

Eagle Athletics Database Visualizer

Team 2 - Database Boys

Submitted By:

Nathan Agcaoili - na01974@georgiasouthern.edu

Changgyun Han - ch28202@georgiasouthern.edu

Earnest Hall - eh01014@georgiasouthern.edu

Elijah Horowitz - eh04987@georgiasouthern.edu

Hayden Spinos - hs09208@georgiasouthern.edu

Reviewed By: Team 3

Paul Thomas - pt03837@georgiasouthern.edu

Joel Thompson - jt10280@georgiasouthern.edu

Scott Walker - sw25350@georgiasouthern.edu

John Williams - jw17181@georgiasouthern.edu

Hazoof Alahmadi - ha01045@georgiasouthern.edu

Document Version Dates

- Current Version Date: 4/25/2021
- February 24, 2021
- February 19, 2021
- February 12, 2021
- February 5, 2021
- January 25, 2021
- January 22, 2021
- January 19, 2021

Reviewal Dates

- February 12, 2021
- January 28, 2021

Table of Contents

1. Title Page	1
2. Version History.....	2
3. Narrative Description.....	4
4. Information Needs.....	5
5. DBMS Architecture.....	5
6. Entities	6
7. Relationships.....	6
8. Possible Informal Queries and Update Operations.....	7
9. Additional Views.....	8
10. Possible Interfaces.....	8
11. Integrity Constraints.....	9
12. Schema Diagram / Tables	10
13. Possible Forms and Reports.....	11
14. Wire Frames.....	14
15. ER Diagram / Conceptual Design.....	20
16. Logical / Physical Design.....	21
17. Example Print out of Form, Report, and Query.....	24
18. Our Experience During the Team Project.....	27
19. Installation Instructions.....	34
20. GitHub Repository/Source Code.....	47

Narrative Description

Eagle Athletics Database Visualizer (EADV) is a web-based application for a made-up athletic clothing company, Eagle Athletics, that visualizes stored company transaction information details for both employees and clients into their respective tables within a database. The application will keep track of information on employee sales (revenue earned, quantity of product, sold to which client, etc.)

The tables within the database will be related by common attributes. For example, the employee table will have a primary key of 'employee_id' and one of the columns of said table will refer to what branch of the company the employee works for titled 'branch_id'. This branch column will serve as a foreign key to another table that contains the information on all the branches of the company. Within the branch table, there will be another foreign key column for the manager of the branch called 'manager_id', which corresponds with the 'employee_id' of the manager, thus leading back to the employee table.

Information Needs

- Company information
 - Solid understanding of the fundamentals of our made-up company, Eagle Athletics
- Web development technologies and procedures
 - Solid understanding of both front-end and back-end development for designing and developing a UI and writing and retrieving information from a stored database

DBMS Architecture

- Three tier Client/Server Architecture, the business logic is placed in application server or webserver. Basic and two tier architecture are not going to be used because if business logic is placed in a database server or client, there will be a burden.

Entities

1. Employee
2. Department
3. Branch
4. Customer
5. Products
6. Sales Relations
7. Supplier Shipments
8. Supplier
9. Region
10. Order History

Relationships

- Each EMPLOYEE works with CUSTOMERS
- CUSTOMERS and EMPLOYEES both can view ORDER HISTORY
- Each EMPLOYEE works in BRANCH and DEPARTMENT
- Each BRANCH has DEPARTMENTS
- Each BRANCH stores PRODUCT
- Each BRANCH is located in a REGION
- Each SUPPLIER supplies PRODUCTS to BRANCH
- SUPPLIER and EMPLOYEE both view SUPPLIER SHIPMENTS
- EMPLOYEE handles SALES RELATIONS and SUPPLIER SHIPMENTS
- One EMPLOYEE supervises multiple other EMPLOYEES
- One EMPLOYEE manages a BRANCH

Possibles Informal Queries and Update Operations

- Queries
 - Checking order histories
 - Checking remaining inventory
 - Viewing previous supplier shipments
 - Checking on sales relations between employees and customers to check performance
 - Pulling up customer information and contact information
 - Pulling up a supplier's contact information
 - Pulling up an employee's information (includes all attributes such as name, age, sex, salary, branch, supervisor, and department etc...)
- Update Operations
 - Updating order history (adding new orders)
 - Updating Inventory for specific products
 - Updating supplier shipments (logging new shipments)
 - Updating the sales relations between employees and customers to track performance
 - Creating/ changing customer information for new/ existing customers
 - Creating/ changing supplier information for new/ existing suppliers
 - Creating/ changing employee information for new/ existing employees
 - Changing the price for certain products
 - Changing branch suppliers

Additional Views

- We hoped to have multiple views for customers, employees, and supervisors.

Where through a login function we would determine if the user is a customer, employee, or supervisor, and then show them the correct view so that they could only complete certain tasks that would edit the database. However due to time constraints during development of the implementation we were unable to add them.

Possible Interfaces

These may not all be in the implementation and are just possible ones listed through brainstorming

- For the customer, they would require an application that gives them the ability to manage their orders and to contact their sales representatives.
- For the employee, they would have an application that similarly allows them to manage orders and contact customers. They would also need a similar application to manage shipments and contact suppliers.
- For the supervisor, they would have an application that allows them to view the productivity of their employees and manage products, branches, departments, and suppliers.
- For the supplier, they would require a similar application to the customer except with access to supplier shipments instead of order history.

Integrity Constraints

sex: must fit a char of either 'M' or 'F'

contact_information: must be a int of index 9

price_per_unit: must not have more the two decimal places

delivery_date: must be after the order_date

Region_name: must be one of

"West", "MidWest", "SouthWest", "SouthEast", "NorthEast"

employee_id: must be unique

department_id: must be unique

branch_id: must be unique

customer_id: must be unique

product_id: must be unique

region_id: must be unique

supplier_id: must be unique

Schema Diagrams / Tables

Employee								
employee_id	first_name	last_name	birth_date	sex	salary	supervisor_id	department_id	branch_id
1001	John	Doe	04/09/1968	M	\$410,000.00	NULL	0002	10
1002	James	Smith	07/22/1984	M	\$120,000.00	1001	2221	12
1003	Sally	Johnson	02/15/1992	F	\$52,000	1002	2222	12

Department		
department_id	department_name	department_head_id
0002	Corporate	1001
1221	Human Resources	1499
1222	Sales	1599
2221	Human Resources	2499
2222	Sales	2599

Branch			
branch_id	branch_name	branch_head_id	region_id
10	Dayton	1001	1400
20	Scranton	2001	2400

Customer				
customer_id	first_name	last_name	customer_address	contact_information
1001	Jane	Doe	1002 Neverland Rd	nsdad@yahoo.com
2002	Chris	Smith	2463 Dunken Rd	ChrisS547@hotmail.com

Supplier			
supplier_id	company_name	contact_information	address
1001	Adidas	supplier@adidasl.com	1965 Eagle Court
2002	Nike	supplies@nike.com	123 Office Row

Products			
product_id	product_name	price_per_unit	inventory
14401	Adidas hat	\$15.00	1,500
14402	Nike shirt	\$35.00	750

Sales Relations		
employee_id	customer_id	total_sales
1123	1001	\$75.0
1124	1001	\$150.0

Supplier Shipments				
branch_id	supplier_id	shipment_id	product_id	shipment_date
1001	1001	100808	1001	12/14/2021

Region		
region_id	region_name	region_sales
1	SW US	\$880,888.00

Order History				
customer_id	order_id	product_id	order_date	delivery_date
1001	110080	1001	12/14/2021	12/25/2021

Possible Forms and Reports

- Forms
 - These forms listed may not all be in the final implementation as they are just the “possible” ones and may not be used.
 - The first form would be a purchasing form where users could fill out information on what products they would like to purchase. This would change values for inventory of the products they ordered, create new entries in order history, and add on to total sales.
 - Another form would be a contact information update form, where the user can update contact information and addresses for clients/customers.

- A third form would be used when adding a new employee to the company. This form would have text fields to enter the new employee's name, salary, and assign a supervisor. This form would also have drop downs to assign a branch and department to the employee profile. There will be a calendar option to select the employee's date of birth and a drop down menu to select their sex.
- A form to update/change employee information, like if they move or change salary or job.
- A form to update department head for branches.
- A form to create new customers or update customer information inside the database.
- A form to add new supplier or update current supplier information
- A form to update product price
- A form to add a new product to the database.
- Reports
 - For our reports, they will be generated by clicking on buttons through a navigable menu. For example, to generate a report containing an employee's information, the user would navigate to an employee list and all employee information will be available there. Also these forms listed are just ones that we thought would be possible and all of them may not be in the final implementation.
 - A report for checking order histories
 - A report to check remaining inventory
 - A report to view previous supplier shipments
 - A report for checking on sales relations between employees and customers to check performance
 - A report for all customer information and contact information
 - A report for supplier's contact information
 - A report for seeing employee's information (includes all attributes such as name, age, sex, salary, branch, supervisor, and department etc...)
 - A report for employee sales within a set date window

- A report for region sales within a set date range
- A report for customer purchases within a set date range
- A report for branch sales within a set date range
- A report for products a supplier provides and information on those products

