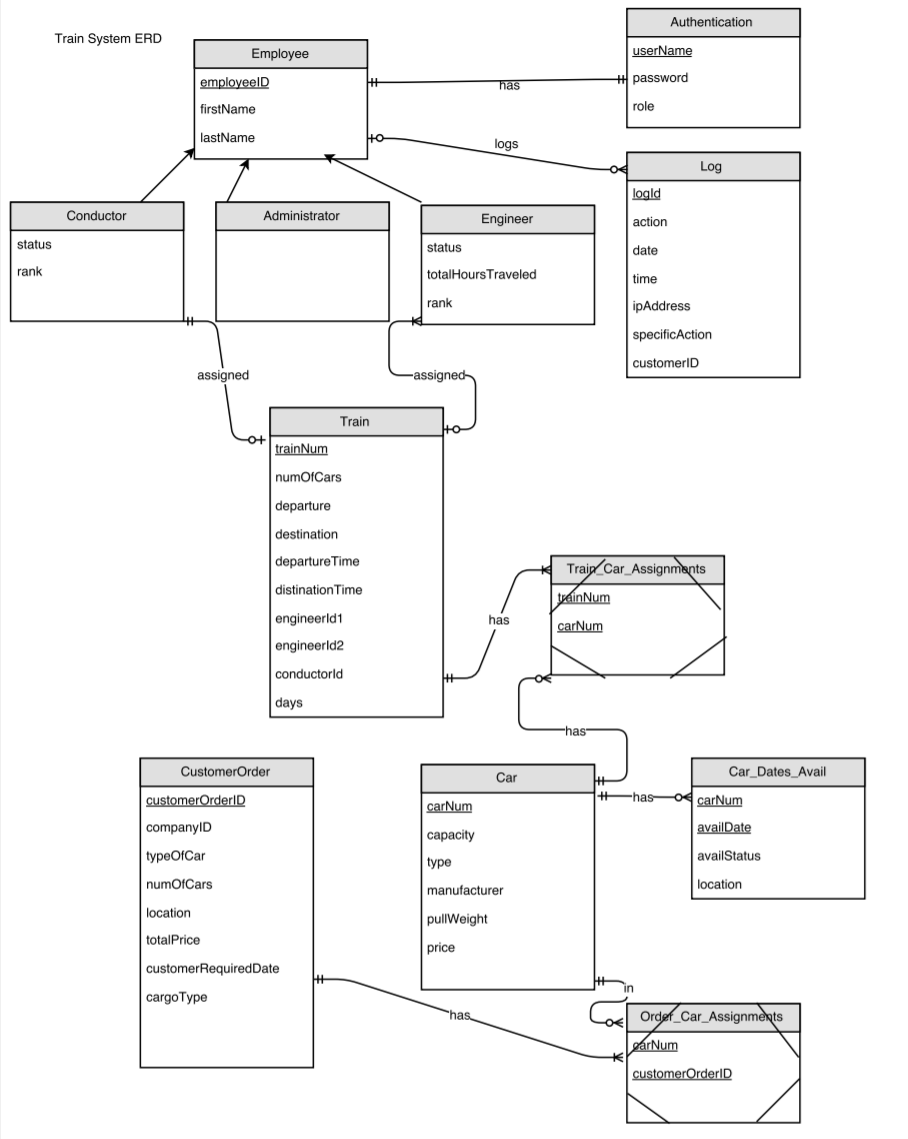
**Group 2: Final Report**

Members: *Candace Hicks, Brian Riddle, Aaron Neidlinger, Han Song, Lukas Heading, Tomer Negrin*

1. Create a website for establishing a railroad car system, using MYSQL specific database for the backend. With a front end that using html, and utilizes php to communicate with the server via queries. All while working as a group, and making it a user friendly experience.
   1. Website: <http://cs3380.rnet.missouri.edu/~GROUP2/>
      1. Engineer: **User:**Engineer **Pw:** test
      2. Conductor Sr: **User:** Conducto**r Pw:** test
      3. Conductor Jr: **User:** junior  **Pw:** test
      4. Admin: **User:** abc **Pw:** 123
   2. HTML(UI)/PHP-works with the implementation and design of the web application and handles back end communication.
      1. Candace Hicks: HTML/CSS/PHP for the customer booking page; HTML/CSS and PHP for the employee register page; HTML/CSS and PHP for the Engineer and Conductor pages.
      2. Tomer Negrin: HTML/CSS for the customer/administrator orders page.
      3. Aaron Neidlinger: Booking.php
      4. Han Song: HTML/CSS/PHP for the Login page; HTML/CSS for the index page; HTML/PHP for the Admin page; HTML/CSS for the about us page; co-working on Booking page;
   3. Database-implements the database using SQL:
      1. Lukas Heading: Built the initial ERD, helped explain database/table relationships, helped refined database
      2. Brian Riddle: Updated ERD, while getting table statements made, helped explain database/table relationships
2. ERD



