive all of the transactions of a certain guest. It retrieves information like their name, payment method, phone number, etc.

Technical Details:

- In terms of the web development we used HTML, CSS, and PHP

- Our web pages are structured with a navigation bar on the top of the screen that allows us to transition between our different web pages. In total we have 5 web pages, A home page, an employee page, a guest page, a reservation page and a transactions page. Our web pages serve to allow a user to input data into the databases, the name of each page corresponds to the table the data will be submitted. In a hypothetical scenario our web pages were built to allow employees of a hotel to record data about the running of the hotel.

- We started off by creating the HTML structure, which included the **<head> and <body>** sections

- Inside the <head>, we linked our external CSS file to handle all the styling:
  <link rel="stylesheet" href="styles.css">

```
<link rel="stylesheet" href="style.css">
```

- For the main layout, we created a title and tagline section to introduce the hotel's name and slogan:
  <h1>The Serenity Hotel</h1>
  ** Indulge in Luxury.
Stay in Serenity. **

- We then created a navigation bar to help users move between pages of the website. Each link is a placeholder that connects to the corresponding PHP pages:

```
<nav>
<div class="topnav">
<a class="active" href="/Groupdatabase_project/Html/Ari/index.html">Home</a>
<a href="Guest.php">Guest</a>
<a href="Reservation.php">Reservation</a>
<a href="Employee.php">Employee</a>
<a href="Transaction.php">Transaction</a>
```

- By utilizing the **<a>** tags we can point to where all of our php files are located thus enabling the user to click on a link and be redirected to whatever page they need.

- Following the nav bar we have the main body of code that contains the forum that users will fill out. This form is housed in a div with a css class called wrapper that specifies the

size of the container housing the form as well as the background color of the container.

```
.wrapper{
    width: 420px;
    background: transparent;
    border: 2px solid rgba(255, 255, 255, .2);
    backdrop-filter: blur(10px);
    color: #fff;

    border-radius: 12px;
    padding: 30px 40px;
}
```

- Inside of the form we have created first a label to identify what data is being requested from the user. Followed by the input box, we are using the post method for submitting our data and each web page gets its own php processing form that handles retrieving and Inserting the data into the proper tables as needed. We also included a place holder to show the user the format desired of the data

```
<form method="post" action="PHP/Employee_processor.php">
    <h1>Employee</h1>
    <div class="inputbox">
        <label for="name"> Name</label>
         <input type="text" name = "name" placeholder="John Doe">
    </div>
```

- This is the standard layout for our webpages consisting of two parts: a nav bar and a div that holds the form. We center the form by adjusting the properties of the body tag as shown below:

```
body{
    display: flex;
    justify-content: center;
    align-items: center ;
    min-height: 100vh;
    background: url(Img/Outside.jpg);
    background-size: cover;
    background-position: center;
}
```

- After the navigation bar, we added a description section to introduce the hotel's theme and environment using:
  ** **

- We also included an image gallery to display visual of the hotel using:
  ** **
  <img src="building.webp" alt="Hotel Exterior">
  ** **

- We have many css classes in our style sheet that handles the font size of our text or size of our input boxes.

```css
.inputbox input{
    width: 100%;
    height: 100%;
    background: transparent;
    border: none;
    outline: none;
    border: 2px solid rgba(255, 255, 255, .2);
    border-radius: 40px;
    font-size: 16px;
    color: #fff;
    padding: 20px 45px 20px 20px;
}
```

- In the CSS file, we started with general page setup to ensure the background, font, and text were consistent throughout the homepage. For example:
  body {
  margin: 0;
  font-family: 'Playfair Display', serif;
  background-color: #6495ED;
  color: #222;
  text-align: center;
  }
- For the navigation bar, we used hover effects and borders to create clickable button-like links:
  nav a {
  text-decoration: none;
  color: #333;
  border: 1px solid #333;
  padding: 8px 15px;
  border-radius: 20px;
  margin: 0 5px;
  transition: 0.3s;

```
}

nav a:hover {
  background-color: #333;
  color: white;
}
```