Wireframes

Home Page

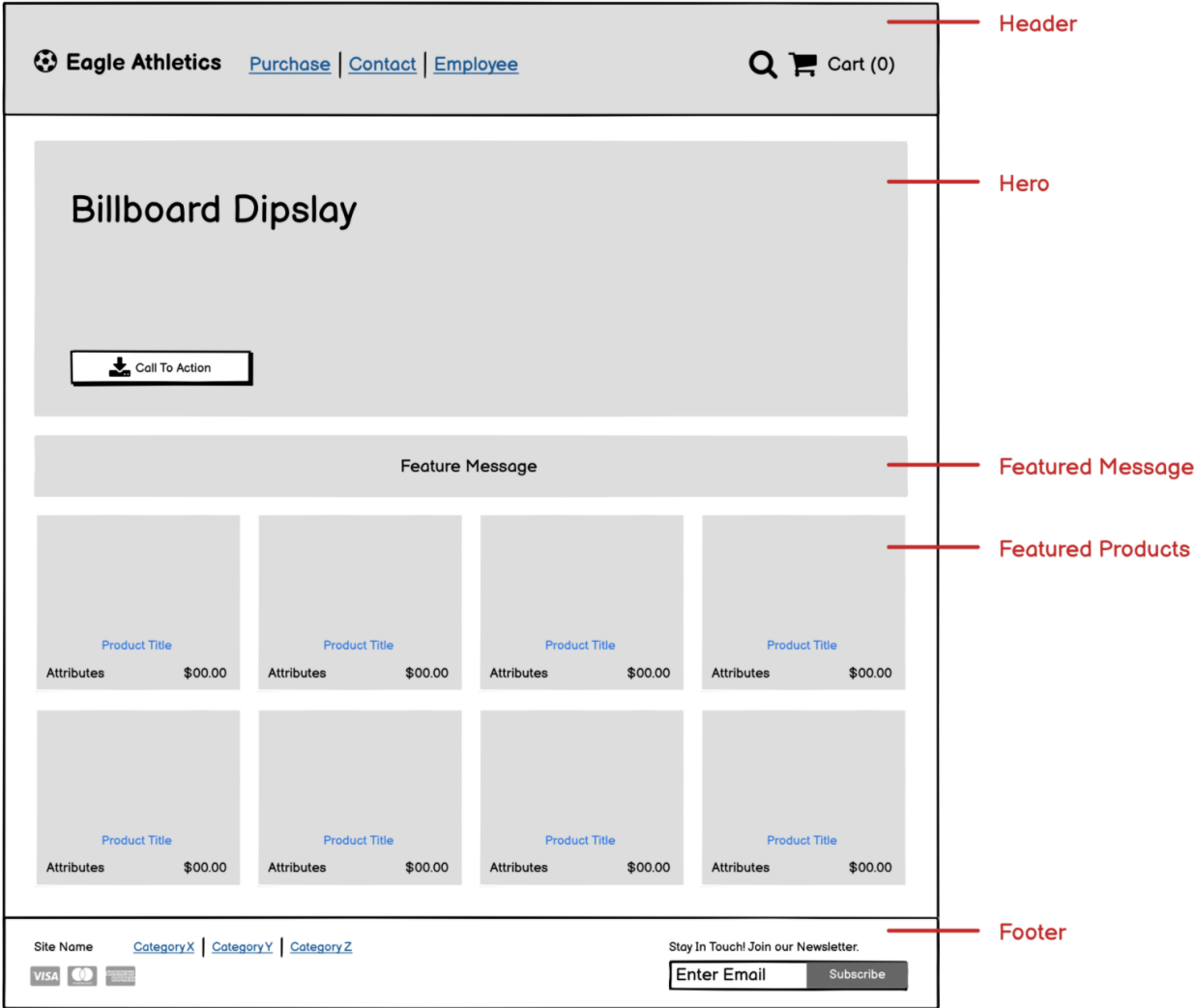


Figure 1.1 : Homepage Wireframe

Product Page - Add To Cart

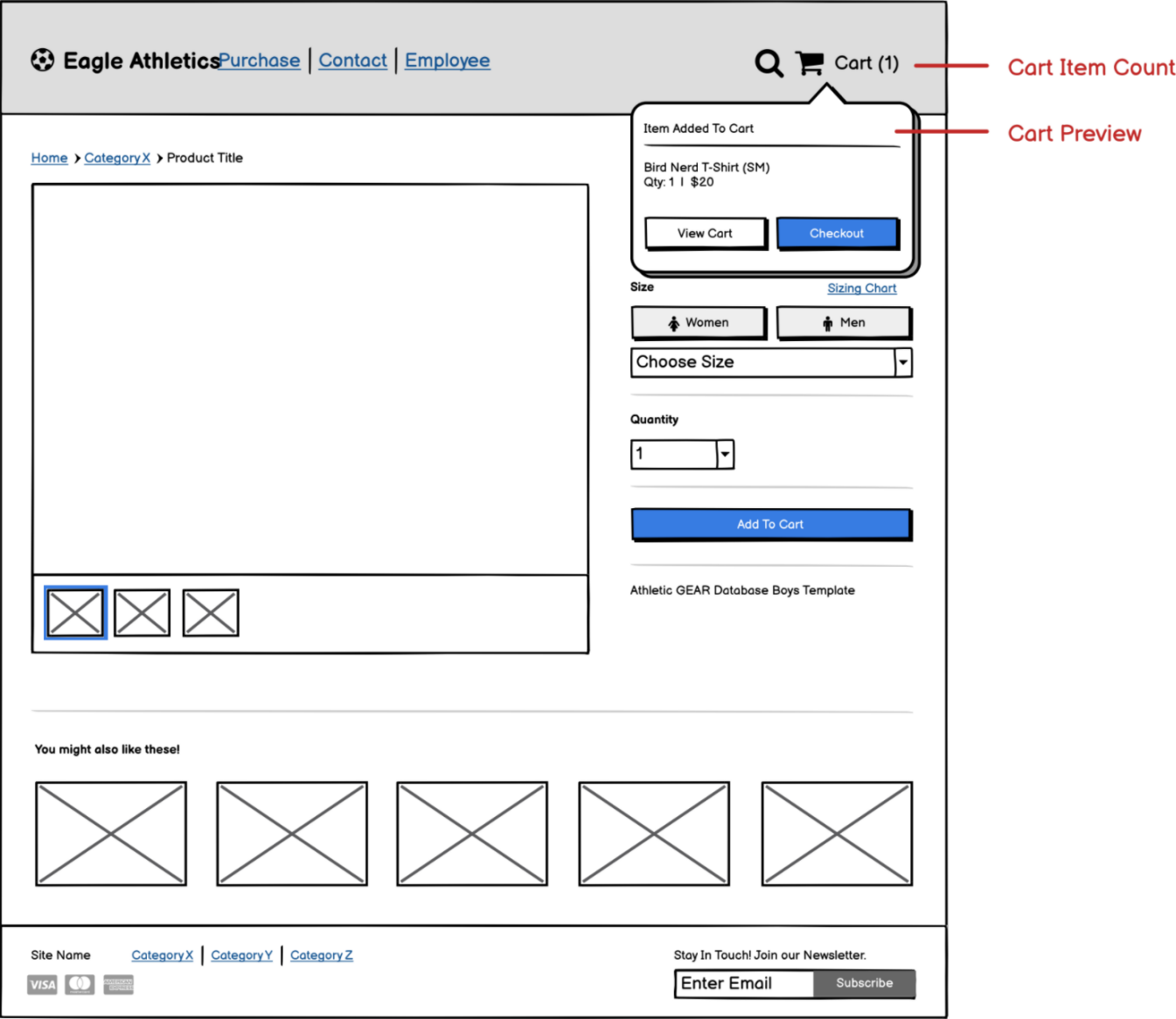





Figure 1.2 : Product Page Wireframe

Figure 1.3 : Checkout Page Wireframe

Update Form copy


[Purchase](#) | [Contact](#) | [Employee](#)


 Cart (1)

Employee

Employee Information Already have an account? [Log In](#)

Email

Address

First Name Last Name

Company (optional)

Address Apt. (Optional)

Country State Zip




Some text

Dates

JANUARY 2021

S	M	T	W	T	F	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						


Site Name [Category X](#) | [Category Y](#) | [Category Z](#)







Stay In Touch! Join our Newsletter.


Figure 1.4 : Example Update Form Wireframe

ADD Employee copy

 **Eagle Athletics** [Purchase](#) | [Contact](#) | [Employee](#)

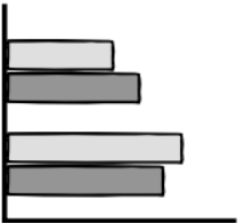
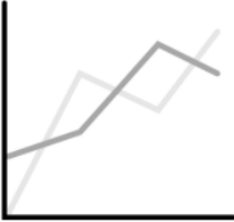
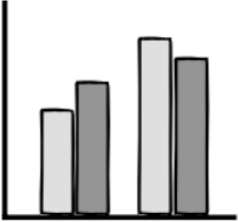
  Cart (1)

Select Employee ▼
Susie Grey
Gabby green
David White






☐ not selected
☒ Employed?
☐ Terminated
☐ Sick
-[x] Vacation
☐ Position
A row without a checkbox

Show Employee Info



Site Name [Category X](#) | [Category Y](#) | [Category Z](#)


Stay In Touch! Join our Newsletter.



Enter Email


Subscribe

Figure 1.5 : Add Employee Form Wireframe

Checkout - Success

 **Eagle Athletics** [Purchase](#) | [Contact](#) | [Employee](#)

  Cart (1)



Order 131217312
Thank you, John!

Your order is confirmed.


We've accepted your order and we're getting it ready.