**Engineer/Conductor User Experience (Candace Hicks)**

For the project I created the pages that Engineers and Conductors see when logging in. The project required that Engineers and Conductors should be able to access the system to see their information. I created a page for Engineers which shows their personal information (Employee ID, Hours Traveled, Rank, and Status) and also shows their past and current trains. I created a page for conductors that shows all of their personal information (Employee ID, Rank, Status) and their past and current trains. For both the engineers and conductors, I made it so that they could change their password and all of their personal information excluding employee ID and hours traveled.

The PHP for both pages was similar, just the querying of tables was changed slightly. The first thing I did is store the session variable so that I could query the Authentication table to get the Employee ID. I ran into some trouble when trying to store this value because stmt->get\_result returns an associative array, not a single value. So I used mysqli->fetch\_array with a foreach statement in order the store the employeeID into a variable. Afterwards, I queried the Engineer table (the conductor table for the Conductor page) using a select statement. I selected everything from the table so that all of the employee’s personal information could be displayed on the page. I added an extra column in the table for an edit button that when clicked would take so the employee to another page so that they could edit their information. Afterwards, I used a mysqli\_fetch\_field, mysqli\_fetch\_array, and echo statements in order to format the results in a tabular form for the user. Afterwards, I queried the Train table for the train number, number of cars, departure, destination, and time where engineerID1 or engineerID2 was equal to the employeeID (I queried where conductorID was equal to the employeeID for the Conductor page). This is what allows the employees to see their past and current trains.

I learned a lot more about PHP and how to communicate with databases from creating these pages. I learned how to properly make prepared statements, which is something I struggled with throughout the semester. I also learned how to format on a webpage using php.

**Database Design and Schema Creation (Brian Riddle)**  
 I was responsible for making a database which would meet the requirements for the Railroad Car System. The tables in the database and their relationships are documented in “Train System ERD and Table Documentation.” The solution to designing the database required that all entities and their relationships be identified from the requirements document. The relationships between the entities were represented in tables with primary and foreign keys which allowed the tables to be linked. Entities with one to many relationships such as the train to car and the car to train needed a link table to allow the relationship. All this information was placed in the ERD and the SQL statements to create the schema were written based upon the information in the ERD. The tables were created and populated with data.  
 The train and car relationship was difficult to hash out, because I had not thought out how the train system would actually work. I had not realized that a physical train could have more than one train number, and that the departure and destination cities actually determined the train number. The initial design placed the train number in the car table as a foreign key and assumed that a car could only be linked to a single train represented by a train number. However, this required that the departure and destination cities be updated on the train table whenever the physical train changed locations. I had also placed the location in the car table which would also have to be updated at the end of each trip. After a team member pointed out that information was going to be lost with the original design I realized it was not workable.   
 The location was moved from the car table to the table containing the availability dates for the car, because the location of the car changes by date as the train travels from its departure city to its destination city. The relationship between the car and dates available table stayed the same. A link table was created between the train and car tables which allowed that a train can have many cars, and that a car may have many train numbers. This was a workable design and information would not be lost. It would also allow cars to be reserved based upon location, destination, and date.  
 I can brag that my team has a workable design, enough data in the tables to test the system, and an ERD that matches the schema. I learned many things. I learned that database design is hard because systems are not simple and there are a lot of relationships between entities in the system. However, I learned that if a mistake is made, tables can be altered using simple statements. The only difficulty I had was in dropping a foreign key because I didn’t know the constraint name had to be used instead of the field name with the ‘DROP FOREIGN KEY’ statement. Also since I did not create the foreign key using a ‘CONSTRAINT’ statement I was not aware of how to find the constraint name. I also learned that communication is a must between team members. If a team member had not brought up that my original design had a problem then the team would not have a design that was workable. The last thing I learned was that using an online ERD drawing tool can be really frustrating, especially trying to connect the entities.

