NYU TANDON SCHOOL OF ENGINEERING

CS-GY 6083 - INET, 2022 Principles of Database Systems

PROJECT PART 2

TEAM

Name: Maohua Shen	NET ID: ms13468
Name: Melody Zhang	NET ID: yz8608
Name: Haoyu Wang	NET ID: hw3256

Submission Date: 08/25/2022

II.Table of Contents

3. Execute summary

This information system made for We Do Secure (WDS), one of the largest life insurance provider company in the USA, for its Auto and Home insurance services. The information system includes customer information, policy information, and asset(home) information, driver(for auto) information, and financial management information

For customer information, this information system includse customer's full name, address, gender, marital status, and customer type. All the information is mandatory information except gender.

For insurance policy information , WDS has two catergory, Auto insurance policy and and Home insurance policy, which are stored separately, the information of auto policy includes the policy start date and end date, insturance amout, insurance status, VIN , driver id and customer id. The information of the home policy includes the policy start date and end date, insturance amount , insurance status,home id and customer id. All previously expired policy data will in the same table. The custmer id will be the forgen key.

For driver information related auto insurance, the system will includes driver lisence number,driver name, vin number, and customer id, the customer id will be the forgen key.

For Asset WDS intend to store home purchase date, home purchase value, home area in Sq. Ft., Type of home (one of the value as, S,M,C,T representing Single family, Multi Family, Condominium, Town house respectively). In order to decide appropriate home insurance premium, WDS intending to store four parameters namely: Auto Fire notification, Home Security System, Swimming Pool, and Basement.

For finance management, there are invoice and payment information collection for the auto and home policy separaterly. The invoice information including invoice date, payment due day, relevant customer id and vehicle or home. The payment information including pay date ,pay method ,pay account number, relevant customer id and vehicle or home.

Employees and customers have different authorization on data access. Admin employees have authority on every table. Other employees can only read employee table, and cannot create, update or delete an employee, and other employees have full authorizations on other tables. In other word, admin employee can manage other employees. Customers can only access data corresponding to this customer. A customer can only read policy and invoice table. A customer can only check the policy and invoice he or she has. If a customer want to buy a new policy or alter an existing policy, he or she must contact one of the employees, and let them to alter the table for the customer. A customer have full authorizations on home, vehicle and driver table. A customer can only create and read payment records. Similarly, if a customer wants to update or delete a payment, he or she must contact one of the employees, only employees have the authority to update and delete a payment record.

4. Brief details of software, programming language, and database used

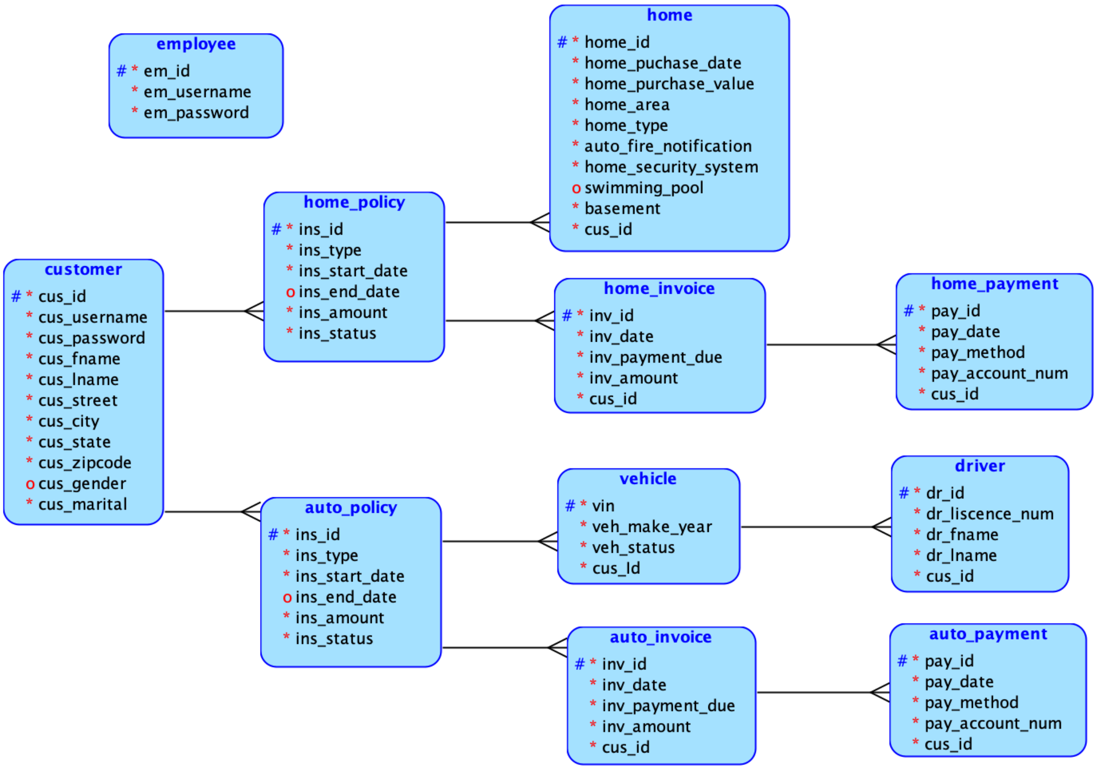
We use PHP to build our frontend webpage and connect it with our backend database. To make the webpage functional and attractive to users, we have different PHP files in the project directory. For example, “index.php” is made to set up the layout and background of our webpage, and “connection.php” to connect our frontend webpage with backend database, so when users enter their information on our webpage, the data is automatically stored in our database and the user's password is encrypted.