Customer Information

SHIPPING ADDRESS
John DOE
[1234 EAGLES LANDING](#)
[STATESBORO ,GA 30458](#)
[United States](#)

Billing Address
Jane DOE
[12 Waldo Point Road](#)
[Upstate, NY 11200](#)
[United States](#)




Shipping Method
UPS Ground (Estimated ship
time of 3-6 days)

Payment Method
 Ending in 3217 — \$23.60

Summary (1 item)

Subtotal	\$20.00
Shipping	\$2.20
Est. Taxes	\$1.40
Gift card or discount code	
<input type="text"/>	<input type="button" value="Apply"/>
Total	\$23.60

Site Name [Category X](#) | [Category Y](#) | [Category Z](#)

Stay In Touch! Join our Newsletter.

Figure 1.6 : Checkout Confirmation Page Wireframe

ER Diagram/ Conceptual Design

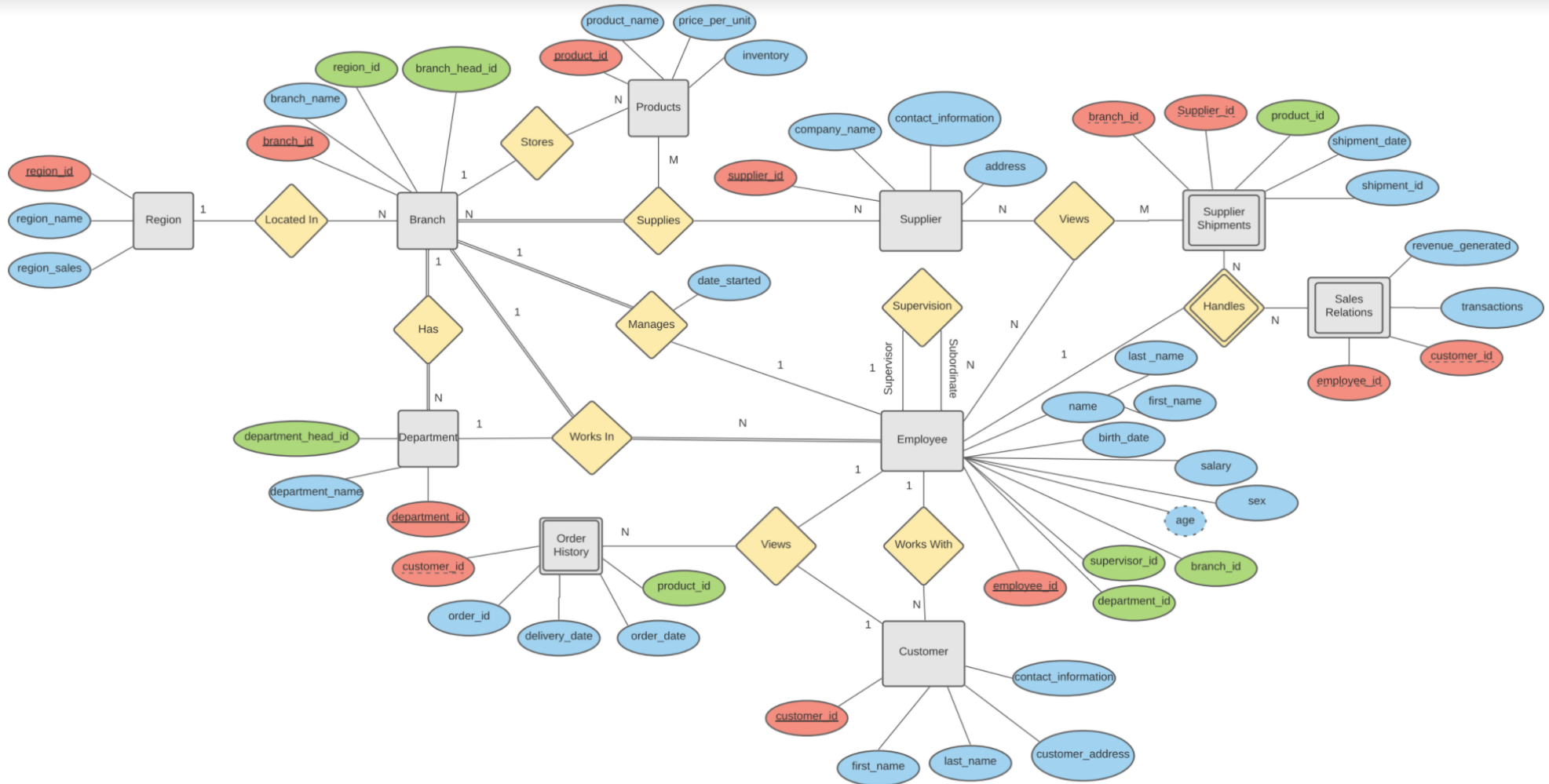


Figure 2.1 : ER Diagram for Eagle Athletics Database Visualizer

- Gray Boxes represent entities, blue ovals represent regular attributes, green are foreign keys, red are primary keys, and yellow diamonds are relationships

Logical/Physical Design

Relational functions

EMPLOYEE

Employ_id determines first_name

Employ_id determines last_name

Employ_id determines birth_date

Employ_id determines sex

Employ_id determines salary

Employ_id determines supervisor_id

Employ_id determines department_id

Employ_id determines branch_id

Employee_id -> {first_name, last_name, birth_date, sex, salary, supervisor_id, department_id, branch_id}

DEPARTMENT

Department_id determines department_name

Department_id determines department_head_id

Department_id -> {department_name, department_head_id}

BRANCH

Branch_id determines branch_name

Branch_id determines branch_head_id

Branch_id determines region_id

Branch_id -> {branch_name, branch_head_id, region_id}

CUSTOMER

Customer_id determines first_name

Customer_id determines last_name

Customer_id determines customer_address

Customer_id determines contact_information

Customer_id -> {first_name, last_name, customer_address, contact_information}

SUPPLIER

Supplier_id determines company_name

Supplier_id determines contact_information

Supplier_id determines address

Supplier_id -> {company_name, contact_information, address}

PRODUCTS

Product_id determines product_name

Product_id determines price_per_unit

Product_id determines inventory

Product_id -> {Product_name, price_per_unit, inventory}

SALES RELATIONS

Employee_id and customer_id determines total_sales

{Employee_id, customer_id} -> total_sales

SUPPLIER SHIPMENTS

Branch_id and supplier_id determines shipment_id

Branch_id and supplier_id determines product_id

Branch_id and supplier_id determines shipment_date

{Branch_id, supplier_id} -> {shipment_id, product_id, shipment_date}

REGION

Region_id determines region_name

Region_id determines region_sales

Region_id -> {region_name, region_sales}

ORDER HISTORY

Customer_id determines order_id

Customer_id determines product_id


Customer_id determines order_date

Customer_id determines delivery_date

Customer_id -> {order_id, product_id, order_date, delivery_date}


EMPLOYEE

<u>employee_id</u>	first_name	last_name	birth_date	sex	salary	supervisor_id	department_id	branch_id
--------------------	------------	-----------	------------	-----	--------	---------------	---------------	-----------




DEPARTMENT

<u>department_id</u>	department_name	department_head_id
----------------------	-----------------	--------------------




BRANCH

<u>branch_id</u>	branch_name	branch_head_id	region_id
------------------	-------------	----------------	-----------



CUSTOMER

<u>customer_id</u>	first_name	last_name	customer_address	contact_information
--------------------	------------	-----------	------------------	---------------------



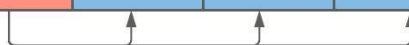
SUPPLIER

<u>supplier_id</u>	company_name	contact_information	address
--------------------	--------------	---------------------	---------




PRODUCTS

<u>product_id</u>	product_name	price_per_unit	inventory
-------------------	--------------	----------------	-----------




SALES RELATIONS

<u>employee_id</u>	<u>customer_id</u>	total_sales
--------------------	--------------------	-------------



SUPPLIER SHIPMENTS

<u>branch_id</u>	<u>supplier_id</u>	shipment_id	product_id	shipment_date
------------------	--------------------	-------------	------------	---------------




REGION

<u>region_id</u>	region_name	region_sales
------------------	-------------	--------------



ORDER HISTORY

<u>customer_id</u>	order_id	product_id	order_date	delivery_date
--------------------	----------	------------	------------	---------------



Example print out of Form, Report, and Query

We had forms for adding an instance of each entity to our database, however they are all the same format, just different names of attributes to avoid redundancy. We just showed the functionality of the employee operations, however we show the navigational screenshots at the end of the section and you can see each entity listed there.

← → ↻ ⓘ localhost/EADB/addEmployee.php

EADB

Add a user

Employee ID

First Name

Last Name

D.O.B.

Sex

Salary

Supervisor ID

Department ID

Branch ID

[Back to Employee Operations](#)

[Back to Homepage](#)

This is to add an employee to the employee list of the database, The next screenshot shows it filled out,

← → ↻ ⓘ localhost/EADB/addEmployee.php

EADB

Add a user

Employee ID

First Name

Last Name

D.O.B.