- This hover effect improves user experience and adds interactivity to the page
- The description section was centered and given a max width to make the text easier to read.
- For the building image, we gave it rounded corners, shadows, and consistent sizing to make it look more clean using:

```
.gallery img {
  width: 600px;
  height: 300px;
  border-radius: 10px;
  object-fit: cover;
  box-shadow: 0 2px 8px rgba(0,0,0,0.2);
}
```

We describe the front end of our website in the pages before now we will take a look at the backend processing that takes place once a user submits a form. We should first specify that each page's php processor is entirely unique, this is due to a problem that was encountered during development due to our Foreign key constraint. However for all of our pages the first step in the php file is retrieve the imputed data and store it into a variable. Since we used the post method to retrieve this data the following code is an example of the process of storing the data in our file.

```php
//Reservation variables
$GuestID = $_POST['GID'];
$Rnum = $_POST['Rnum'];
$CIdate = $_POST['CIdate'];
$CODate = $_POST['CODate'];
$Tnum = $_POST['Tnum'];
```

The next step was to establish a connection to our database, in class we were shown how to use MySQL to handle this connection, however in our php files we opted to use a PDO object to handle the connection. This decision was taken largely due to us not being able to get our

databases to appear in the **localhost/myadmin**

```php
// The code below is for connecting to the database
$dsn ="mysql:host=localhost;dbname=hotel_reservation_sys";
$dbusername = "root";
$dbpsw = "[redacted]";

        // this sets the connections
        try{
    $PDO = new PDO($dsn,$dbusername,$dbpsw  );
            $PDO->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            echo"Connection Sucess";
}
catch(PDOException $e){
```

We recommend that anyone hosting our website locally should go into each php processor file and make sure the information above is correct for them. The **$dbpsw** data should be changed since this variable holds the password to your mysql instance.

Once a connection is established we transition into writing our SQL queries and executing them from our php file. For our queries we create a variable called SQL and store them in there. We also create a prepare statement to begin the process of executing the query, then we execute.

```php
$sql2 = "Select ResvID from guest where GuestId  = $GuestID";
$stmt = $PDO->prepare($sql2);
 $Res = $stmt->execute();
```

We mentioned earlier that we ran into a development problem due to our foreign key constraints, the issue primarily was that we couldn't insert data without violating our foreign key constraints. To solve this issue we made sure that all of our columns in the database could hold null values aside from the primary key and then we had to insert the data in multiple steps to ensure our foreign key constraints were not violated.

Ex.

```php
$sql3 = "update reservations Set Room_num = ?, Check_In = ?,  Check_Out = ? where GuestID = ?;";
$stmt = $PDO->prepare($sql3);
 $stmt->execute([$Rnum,$CIdate,$CODate,$GuestID ]);

 // this code will insert a new row into the transaction table
  $PM = "Debit";
 $sql = "Insert into transactions(Trans_ID,GuestID, amount, Payment_Method, Date) values (?,?, 100.00,?,? );";
 $stmt = $PDO->prepare($sql);
 $stmt->execute([$Tnum,$GuestID,$PM,$CIdate]);
 $Trans_num = $PDO->lastInsertId();

 // this code will update the reservation table with the transaction ID for the reservation
  $sql4 = "update reservations Set Trans_num = ? where GuestID = ? ;";
  $stmt = $PDO->prepare($sql4);
 $stmt->execute([$Tnum, $GuestID ]);
```

Depending on the form the order in which we insert or update the Database varies

User Guide:

1. **Visiting the Website**

- Open your web browser
- Type the website link in the address bar and press enter

2. **Home page**

- On the homepage, you will be able to see our hotel name, tagline, and below that the navigation bar to move around:

  Home – returns to this page

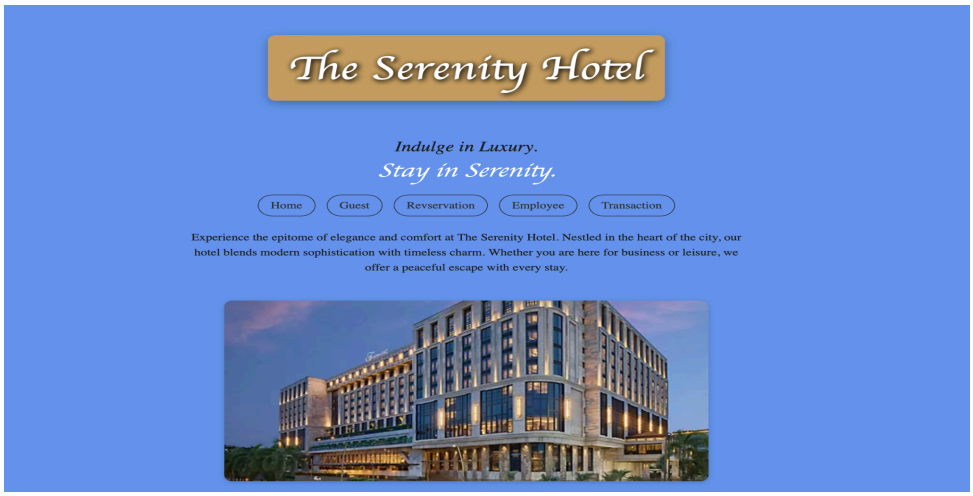
  Guest – guest information page

  Reservation – booking page

  Employee – employee section

  Transaction – payment section

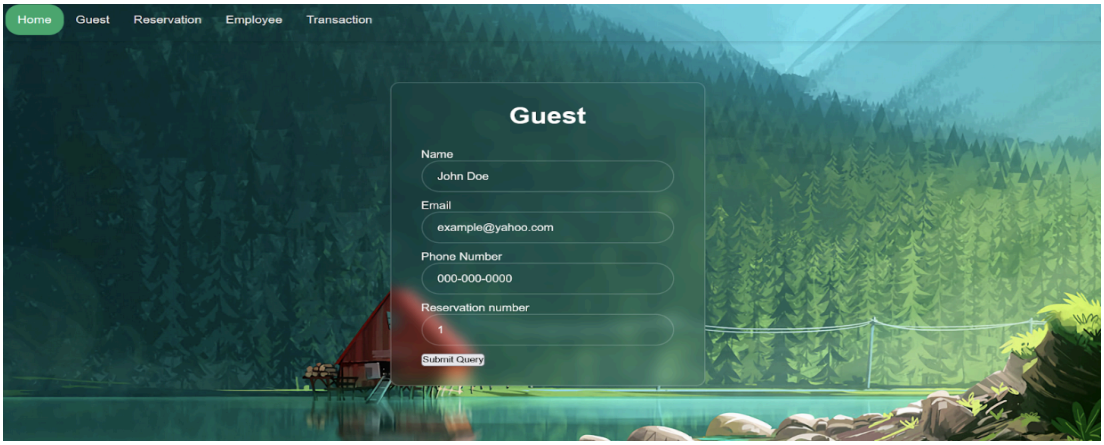
- Going down you'll see the hotel welcome message and below that you will see a gallery image of the hotel



3. **Guest Page**

- Next you're going to click on the Guest tab in the navigation bar
- When you enter the Guest page, you will fill out the following fields on the form:
  - **Name**
  - **Email**
  - **Phone Number**
  - **Reservation Number**
- Once you have completed the fields, you can click the **"Submit Query"** button to save your information

- You will then be directed to the next page saying “data inserted successfully.”
- Below that you will see a GuestID number that you must use in the reservation form
- Note, If you are inserting a new guest into the system, make sure that you fill out the reservation form. Inside the form, make sure to add the GuestID
- If you create a guest in the database on the backend you automatically create a reservation for that guest. The website will prompt you to go fill out a reservation form with the GuestID that you receive to fill out the rest of the reservation table