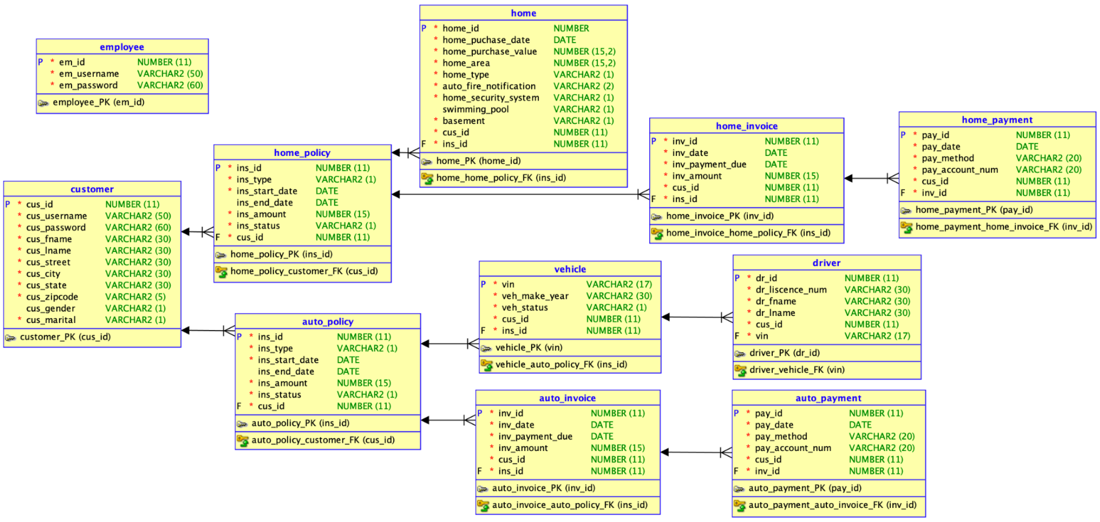
Our database is built in phpMyAdmin using MySQL script. With phpMyAdmin we can administrate our database conveniently. It allows us to add or delete tables, columns or rows. In addition, to ensure the security, we use it to create a new user account for our database with custom password and username.

Finally, we use XAMPP to run our webpage on a local web server. Every time we open our webpage, we start Apache and MySQL in the XAMPP control panel to create a local server and use it to connect with the backend database.

5. Logical and relational data model, and assumptions if any

In order to better implement and fit the design idea of the front end, our model is modified from the project part 1, as shown below.





6. List of tables, and total number of records of each table

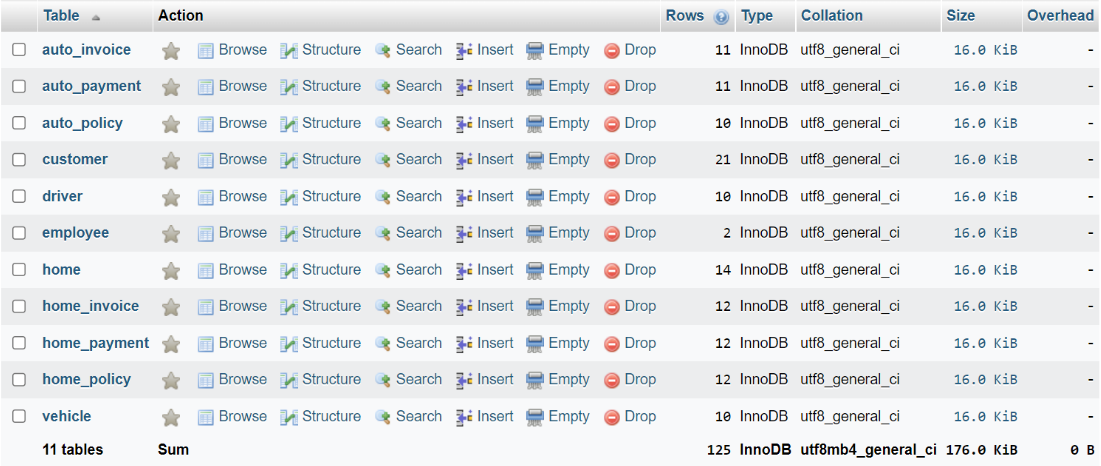
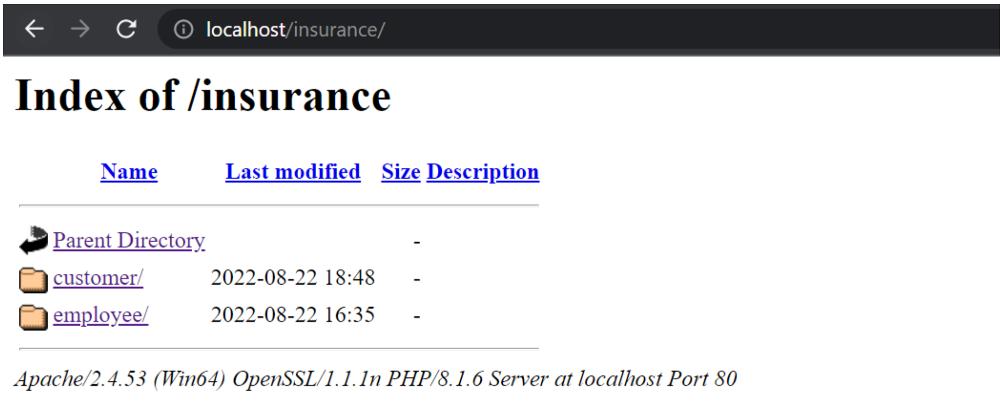


	Table	Action	Rows	Type	Collation	Size	Overhead
□	auto_invoice	☆ Browse Structure Search Insert Empty Drop	11	InnoDB	utf8_general_ci	16.0 KiB	-
□	auto_payment	☆ Browse Structure Search Insert Empty Drop	11	InnoDB	utf8_general_ci	16.0 KiB	-
□	auto_policy	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8_general_ci	16.0 KiB	-
□	customer	☆ Browse Structure Search Insert Empty Drop	21	InnoDB	utf8_general_ci	16.0 KiB	-
□	driver	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8_general_ci	16.0 KiB	-
□	employee	☆ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8_general_ci	16.0 KiB	-
□	home	☆ Browse Structure Search Insert Empty Drop	14	InnoDB	utf8_general_ci	16.0 KiB	-
□	home_invoice	☆ Browse Structure Search Insert Empty Drop	12	InnoDB	utf8_general_ci	16.0 KiB	-
□	home_payment	☆ Browse Structure Search Insert Empty Drop	12	InnoDB	utf8_general_ci	16.0 KiB	-
□	home_policy	☆ Browse Structure Search Insert Empty Drop	12	InnoDB	utf8_general_ci	16.0 KiB	-
□	vehicle	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8_general_ci	16.0 KiB	-
	11 tables	**Sum**	**125**	**InnoDB**	**utf8mb4_general_ci**	**176.0 KiB**	**0 B**

7. Screenshots of some sessions, pages, menus of your Web Application

Our main page allows users to choose to log in as customers or employees.