Sex

Salary

Supervisor ID

Department ID

Branch ID

[Back to Employee Operations](#)

[Back to Homepage](#)

Then you could search for the employee and get a report of their information

← → ↻ ⓘ localhost/EADB/readEmployee.php

EADB

Results

Find employee based on id

Employee Id

View Results

[Back to Employee Operations](#)

[Back to Home Page](#)

Employee Id	First Name	Last Name	D.O.B.	Sex	Salary	Supervisor Id	Department Id	Branch Id
1025	Jerry	Johnson	2021-04-01 00:00:00	M	150000	1001	2	10

Then you could go back to the employee list and edit an employee

← → ↻ ⓘ localhost/EADB/updateEmployee.php

EADB

Update an Employee

Employee Id	First Name	Last Name	D.O.B.	Sex	Salary	Supervisor Id	Department Id	Branch Id	Edit
1001	John	Doe	1968-04-27 00:00:00	M	410000	9000	2	10	Edit
1002	James	Smith	1984-07-22 00:00:00	M	120000	1001	2221	12	Edit
1003	Sally	Johnson	1992-02-15 00:00:00	F	52000	1002	2222	12	Edit
1025	Jerry	Johnson	2021-04-01 00:00:00	M	150000	1001	2	10	Edit

[Back to Employee Operations](#)

[Back to Home Page](#)

You then could select and employee and get a form to edit it

← → ↻ ⓘ localhost/EADB/update-singleEmpl

EADB

Edit a user

Employee_id

1025

First_name

Jerry

Last_name

Johnson

Birth_date

2021-04-01 00:00:00

Sex

M

Salary

150000

Super_id

1001

Department_id

2

Branch_id

10

Submit

[Back to Employee Operations](#)

[Back to Homepage](#)

The following screenshot is an example of our delete query, we show before and after pictures of the employee list and a report shown that will give you the option to delete any employee.

← → ↻ ① localhost/EADB/deleteEmployee.php

EADB

Delete Employee

Employee Id	First Name	Last Name	D.O.B.	Sex	Salary	Supervisor Id	Department Id	Branch Id	Edit
1001	John	Doe	1968-04-27 00:00:00	M	410000	9000	2	10	Delete
1002	James	Smith	1984-07-22 00:00:00	M	120000	1001	2221	12	Delete
1003	Sally	Johnson	1992-02-15 00:00:00	F	52000	1002	2222	12	Delete
1025	Jerry	Anderson	2021-04-01 00:00:00	F	150000	1001	2	10	Delete

[Back to Employee Operations](#)

[Back to Home Page](#)

After

← → ↻ ① localhost/EADB/deleteEmployee.php?employee_id=1025

EADB

Delete Employee

Employee Id	First Name	Last Name	D.O.B.	Sex	Salary	Supervisor Id	Department Id	Branch Id	Edit
1001	John	Doe	1968-04-27 00:00:00	M	410000	9000	2	10	Delete
1002	James	Smith	1984-07-22 00:00:00	M	120000	1001	2221	12	Delete
1003	Sally	Johnson	1992-02-15 00:00:00	F	52000	1002	2222	12	Delete

[Back to Employee Operations](#)

[Back to Home Page](#)

Navigational Screenshots

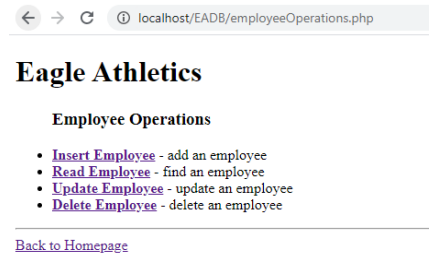
Our first picture here is a picture of the list of entities so you would select the entity you want to edit.

← → ↻ ① localhost/EADB/index.php

Eagle Athletics

- [Employee Operations](#)
- [Department Operations](#)
- [Product Operations](#)
- [Branch Operations](#)
- [Region Operations](#)
- [Supplier Operations](#)
- [Customer Operations](#)
- [Order History Operations](#)
- [Sales Relations Operations](#)
- [Supplier Shipments Operations](#)

Then you would be brought to the operations screen for that entity with a list of operations, functionality for each operation is what's shown first in this section.



Our Experience During the Team Project

1. Did you achieve whatever you planned in your project?

- Yes, we did achieve our main goal of learning about database systems and how to correctly design and implement them into a functioning application.
- We learned how to set up a web application project and gained insight as to what should be prioritized during the development process.

2. An assessment on the quality of your project. Feel free to discuss what parts of your project you felt are strong and what parts would need more work to bring up the quality.

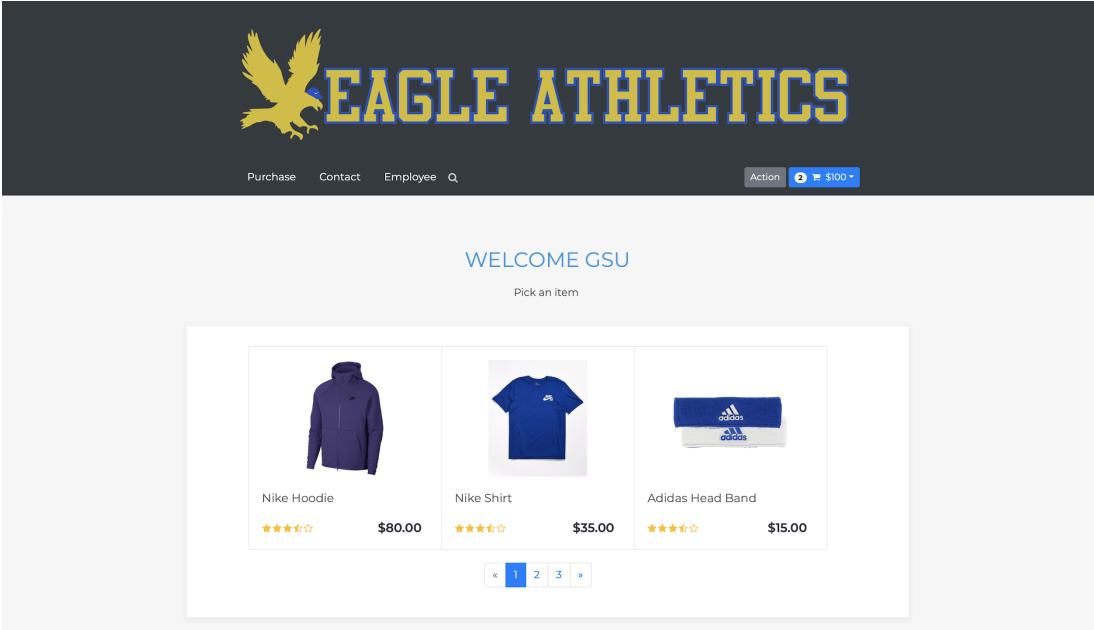
- We feel pretty confident in the overall quality of our project. With this being a fairly new experience for most of us, we feel proud of our submissions.
- One part of this whole project process that we feel strong about is our ability to work effectively and gel with one another.
- One part of the project that we could work on improving is the application programming. We still feel that it matches the quality of the rest of our project but full stack development was the most foreign topic to most of us.

3. How has your project evolved over time?

- We originally had 8 entities but our project got bigger and had 10 entities. Some of our tables and relationships had to be redefined as well.
- During the implementation phase of our project, we originally were too focused on the user interface. (As shown below)
- Concept Logo Art:




- Original Pages:



VIEW / DELETE

SELECT EMPLOYEE



Show Employee INFO Delete Employee INFO

Name	Position	Office	Age	Start date	Salary
Tiger Nixon	System Architect	Edinburgh	61	2011/04/25	\$320,800
Garrett Winters	Accountant	Tokyo	63	2011/07/25	\$170,750
Shou Itou	Regional Marketing	Tokyo	20	2011/08/14	\$163,000



The screenshot shows the Eagle Athletics website header with the logo in yellow and blue. Below the header is a navigation bar with links: Purchase, Contact, Employee, and a search icon. A status bar shows 'Action' and a currency selector set to '\$100'. The main content area is titled 'ENTER CUSTOMER INFO' in blue. It contains a form with fields for Email, First Name, Last Name, and Address. To the right of the form is a 'SUMMARY' table showing Subtotal (\$360), Discount (\$0), Shipping (\$0), and Total (\$360). Below the table are buttons for 'Send data' and 'Continue To Shipping'.

SUMMARY	
Subtotal	\$360
Discount	\$0
Shipping	\$0
Total	\$360

- While these pages were visually appealing and relatively matched our wireframe diagram, this ultimately ended up hurting us because a lot of the backend database functionality was not able to be properly implemented.
- We ended up having to rebuild the project from scratch due to the countless issues that the stylized version was giving us. This time around; however, we focused primarily on communicating with the database functionality.
- Even though our new web application currently doesn't contain styling like the original, it displays our knowledge of the core concepts of database design and implementation as it contains a lot of the functionality that our UI-friendly counterpart lacked.
- For the most part, the progress on our project grew at a steady pace in the intended direction.

4. Each team member's individual experience by doing this project: Changgyun 'Shawn' Han

4.1. Which parts were the most fun?

- The most fun part for me was when we were all working on making an ER Diagram.
- We were communicating and felt like we were working as a team.

4.2. Which parts were the most challenging? How did you solve those challenges?

- The project itself was not that challenging. However, with all the submission we had to do we needed more time working on the project.
- we had to meet more often.

4.3. Which parts were the easiest?

- Easiest part for me was schema diagrams and tables.
- It was a lot of work but was not hard to do.

4.4. What did you learn that you did not imagine you would have?

- Learned how to make time when I am having the busiest week of the semester.

4.5. If you had to do it all over again, what would you have done differently?

- I would suggest working on the programming part earlier than using all our time doing the documents we had to submit

4.6. What is your overall experience of working in a team?

- It was really helpful. I was having a hard time learning in class, so working as a team actually taught me how to do stuff.