**Database Design and Schema Creation (Lukas Heading)**

I was one of the two people responsible for making a database which would meet the requirements for the Railroad Car System. Brian made a nifty documentation sheet called “Train System ERD and Table Documentation.” which is included on the flashdrive, which further explains our ideas. I made the initial ERD and database design we based everything off of, I really like working with ERDs because you get to see the flow of everything. I also learned that it is not always easy though, Brian did help with fixing issues with our ERD/Database as well as other teammates giving good input. A big thing is getting everyone on the same page, we found ourselves arguing over in what style we wanted the trains to work for orders, and that made me realize teamwork, collaboration, and different views are so important when working together with other people. Because this allows you to think of so many more ideas, and make sure you have the best. After getting our database up and running we filled the database with data so we could test and make sure everything worked out. We also had to do some updates when we found stuff didn’t work sometimes.

Also being one of the people in charge of the ERD/Database I found myself explaining what we meant by things in the ERD/Database. I would also help by explaining variables in the database, and how they should work to fellow group members. I also helped teammates with sql/php command lines and formatting them to work in the best way.

I would like the brag about my team above all else, we came together and everyone contributed, there was never a real issue of anyone not doing what they were supposed to. And if someone didn’t know how to do something they were never afraid to ask. I have also heard the horror stories of groups from other groups where some people never contributed and it fell on the few. I am also proud of the ERD, even though there were issues with it, we fixed them and made it work.

**Booking Pages (Aaron Neidlinger)**

During the project I primarily worked on the booking pages. The system required a way for users to book a trip. For our implementation, we decided to create a table of premade trains that run regularly to and from various destinations. Users are able to order cars to be tacked onto the train through the booking pages. The first booking page asks for various user input, then uses prepared statements to display in table format the cars that can be ordered and then trains that can be used that match the user input. For each entry in the train table, an additional button is displayed which allows users to select a train to be used in their order. Once a train is selected, the user is redirected to the confirm booking page. The booking page also passes various user input to the confirm booking page to be used in the table inserts. Using hidden inputs, confirm booking receives train number, the number of cars, the cargo type, type of car, trip date, company ID, departure, destination, and total price of order. I used prepared statements to query the database with the user input. In addition to this, I used the like operator for one particularly difficult part. Because we stored the days that a train runs in the format of , Monday/Wednesday for example, the day of the week had to be determined by using a date . From the user input of a date for their trip, I had to convert the yyyy/mm/dd into a day of the week using $day=strftime("%A",strtotime($trip\_date)). After the day of the week was retrieved, I concatenated wildcard operators onto either side of it and then used it as a parameter for the like clause in the prepared statement. After the user selects a train, they are taken to the confirm booking page. This page summarizes the order contents, gotten from the user and allows them to review an order a final time before it is placed. Once the user places the order, the php updates the various orders tables. Once this is done, an order if officially in the system.

I had a lot of fun working on this project. Not only did I learn a lot about PHP in a way that labs and class couldn’t, but my teammates made a really strong team. This was my first real experience with a large team working on a single project and it showed me the difficulties and rewards of having a team to rely on. Our team worked really excellently together, and I was especially proud of that.

**Order Page (Tomer Negrin)**

Aside from the general task of basic collaborative consulting for design and implementation for our database, which were all adamant on, my main focus was the orders page of the website. The webpage was created using html for layout, css for style, and php for construction/execution of SQL queries and all other interactions with the server. The order page is designed to tailor to the type of user that is accessing it and show the orders they have mad or have access to, and the information specific to those orders. If a customer, which requires no authentication, opens the order page, they will be prompted with an empty table and a search bar requiring a previously set companyID. If the companyID entered already exists in our database, the table will fill in all applicable orders tied to that companyID using php to run a selection query on the CustomerOrder table in our database. In addition I have given the users the option of filtering their results according to a few potential search inquiries (Ex: orderID, car type, cargo type, etc.). This is essentially the extent to which a customer and conductor/engineer are able to utilize this page barring navigating to other parts of the website. However, utilizing a session to ensure the authentication of a logged in administrator, I have provided certain features that are reserved strictly for administrators within the orders page. In contrast to startup by a customer or lower level employee, when an administrator loads the orders page, the table is automatically filled with all orders stored in our system. The same filter options are available for the administrator, however, in addition to this they have an update and delete button for altering the system data as they see fit. My overall goal with this webpage was to cater to the system requirements, but to do so in a very user friendly and clean looking design.