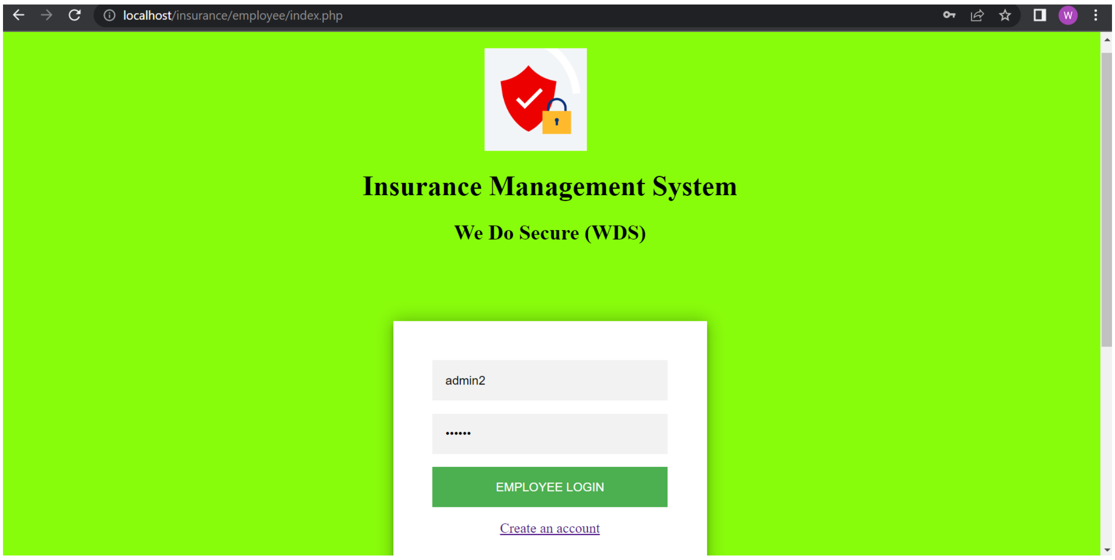


Index of /insurance

Name	Last modified	Size	Description
Parent Directory		-	
customer/	2022-08-22 18:48	-	
employee/	2022-08-22 16:35	-	

Apache/2.4.53 (Win64) OpenSSL/1.1.1n PHP/8.1.6 Server at localhost Port 80

Below is employee login page.



localhost/insurance/employee/index.php

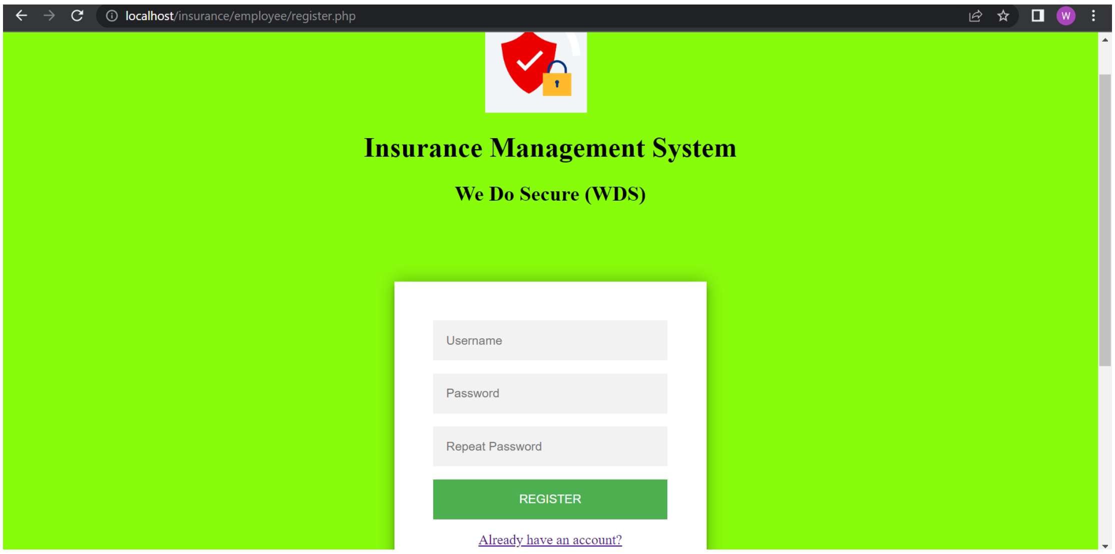
Insurance Management System

We Do Secure (WDS)

admin2

••••••

If this is the employee's first time logging into the system, the employee can create an account at here.

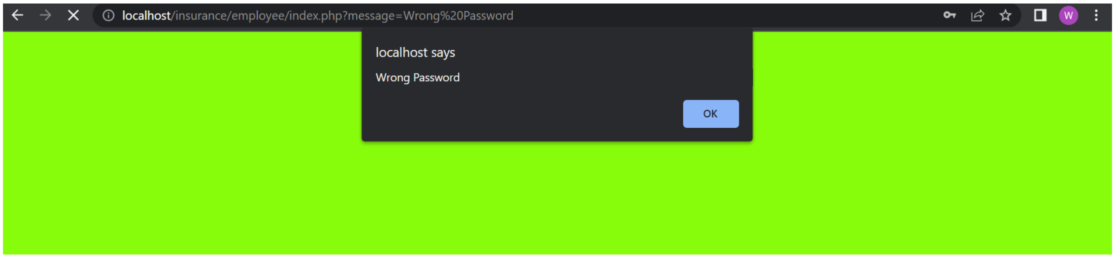


localhost/insurance/employee/register.php

Insurance Management System

We Do Secure (WDS)

If the employees enter the wrong password, the following box will pop up, and click OK to navigate back to the login page.