4.7. Other comments and conclusions

- I wish there weren't as many documents to turn in and more time to work on the final product.

Nathan Agcaoili

4.1. Which parts were the most fun?

- The part of the project that I enjoyed the most was getting the opportunity to develop a project in a team environment while at the same time getting to know my group members and watch as our individual strengths work together to achieve our goals and make deadlines.

4.2. Which parts were the most challenging? How did you solve those challenges?

- Implementing our design into a functioning database application was the part that I found to be the most difficult, as the majority of our group members, including myself, didn't have any web/full stack development experience. We were able to overcome this challenge by dedicating a lot of our free time to self study and learning new tools and technologies such as the PHP programming language.
- Another aspect of this project that I deemed difficult was finding times in which all of our members can meet to work on the current tasks at hand. We overcame this by communicating with each other our generic availability and basing a weekly meeting time off of that. We are also sure to properly update everyone in the group if we are unable to meet for any specific reason so that we can collectively figure out a new meeting time for said week.

4.3. Which parts were the easiest?

- I found that working with my teammates was the easiest part of the project because we all equally pulled our weight to make deadlines and produce an application that we are proud of. There were no conflicts between members so it really streamlined the ability to take on the heavy workload with no issues.

4.4. What did you learn that you did not imagine you would have?

- One invaluable concept that I learned this semester by doing this project was the importance of self study. Without the extensive research on PHP and web development that we all dedicated outside of the classroom, we would have never been able to finish this project on time.

4.5. If you had to do it all over again, what would you have done differently?

- Knowing what I know now, I would have definitely started on the implementation portion of the assignment a lot sooner. In addition, I would not spend so much time on the UI until we have a functional backend, then add the styling after.

4.6. What is your overall experience of working in a team?

- As mentioned in my previous answers of 5.1 and 5.3, I had a very positive experience working with this group. I am very thankful that I had a reliable group of studious and knowledgeable people to work with. Without our conjoint efforts, we would not have the final product that we have today.

4.7. Other comments and conclusions

- Overall, I am very pleased with the information and skills I have learned through working on this project. Working with a team is a very important part of the software development industry so getting the opportunity to to experience that was great. In addition, I am proud of myself and my group mates for achieving the goals that we set for ourselves at the early stages of the project.

Earnest Hall

4.1. Which parts were the most fun?

- Interacting with my peers.

4.2. Which parts were the most challenging? How did you solve those challenges?

- Implementation of the mysqli function in php / mysql
Troubleshooting the database

4.3. Which parts were the easiest?

- Downloading the mysql engine

4.4. What did you learn that you did not imagine you would have?

- I learned that there are various ways to connect a mysql database , and that a ground up workflow implementation would be better that working on the front end first.

4.5. If you had to do it all over again, what would you have done differently?

- I would have started with less and requirements and features and added additional features when the time came.

4.6. What is your overall experience of working in a team?

- Great my team is great.

4.7. Other comments and conclusions

- Finishing exams before a final project submission would have been beneficial.

Elijah Horowitz

4.1. Which parts were the most fun?

- To me the most fun part was working with the team and trying to find solutions to our problems.

4.2. Which parts were the most challenging? How did you solve those challenges?

- I think the implementation was the most difficult because not many of us had worked with SQL or PHP before so it was new to all of us, and we may have unknowingly spent time on the wrong things because of this inexperience.

4.3. Which parts were the easiest?

- I think filling out and making a lot of the documents were easiest but they were really time consuming.

4.4. What did you learn that you did not imagine you would have?

- PHP because I didn't think we would be using it because it wasn't in the course material.

4.5. If you had to do it all over again, what would you have done differently?

- Start on the implementation way earlier because we felt like we needed way more time to implement everything we wanted to implement

4.6. What is your overall experience of working in a team?

- I loved working with my group, we all put in the work when we needed to, and everyone tried their best to make sure the project was as best as we could make it with the given time.

4.7. Other comments and conclusions

- Like Shawn I do kinda wish there was less time spent on the graphs and documents and more time spent on the implementation, and maybe more class time on SQL and some PHP because it did feel like we were expected to do a lot of that on our own.

Hayden Spinos

4.1. Which parts were the most fun?

- The most fun part for me was getting an opportunity to do real work in a team setting. I also got to teach myself a lot about web development which really boosts my value as a computer scientist.

4.2. Which parts were the most challenging? How did you solve those challenges?

- The most challenging part of the project was learning how to do web development and figuring out all of the bugs in connection our database to our web app.

4.3. Which parts were the easiest?

- The easier part for me was getting along with group members and contributing meaningful ideas and concepts as well as taking in meaningful ideas and concepts. This group was stout all around and worked really hard to make everyone else's job easier.

4.4. What did you learn that you did not imagine you would have?

- I learned that I could teach myself full coding languages in only a few days through the power of youtube and a little bit of free time.

4.5. If you had to do it all over again, what would you have done differently?

- I definitely would have started the implementation part sooner if I had to do this project again. I might have gone to some workshops or talked with group members more to boost overall understanding of web development since this whole project was new material for most of us.

4.6. What is your overall experience of working in a team?

- My overall experience working with a team was a good one. We stayed true to our meeting times for the most part, set agendas for each meeting that stated what we wanted to accomplish, and usually when it mattered most, we pulled through to get work done.

4.7. Other comments and conclusions

- This project was fairly challenging and I think it will provide some useful material for a job portfolio that demonstrates my ability as a programmer and team member. I do wish however that there could have been more instruction on the web development portion of the project since it took up a large portion of the workload.

5. Tasks for future work to expand or improve your project.

- We spent a lot of time originally making a really nice stylized html website however because of time constraints we weren't able to connect the SQL and PHP to the html so the website didn't have functionality. We think that it would be really nice to have both those working. You can see the pictures of the html stylized website earlier in the document in the "Our Experience During the Team Project" section.
- We also had to scrap an idea to have 3 different views and a login function. We would add that in the future to improve the project.

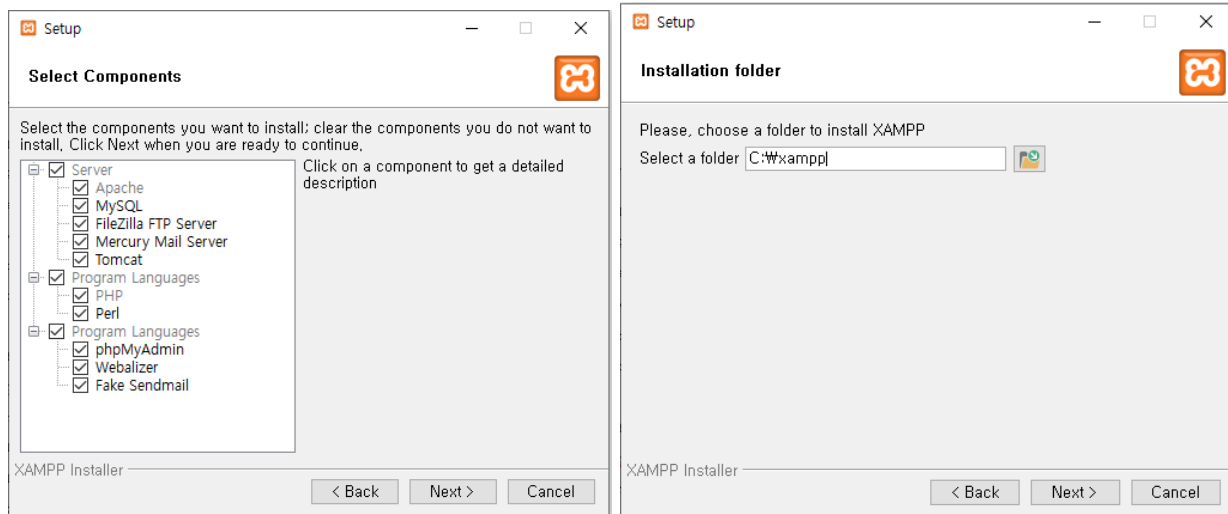
- Through these new views there would be a way for customers to be able to purchase goods on the website, so we would need to add a cart feature to save items to purchase later

Installation Instructions

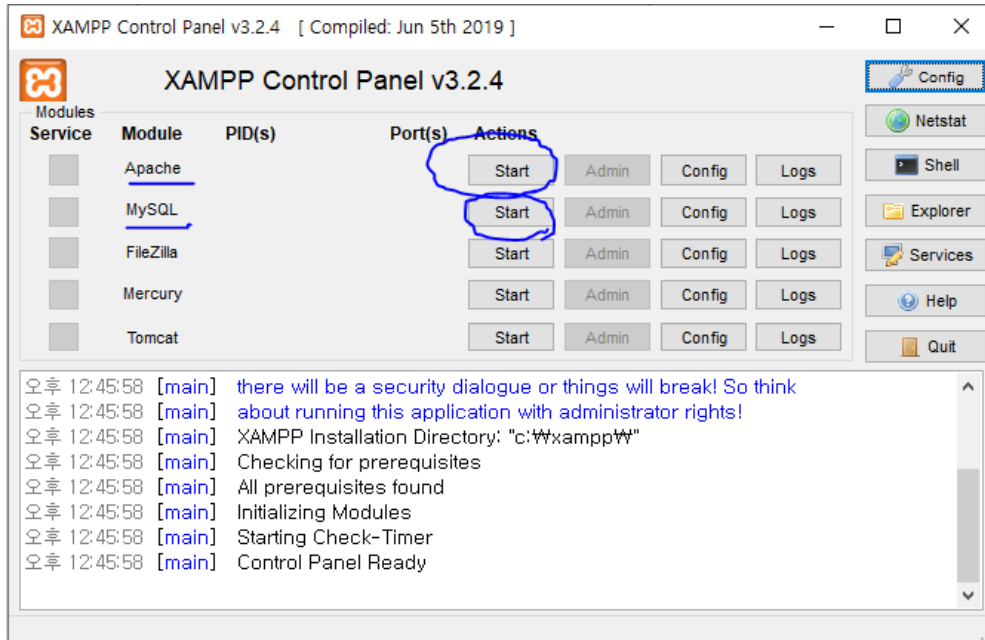
1. Install XAMPP Control Panel, the correct installer is located inside the "XAMPP Installer" folder.