In regards to my team, I feel as if every single member was always ready to step up to the plate and take on a leadership role. Whether it be in the task they were designated or consulting in a different aspect of the project, noone was ever afraid to make suggestions and try to improve the overall design and experience. On the flip side as well, when different group members had uncertainties, we all had a very good rapport and desire to construct a functional system so no one was afraid to ask questions. Accountability is key in a group setting and I feel that every member delivered on what they were designated. Through this group design, I was fortunate to learn a lot about how to collaborate efficiently with colleagues and combine differing ideas for the best possible outcomes. And on the technical side, I feel as though this project has offered me so much more knowledge and experience in webpage design across html, css, and php to the point where I can now feel confident constructing my own and understanding how others’ work.

**Login/Main/Admin page (Han Song)**

For the project, I am responsible for the Login page, main page, and Admin page. The assignment required us to implement the functionality as an Administrator; In that case, the Administrator can view the employee logs with their names and Authentication information. Also, the Admin has the right to view the IP addresses from the employee who logged in our system, and Admin can change the username or reset the password. For the Login page, Employees can use their unique username to login to the system, and different roles will be directed to different pages; For example, Admin will be led to the Admin page.

The UI of the html are similar for those three pages which have the same background color and same UI buttons. The PHP for the Login page is to query the tables from the database which are Employee and Authentication table. The first thing I did is that strat a session and check if the ‘username’ and the ‘password’ being set; and then select the specific information such as username and password from the database to check if the username that the user typed in fits the one that in our database; If 0 information being found in the database, the message with failure will display on the screen. Otherwise, the code will keep checking if the password is same with the one in the database. For this part, I used mysqli\_stmt\_bind\_result to store the password and other datas with variable names; then I used the hash\_password\_verify to check if the user typed in password meets the one in the system. If not, the failure message will display; otherwise, the \_Session will start; IP address will be stored in the Log table in our database with all of this user information and what he did. For the Admin page. I use different PHP code to function the requirement. I had two buttons in the page; one is to display employee logs, another one is to display the IP logs. To display the employee logs, I made a view in the database with all of the data that we want. In addition, Administrator has right to delete and update employee datas, so I used the delete\_on\_cascade to implement this.

I learned a number of PHP functionality and how it communicates with the database. Such as execute queries and bind parameters.

* 1. Home Page: At the top of the page their is a bar, that has tabs with multiple links such as Booking, Orders, Trains, About, User Register, and Login
  2. Booking: A company provides its company ID and then have a selection of filters to pick from, and then based on that car options are provided to the user.
  3. Orders: Customers can enter their ID or their order ID and search past orders. There is a selection of filters as well to narrow down the orders that are shown.
  4. About: Talks about the developers on the site and our respective jobs
  5. Logs: Logs is a table stores data such as IP address and Employee information
  6. Register: For new employees, requires that their employeeID is already in the database in the employee table. Allows them to sign up for the site so that they can access their information.
  7. Login: An employee would enter a username, and a password and click login. There is also a link to the register page if no current credentials.
  8. Logging in as an Administrator: After logging in, you will see two button on the page. One is for searching employee information, and another one is to show the IP address logs with employeeID and what specific action they did to our system.
  9. Logging in as an Engineer: After logging in, you will see your personal information as well as past and current trains that you run. If you want to edit your personal information you click edit and then you will be redirected to a page where you can change your rank, password, or status.
  10. Logging in as a Conductor: After logging in, you will see your personal information as well as past and current trains that you run. If you want to edit your personal information you click edit and then you will be redirected to a page where you can change your rank, password, or status.
  11. Issues, frustrations, areas of difficulty.
      1. An online ERD drawing tool can be really frustrating, especially trying to connect the entities.
      2. Understanding how to use PHP to communicate with the database and website
  12. What did you collectively learn or get out of the group project?
      1. A good team is a must, a good team helps balance the workload.
      2. Communication is a must between team members.
      3. A good division of Labor
  13. What would you do differently next time?
      1. Start the project earlier
      2. Learn more PHP on my own

1. We came together to build a working website utilizing an html/PHP front end and a MYSQL database schema. And not to brag, but we would like to say it is pretty good looking and works pretty well, we could see a small business using a setup similar to ours. We learned a lot doing this project, the labs definitely helped us get here. We do wish we had started PHP sooner in the semester, there is a lot to learn, all of which has its use. We all had issues, frustrations, and or difficulties and we overcame these to finish our projects, and learning along the way. A good team is key to a good project, and our team came together and communicated well.