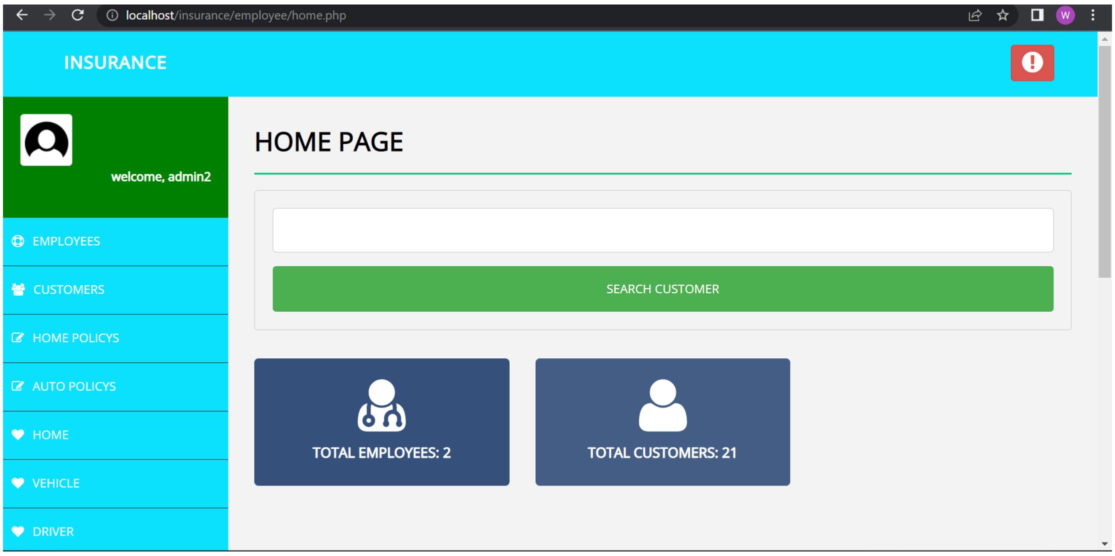


localhost/insurance/employee/index.php?message=Wrong%20Password

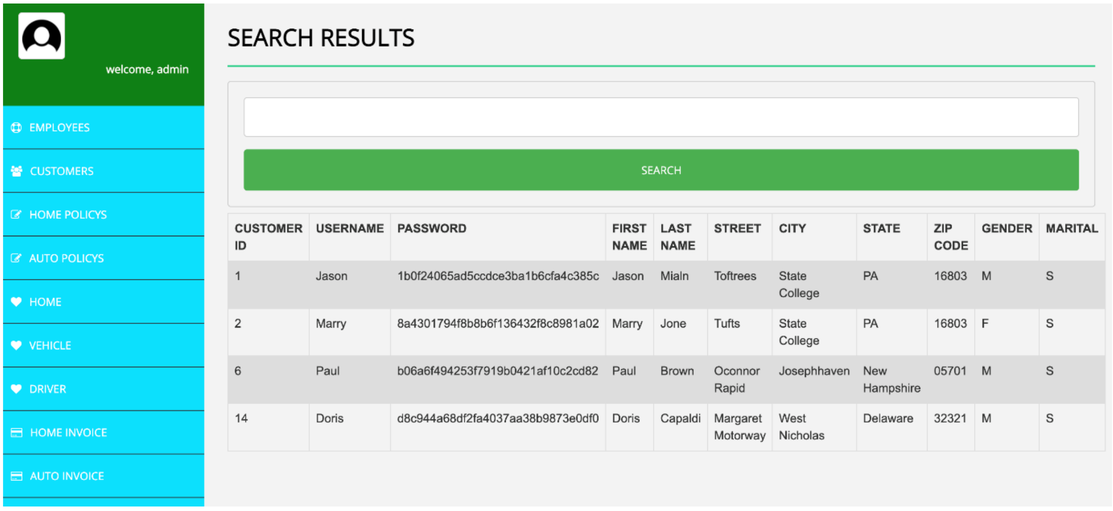
localhost says

Wrong Password

When the employee log in, the main page looks like this. The search customer button will return all customer records if you leave it as blank.

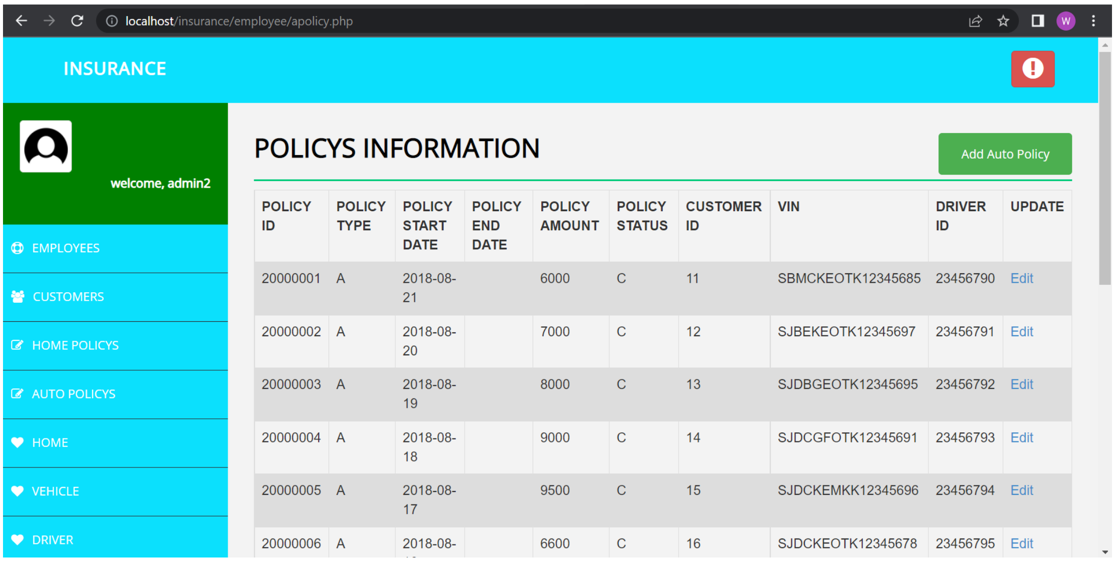


Users can also enter information in the search bar and the system will do fuzzy query and returns any records which contains that information. For example, if user enters “pa” in the search bar, the system will return the four rows as shown below, since the first two rows contain “PA” in the “state” column, and the third and fourth row contains “pa” in username and last name respectively.