- Video showing XAMPP install:

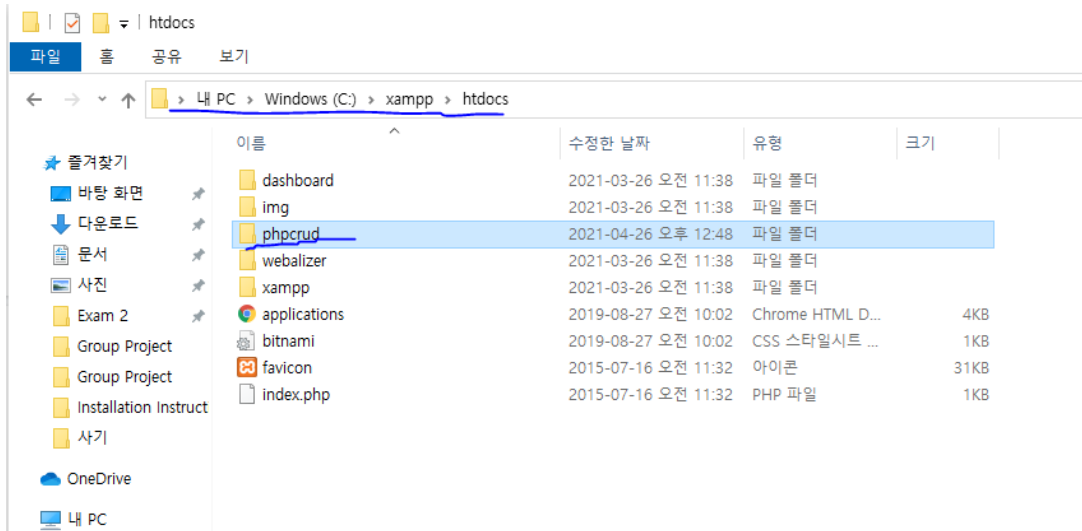
https://www.youtube.com/watch?v=-f8N4FEQWyY&ab_channel=edureka%21



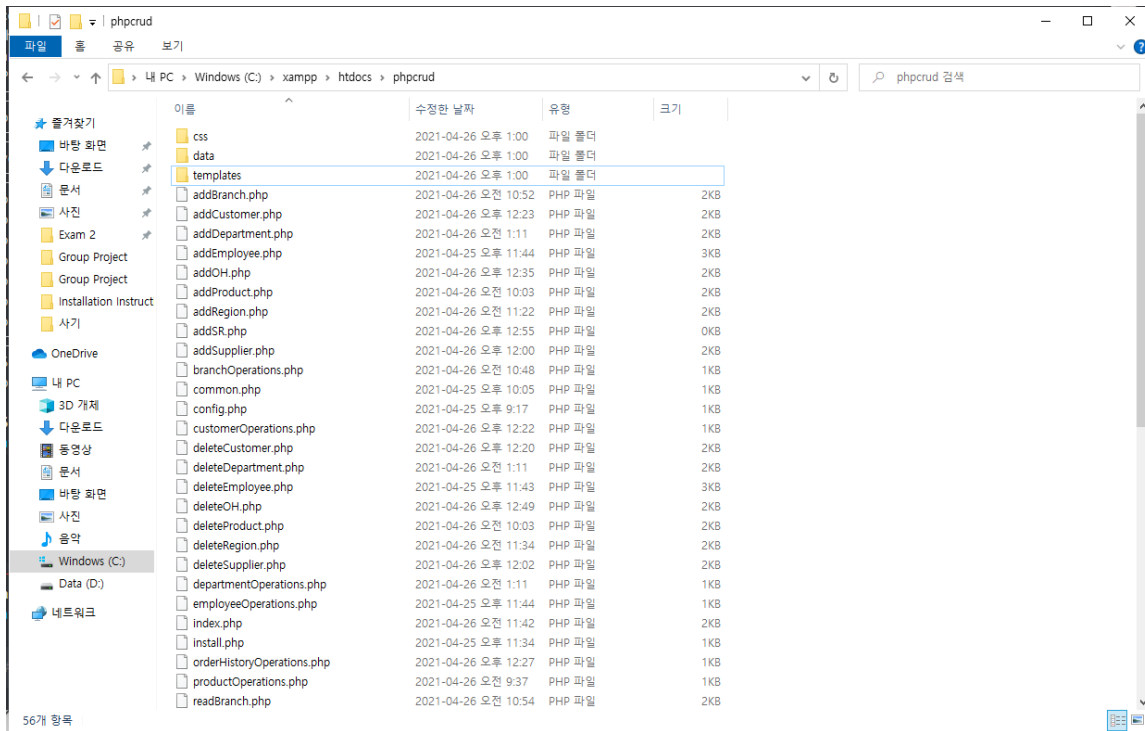
2. Once you have the XAMPP Control Panel up, start both "Apache" and "MySQL" modules.



3. In the directory where you installed XAMPP, go to `xampp/htdocs` and create a folder called "phpcrud". (Most likely path will be `C:\xampp\htdocs\phpcrud`)



4. Into this "phpcrud" folder, copy the source code files that are located in the folder named "CRUDAPP source code".



5. Install PHP if it is not already installed on your system. The download is located in the "PHP" file.

- Video showing how to install PHP on Windows 10:

https://www.youtube.com/watch?v=iW0B9NTId2g&ab_channel=EmlinCharly

PHP For Windows: Binaries and x +

windows.php.net/download/

Home | Downloads | QA Releases | Snapshots | Team | PHP.net site

Are you seeing a warning from Windows Defender? Check out [this info](#).

PHP For Windows
This site is dedicated to supporting PHP on Microsoft Windows. It also supports ports of PHP extensions or features as well as providing special builds for the various Windows architectures.

If you like to build your own PHP binaries, instructions can be found on the [Wiki](#).

PECL For Windows
[PECL extensions](#) for Windows is being worked on. Windows DLL can be downloaded right from the [PECL website](#).

The PECL extension [release](#) and [snapshot](#) build directories are browsable directly.

Which version do I choose?

IIS
If you are using PHP as FastCGI with IIS you should use the Non-Thread Safe (NTS) versions of PHP.

Apache
Please use the Apache builds provided by [Apache Lounge](#). They provide VC15 and VS16 builds of Apache for x86 and x64. We use their binaries to build the Apache SAPIs.

PHP 8.0 (8.0.3)

[Download source code](#) [23.97MB]
[Download tests package \(.phpt\)](#) [13.3MB]

VS16 x64 Non Thread Safe (2021-Mar-03 00:15:26)

- [Zip](#) [25.33MB]
sha256: be8d29c31d719dafd2730a63610a5b385ad6cc122160f585eb266713f0a5d018
- [Debug Pack](#) [24.49MB]
sha256: 58cc855f6e1447d15967ef04eb82d830b2178f3ef0dae83241b3af902f1d09b0
- [Development package \(SDK to develop PHP extensions\)](#) [1.16MB]
sha256: b01c0bf9b7306c9f244af2358a3f389a0c321427c4f3cd2a169932533f0ed874

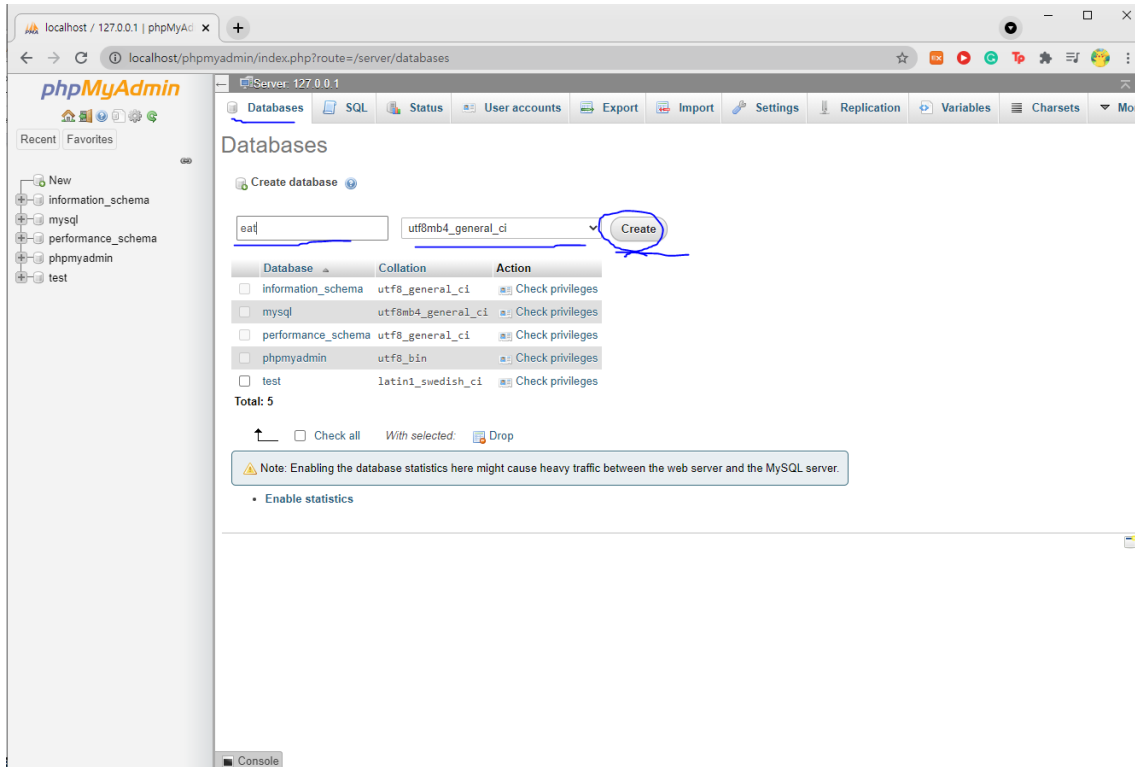
VS16 x64 Thread Safe (2021-Mar-03 00:21:37)