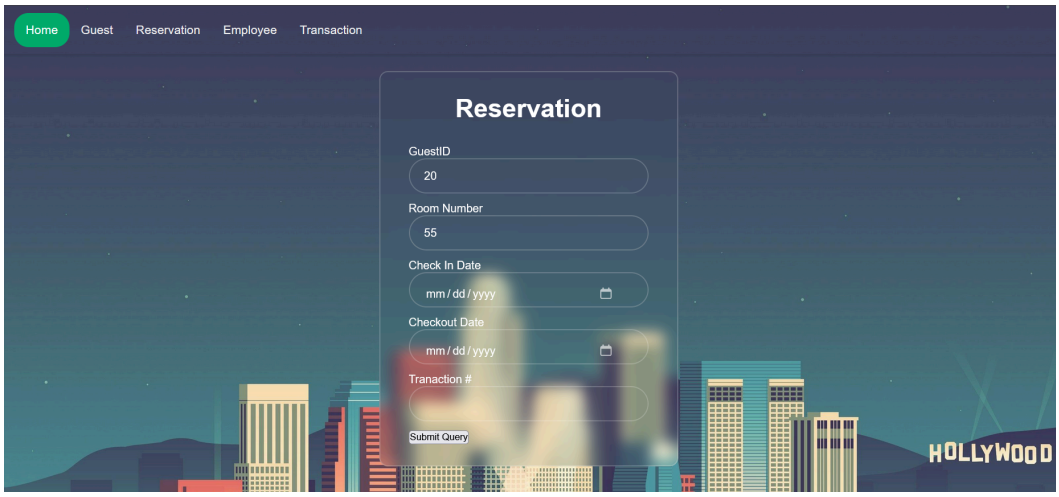


4. **Reservation Page**

- Next, click the Reservation tab in the navigation bar
- You will then fill out the following fields in the Reservation form:
    - GuestID
    - Room Number
    - Check in Date
    - Check out Date
    - Transaction (Note that for this section, you can fill out whatever number you want)

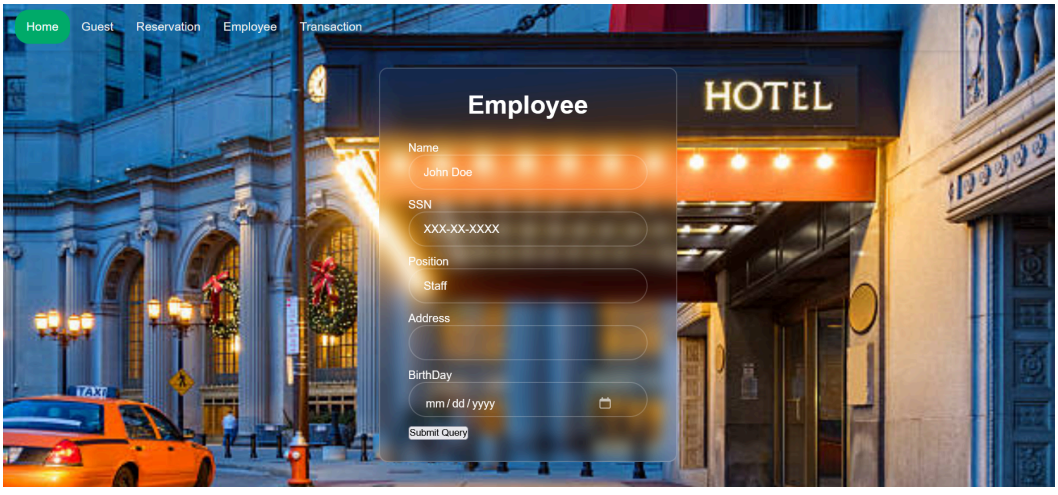
- When filling out the form, on the back end it’s going to check if there is already an existing guest with that ID in the guest table. If there is an existing guest we are going to update the already existing reservation for the guest with the information collected from the form
- Note: When submitting the form its going to create a new row in the transaction table with the TransactionID, GuestID, Amount, and Payment in the code
- If the server finds there isn't an existing guest with that ID, it's going to create one. Then it will populate that table with data collected from the form. The web page will prompt the user to fill out the guest form with information on the guest. It will provide the reservation ID for the guest. Upon completing the guest form, it will update the guest table, then we would have have to go to the transaction table

- After completing the fields, you'll click **"Submit Query"**



5. Employee page

- Next is the Employees page (which is specifically for Managers and Staff)
- Employees will click on the employee tab and fill out a form to enter their information
- The employees will then fill out the following fields:
    - Name
    - Social Security Number
    - Position
    - Address
- And after they'll click **"Submit Query"**



6. Transaction Page

- Lastly is the Transaction page. Click on the Transaction tab on the navigation bar.
- The form shown allows you to log your payment details
- Fill out the following fields:
    - GuestID

    - Amount
    - Payment Method
    - Date
- Note that the form will not allow you to submit a query if there is an incorrect GuestID
- After completing the fields, you'll click **"Submit Query"**