CUSTOMER ID	USERNAME	PASSWORD	FIRST NAME	LAST NAME	STREET	CITY	STATE	ZIP CODE	GENDER	MARITAL
1	Jason	1b0f24065ad5ccdce3ba1b6cfa4c385c	Jason	Mialn	Toftrees	State College	PA	16803	M	S
2	Marry	8a4301794f8b8b6f136432f8c8981a02	Marry	Jone	Tufts	State College	PA	16803	F	S
6	Paul	b06a6f494253f7919b0421af10c2cd82	Paul	Brown	Oconnor Rapid	Josephhaven	New Hampshire	05701	M	S
14	Doris	d8c944a68df2fa4037aa38b9873e0df0	Doris	Capaldi	Margaret Motorway	West Nicholas	Delaware	32321	M	S

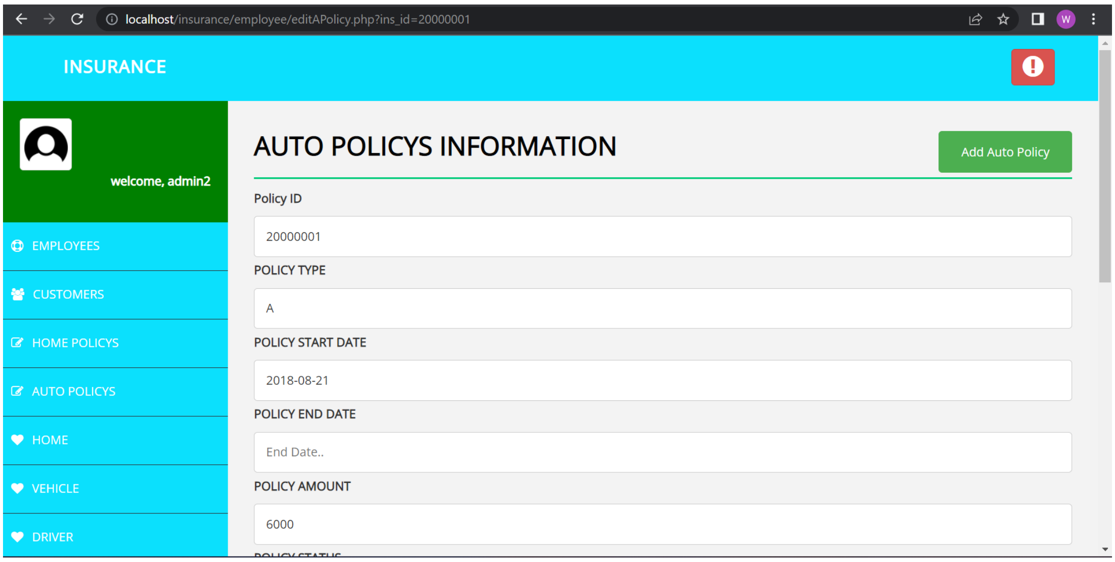
Clicking the auto policies tab on the left, the employee can go to the auto policies information page, and the employee can add, edit or delete policy information.



POLICYS INFORMATION

POLICY ID	POLICY TYPE	POLICY START DATE	POLICY END DATE	POLICY AMOUNT	POLICY STATUS	CUSTOMER ID	VIN	DRIVER ID	UPDATE
20000001	A	2018-08-21		6000	C	11	SBMCKEOTK12345685	23456790	Edit
20000002	A	2018-08-20		7000	C	12	SJBEKEOTK12345697	23456791	Edit
20000003	A	2018-08-19		8000	C	13	SJDBGEOTK12345695	23456792	Edit
20000004	A	2018-08-18		9000	C	14	SJDCGFOTK12345691	23456793	Edit
20000005	A	2018-08-17		9500	C	15	SJDCKEMKK12345696	23456794	Edit
20000006	A	2018-08-		6600	C	16	SJDCKEOTK12345678	23456795	Edit

Here are the pages for employees to edit, delete and add a specific auto policy's information.



AUTO POLICYS INFORMATION

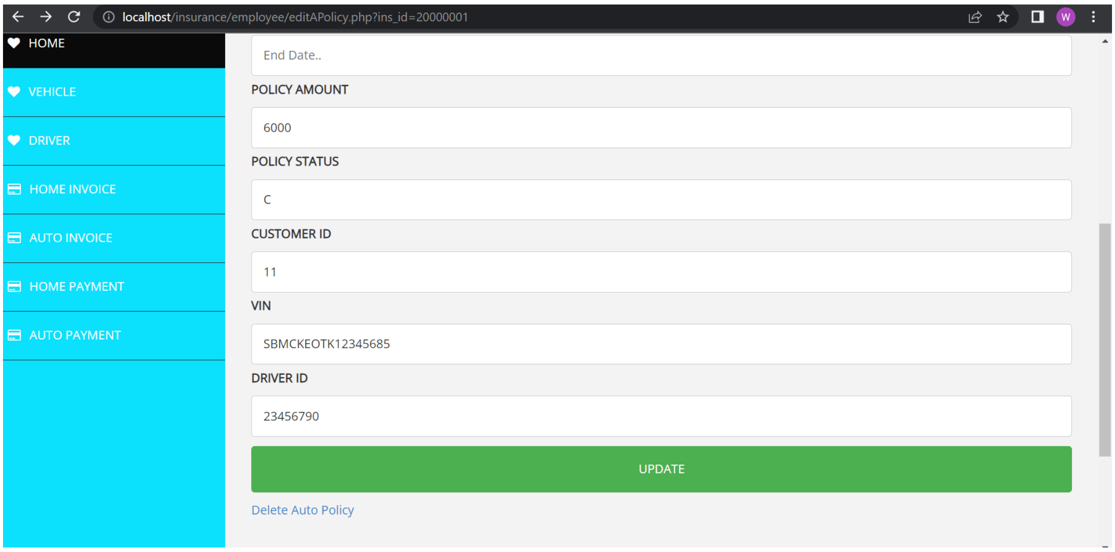
Policy ID
20000001

POLICY TYPE
A

POLICY START DATE
2018-08-21

POLICY END DATE
End Date..

POLICY AMOUNT
6000



End Date..