- [Zip](#) [25.44MB]
sha256: 04f64347cbf143f9ae2ede48e3b804b13b713751da6a9d1edd53c45a2de103fd
- [Debug Pack](#) [24.31MB]
sha256: 598868249ceca85d9fe999db13fcc98cfc57a7240eaa4d61d612b87e94c225ca
- [Development package \(SDK to develop PHP extensions\)](#) [1.16MB]
sha256: 3a21287369f1aabf9653e6ebff71dd7800dd2f257fa87f4091902129a20bebf2

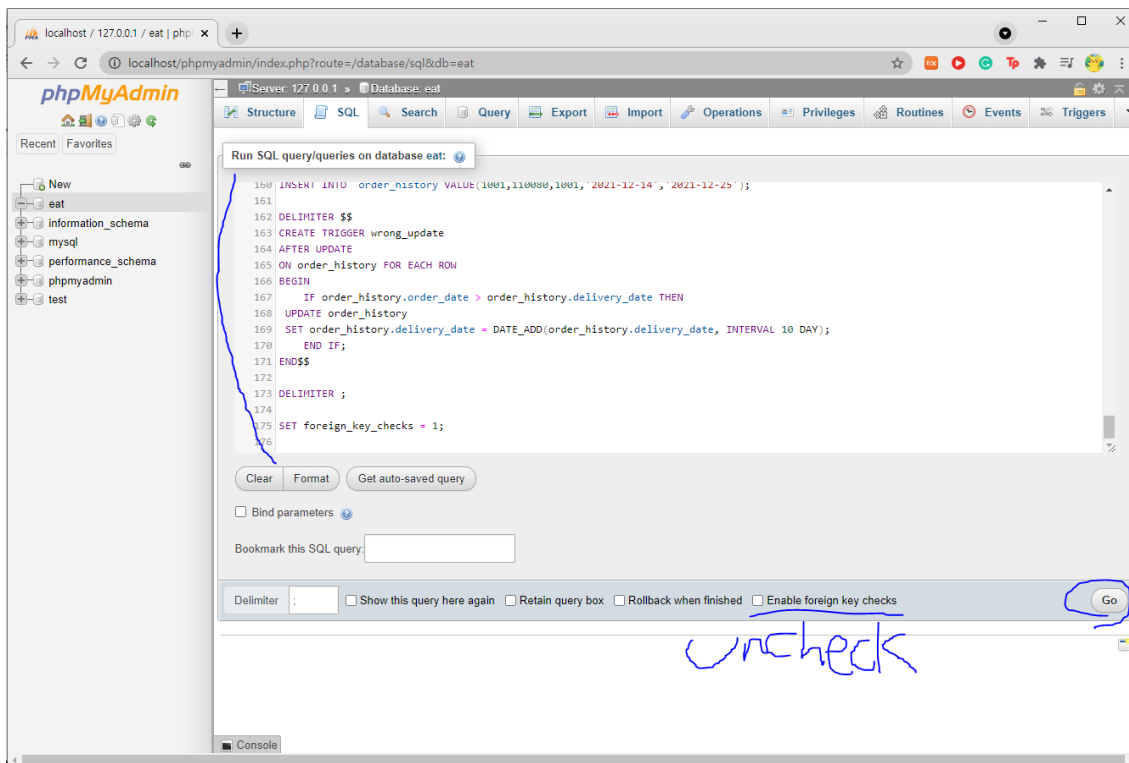
VS16 x86 Non Thread Safe (2021-Mar-03 00:17:29)

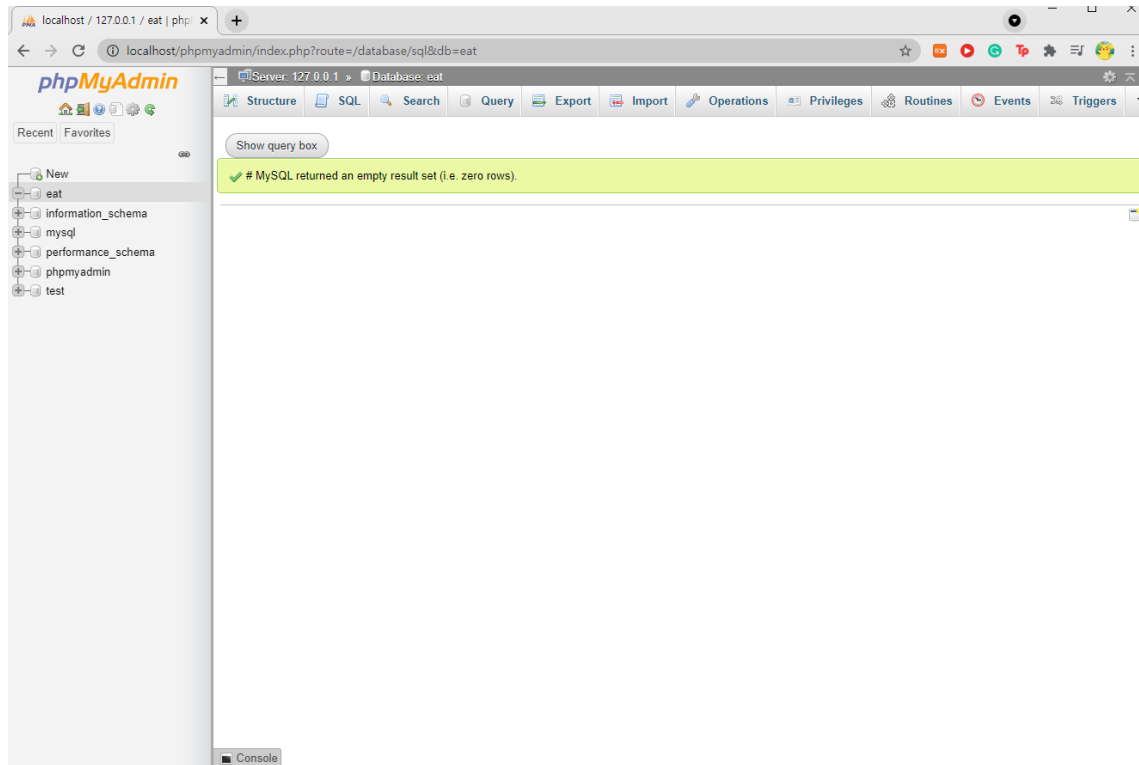
- [Zip](#) [23.36MB]
sha256: f66ba2b11a7f86f03c5eb4ff31504aa72722bf7cf7449b4d3e521caf373edc28

6. Open a web browser and navigate to "http://localhost/phpmyadmin/" > click "Databases" at the top > Under "Create Database" type in "eat" as the Database name and keep "utf8_general_ci" as the collation > click "Create".



7. Select the "eat" Database that you just created, click the "SQL" tab and input the following SQL: CHECK the enable foreign keys checkbox





-----SQL-----

```
DROP TABLE order_history;

DROP TABLE supplier_shipments;

DROP TABLE sales_relations;

DROP TABLE products;

DROP TABLE supplier;

DROP TABLE customer;

DROP TABLE region;

DROP TABLE branch;

DROP TABLE department;

DROP TABLE employee;


CREATE TABLE IF NOT EXISTS `employee` (
  `employee_id` INT(11)NOT NULL,
  `first_name` VARCHAR(15)NOT NULL,
```

```
`last_name` VARCHAR(20) NOT NULL,  
`birth_date` DATETIME,  
`sex` varchar(1) NOT NULL,  
`salary` INT NOT NULL,  
`super_id` INT NOT NULL,  
`department_id` INT NULL,  
`branch_id` INT NOT NULL,  
PRIMARY KEY(`employee_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `department`(  
`department_id` INT NOT NULL,  
`department_name` VARCHAR(40) NOT NULL,  
`department_head_id` INT NOT NULL,  
PRIMARY KEY(`department_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `branch`(  
`branch_id` INT NOT NULL,  
`branch_name` VARCHAR(20) NOT NULL,  
`branch_head_id` INT(11) NOT NULL,  
`region_id` INT NOT NULL,  
PRIMARY KEY (`branch_id`)) ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `region`(  

```



```
`region_id` INT NOT NULL,  
`region_name` VARCHAR(10) NOT NULL,  
`region_sales` INT NOT NULL,  
PRIMARY KEY (`region_id`)) ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `customer` (  
`customer_id` INT NOT NULL,  
`first_name` VARCHAR(15) NOT NULL,  
`last_name` VARCHAR(20) NOT NULL,  
`customer_address` VARCHAR(40),  
`contact_information` VARCHAR(40),  
PRIMARY KEY (`customer_id`)) ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `supplier` (  
`supplier_id` INT NOT NULL,  
`company_name` VARCHAR(20) NOT NULL,  
`contact_information` VARCHAR(40),  
`address` VARCHAR(40) NOT NULL,  
PRIMARY KEY (`supplier_id`)) ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `products` (  
`product_id` INT,  
`product_name` VARCHAR(20) NOT NULL,
```

```
`price_per_unit` double NOT NULL,  
`Inventory` INT NOT NULL,  
PRIMARY KEY (`product_id`)) ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `sales_relations`(  
  `branch_id` INT NOT NULL,  
  `customer_id` INT NOT NULL,  
  `total_sales` double,  
  PRIMARY KEY (`branch_id`,`customer_id`))  
ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `supplier_shipments`(  
  `branch_id` INT NOT NULL,  
  `supplier_id` INT NOT NULL,  
  `shipment_id` INT NOT NULL,  
  `product_id` INT NOT NULL,  
  `shipment_date` DATETIME NOT NULL,  
  PRIMARY KEY (`branch_id`,`supplier_id`))ENGINE=InnoDB  
DEFAULT CHARSET=utf8;
```

```
CREATE TABLE IF NOT EXISTS `order_history`(  
  `customer_id` INT NOT NULL,  
  `order_id` INT NOT NULL,  
  `product_id` INT NOT NULL,
```

```
`order_date` DATETIME,  
`delivery_date` DATETIME,  
PRIMARY KEY (`customer_id`))ENGINE = InnoDB  
DEFAULT CHARSET=utf8;
```

```
ALTER TABLE employee  
ADD FOREIGN KEY(department_id)  
REFERENCES department(department_id),  
ADD FOREIGN KEY(super_id)  
REFERENCES employee(employee_id),  
ADD FOREIGN KEY(branch_id)  
REFERENCES branch(branch_id);
```

```
ALTER TABLE branch  
ADD FOREIGN KEY(region_id)  
REFERENCES region(region_id),  
ADD FOREIGN KEY(branch_head_id)  
REFERENCES employee(employee_id);
```

```
TRUNCATE table `employee`;  
TRUNCATE table `department`;  
TRUNCATE table `branch`;  
TRUNCATE table `region`;  
TRUNCATE table `customer`;  
TRUNCATE table `supplier`;  
TRUNCATE table `products`;  
TRUNCATE table `sales_relations`;
```

```
TRUNCATE table `supplier_shipments`;
```

```
TRUNCATE table `order_history`;
```

```
SET foreign_key_checks = 0;
```

```
INSERT
```

```
INTO `employee` VALUES (1001, "John", "Doe", '1968-04-27', "M", 410000, 9000, 0002, 10);
```

```
INSERT INTO `employee` VALUE (1002, "James", "Smith", '1984-07-22', "M", 120000, 1001, 2221, 12);
```

```
INSERT
```

```
INTO `employee` VALUE (1003, "Sally", "Johnson", '1992-02-15', "F", 52000, 1002, 2222, 12);
```

```
INSERT INTO `department` VALUE (0002, "Corporate", 1001);
```

```
INSERT INTO `department` VALUE (1221, "Human Resources", 1499);
```

```
INSERT INTO `department` VALUE (1222, "Sales", 1599);
```

```
INSERT INTO `department` VALUE (2221, "Human Resources", 2499);
```

```
INSERT INTO `department` VALUE (2222, "Sales", 2599);
```

```
INSERT INTO `branch` VALUE (10, "Dayton", 1001, 1400);
```

```
INSERT INTO `branch` VALUE (20, "Scranton", 2001, 2400);
```

```
INSERT INTO `region` VALUE (1, "swus", 880888.00);
```

```
INSERT INTO `customer` VALUE (1001, "Jane", "Doe", "1002 Neverland Rd", "nsdad@yahoo.com");
```

```
INSERT INTO`customer`VALUE(2002,"Chris","Smith","2463 Dunken
Rd","ChrisS547@hotmail.com");
```

```
INSERT INTO`supplier`VALUE(1001,"Adidas","supplier@adidasl.com",
"1965 Eagle Court");
```

```
INSERT INTO`supplier`VALUE (2002,"Nike","suuplies@nike.com","123 Office
Row");
```

```
INSERT INTO`products`VALUE(12401,"Adidas Hat",15,1500);
```

```
INSERT INTO`products`VALUE(14302,"Nike Shirt",35,750);
```

```
INSERT INTO`products`VALUE(15402,"Adidas Socks",35,750);
```

```
INSERT INTO`sales_relations`VALUE(1123,1001,75);
```

```
INSERT INTO`sales_relations`VALUE(1124,1001,150);
```

```
INSERT INTO`supplier_shipments`VALUE(1001,1001,100808,1001, '2021-12-14');
```

```
INSERT INTO
```

```
`order_history`VALUE(1001,110080,1001,'2021-12-14','2021-12-25');
```

```
DELIMITER $$
```

```
CREATE TRIGGER wrong_update
```

```
AFTER UPDATE
```

```
ON order_history FOR EACH ROW
```

```
BEGIN
```

```
    IF order_history.order_date > order_history.delivery_date THEN
```

```

UPDATE order_history

SET order_history.delivery_date = DATE_ADD(order_history.delivery_date,
INTERVAL 10 DAY);

    END IF;

END$$

```

```

DELIMITER ;

```

```

SET foreign_key_checks = 1;

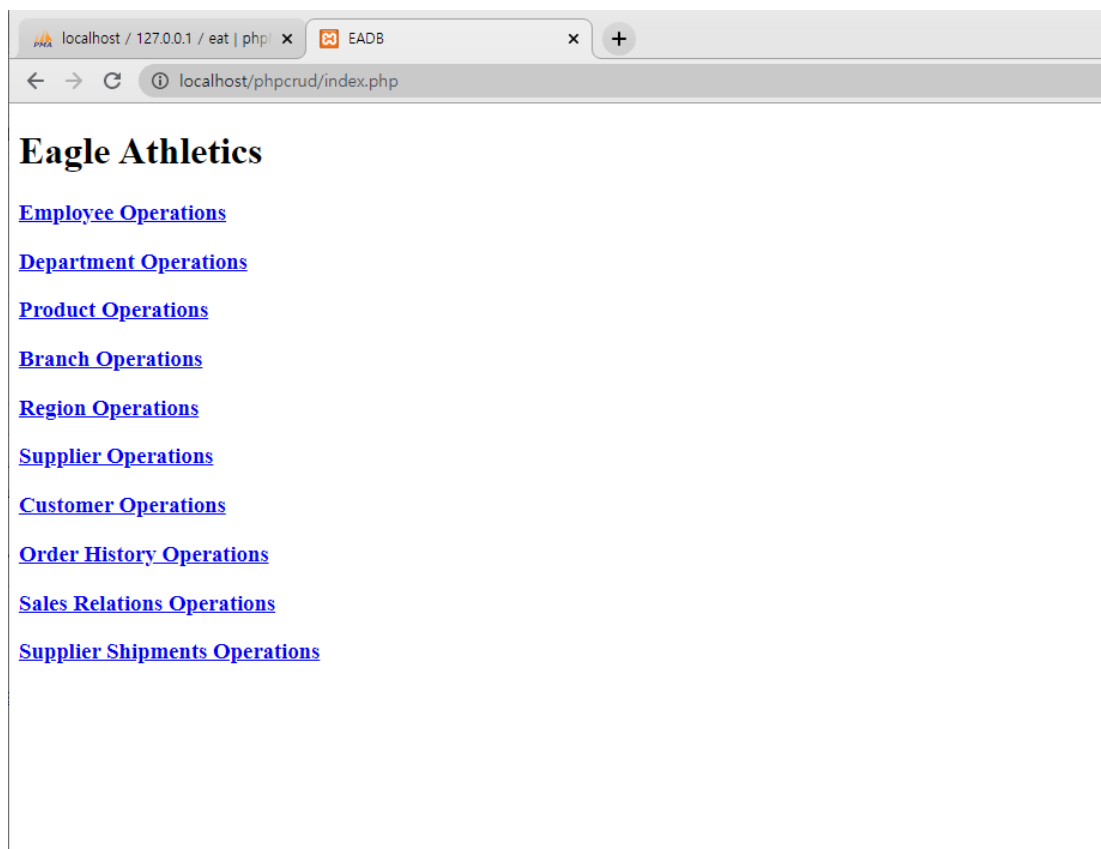
```

```

-----SQL-----

```

8. Now that the Database is populated with some sample data, go back into your web browser and navigate to "<http://localhost/phpcrud/index.php>".



9. You should see the frontend of the project, and it allows you to easily see, create, edit, and delete entries and their data from the database that was just created.

- Feel free to navigate between the phpmyadmin page that displays the database and the read.php page to see the changes you make reflected in the database and vice versa in real time.

GitHub Repository/Source Code

<https://github.com/NateAgcaoili/Eagle-Athletics-Database-Visualizer>