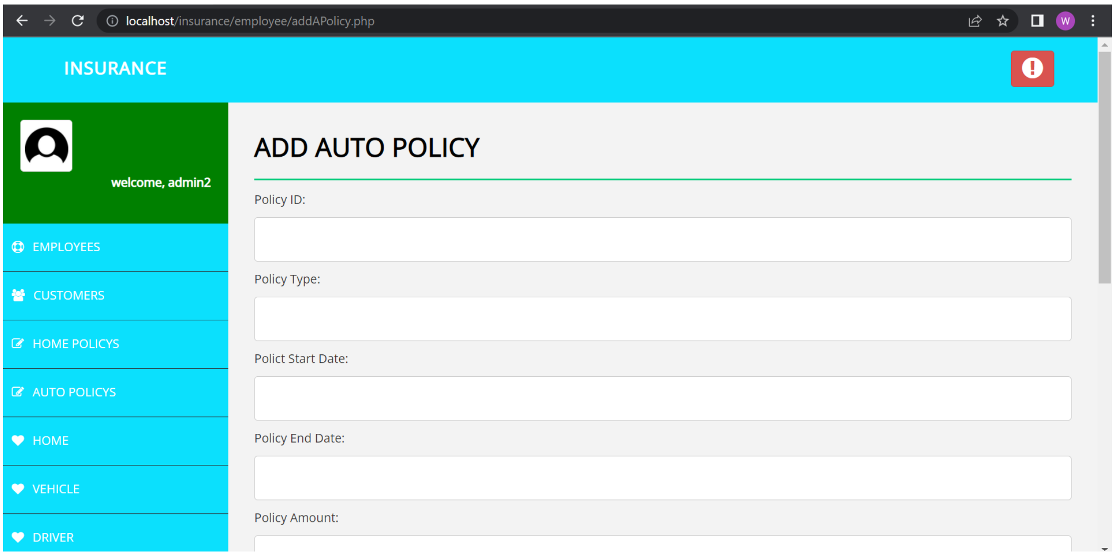
POLICY AMOUNT
6000

POLICY STATUS
C

CUSTOMER ID
11

VIN
SBMCKEOTK12345685

DRIVER ID
23456790



ADD AUTO POLICY

Policy ID:

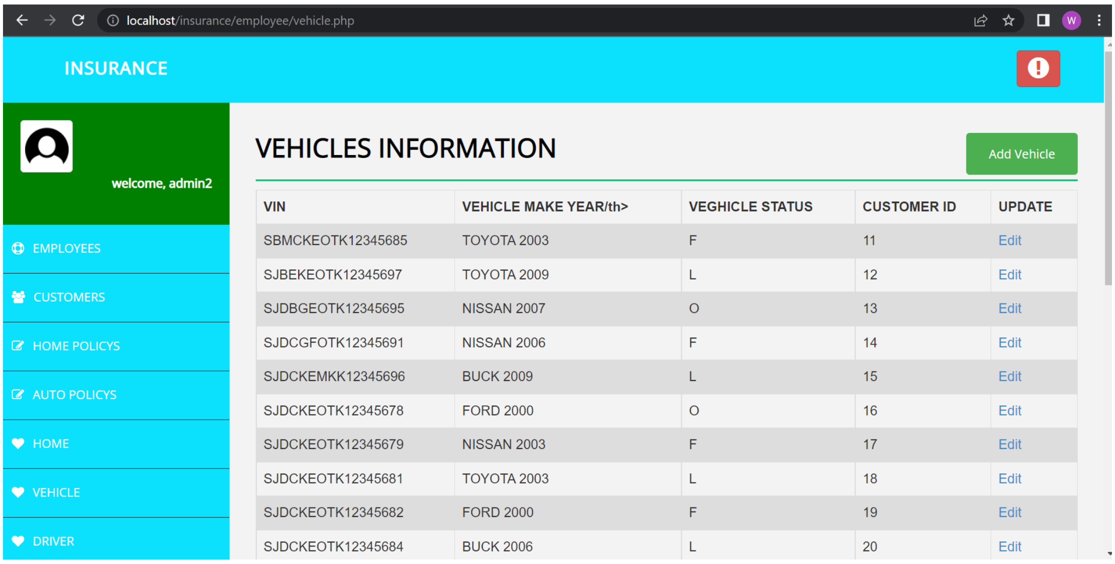
Policy Type:

Polict Start Date:

Policy End Date:

Policy Amount:

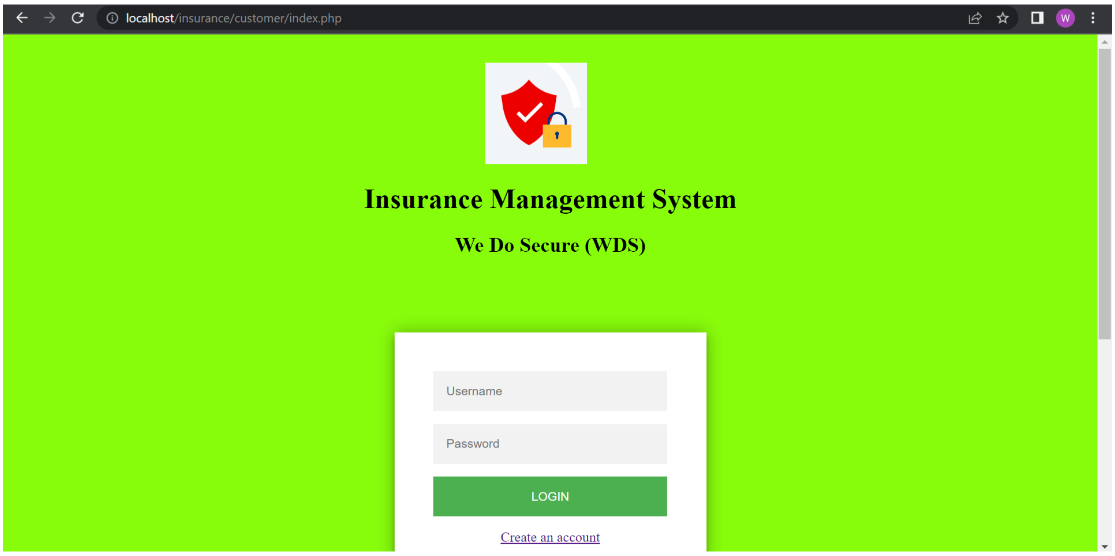
Clicking the vehicle tab on the left, the employee can go to the customers' vehicle information page, and the employee can add, edit or delete vehicle information. The employee can click the red button at right up corner to log out the session.



VEHICLES INFORMATION

VIN	VEHICLE MAKE YEAR/th>	VEGHICLE STATUS	CUSTOMER ID	UPDATE
SBMCKEOTK12345685	TOYOTA 2003	F	11	Edit
SJBEKEOTK12345697	TOYOTA 2009	L	12	Edit
SJDBGEOTK12345695	NISSAN 2007	O	13	Edit
SJDCGFOTK12345691	NISSAN 2006	F	14	Edit
SJDCKEMKK12345696	BUCK 2009	L	15	Edit
SJDCKEOTK12345678	FORD 2000	O	16	Edit
SJDCKEOTK12345679	NISSAN 2003	F	17	Edit
SJDCKEOTK12345681	TOYOTA 2003	L	18	Edit
SJDCKEOTK12345682	FORD 2000	F	19	Edit
SJDCKEOTK12345684	BUCK 2006	L	20	Edit

Below is the page for customer login.



localhost/insurance/customer/index.php

Insurance Management System

We Do Secure (WDS)

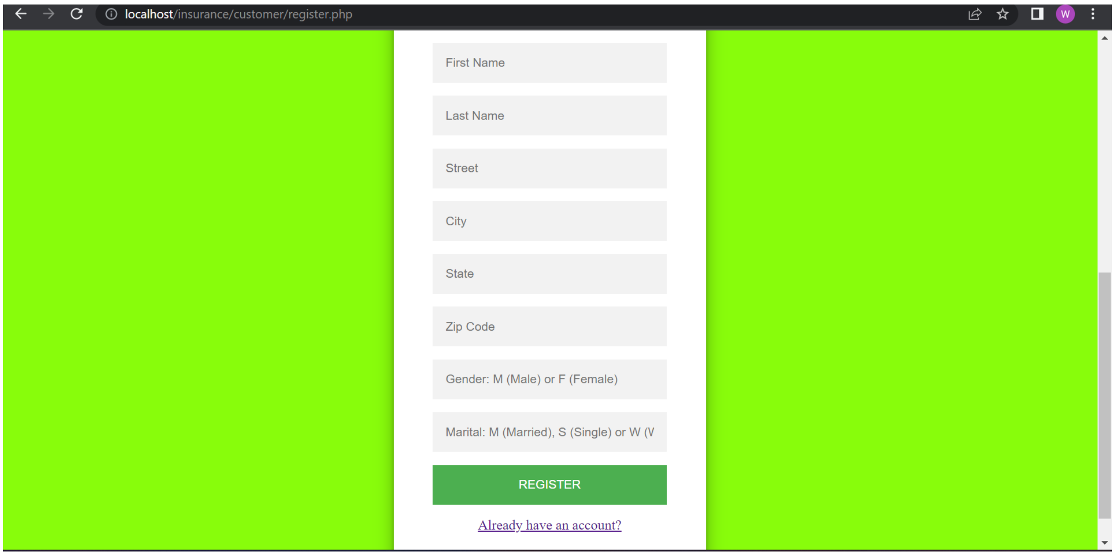
Username

Password

LOGIN

Create an account

If this is the customer's first time logging into the system, here is the page to create an account.



localhost/insurance/customer/register.php

First Name

Last Name

Street

City

State

Zip Code

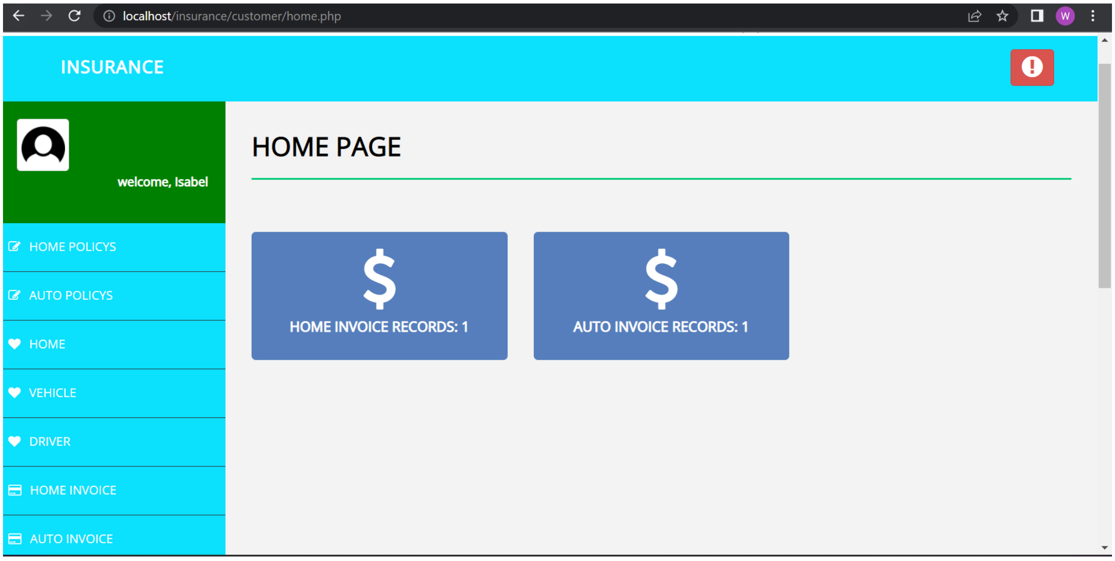
Gender: M (Male) or F (Female)

Marital: M (Married), S (Single) or W (V

REGISTER

Already have an account?

This is the home page after customers login.



localhost/insurance/customer/home.php

INSURANCE

welcome, Isabel

HOME POLICYS

AUTO POLICYS

HOME

VEHICLE

DRIVER

HOME INVOICE

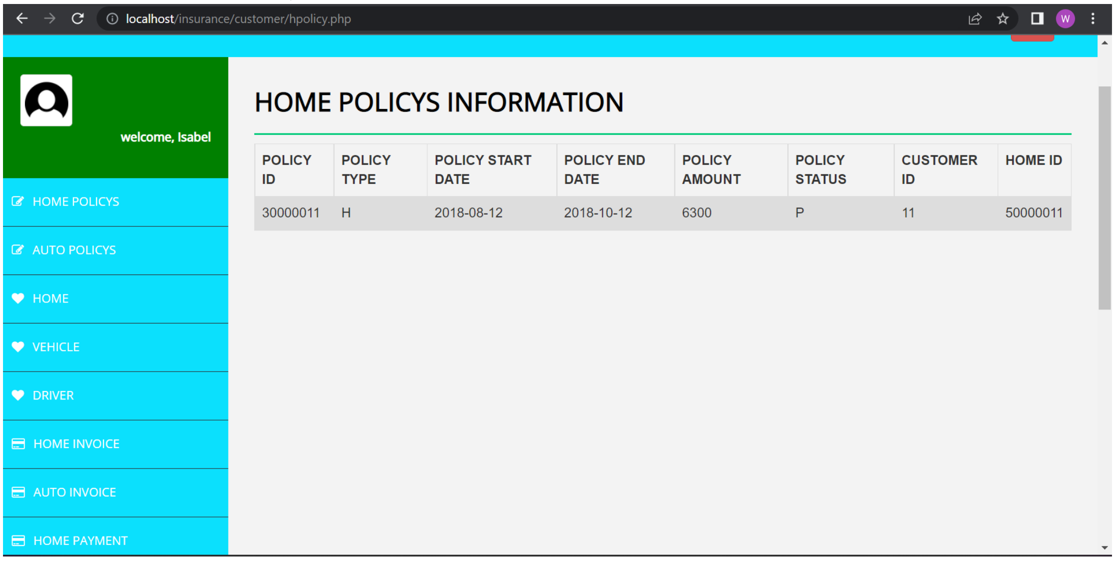
AUTO INVOICE

HOME PAGE

$ HOME INVOICE RECORDS: 1

$ AUTO INVOICE RECORDS: 1

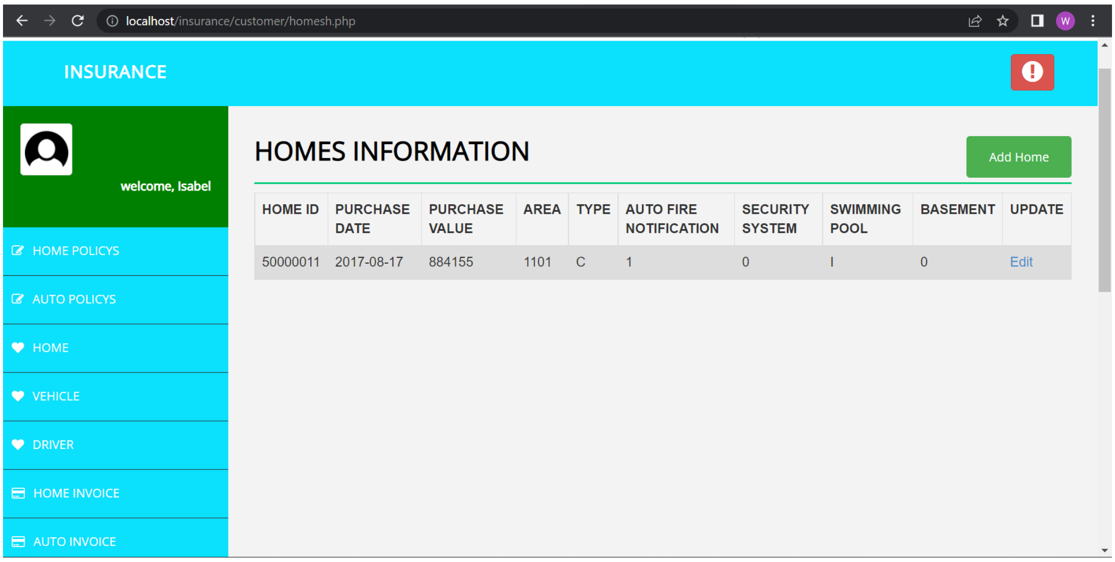
The customers can only see their own information.



HOME POLICYS INFORMATION

POLICY ID	POLICY TYPE	POLICY START DATE	POLICY END DATE	POLICY AMOUNT	POLICY STATUS	CUSTOMER ID	HOME ID
30000011	H	2018-08-12	2018-10-12	6300	P	11	50000011

Customers can add, delete and edit their home information.

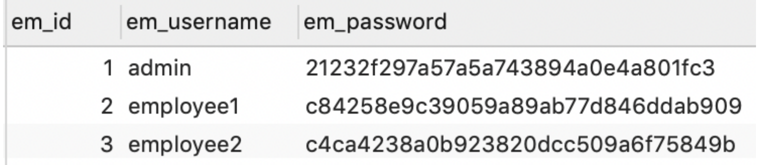


HOMES INFORMATION

HOME ID	PURCHASE DATE	PURCHASE VALUE	AREA	TYPE	AUTO FIRE NOTIFICATION	SECURITY SYSTEM	SWIMMING POOL	BASEMENT	UPDATE
50000011	2017-08-17	884155	1101	C	1	0	I	0	Edit

8.Security Features

8.1 Password is stored in database after encrypted using MD5()



em_id	em_username	em_password
1	admin	21232f297a57a5a743894a0e4a801fc3
2	employee1	c84258e9c39059a89ab77d846ddab909
3	employee2	c4ca4238a0b923820dcc509a6f75849b

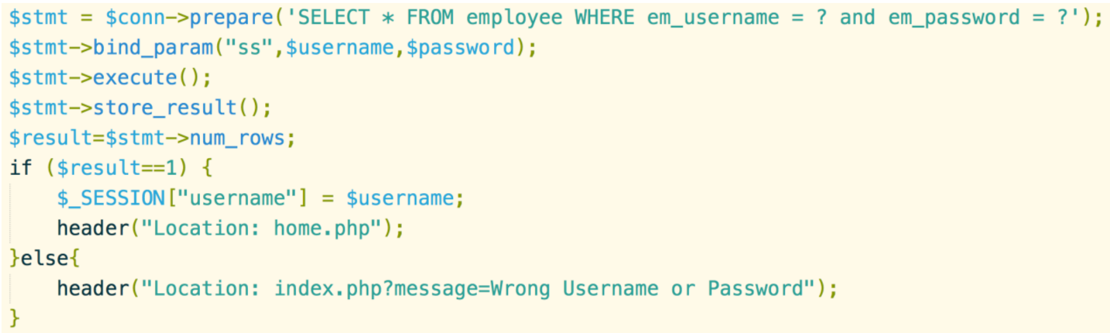
8.2 Employees and customers have different authorization on data access

Admin Employees: CURD on every table

Other Employees: Only read on employee table, CURD on every other table
Customers can only access data corresponding to this customer:
Read on policy, invoice table
CURD on home, vehicle, driver table
Read and Create on payment table

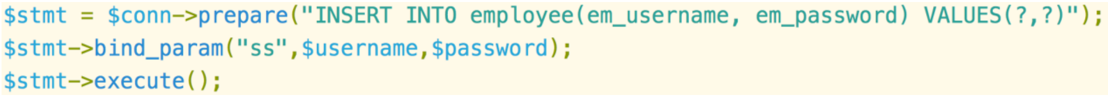
8.3 PreparedStatement to prevent SQL injection

If we use this statement:
$query = “SELECT * FROM employee WHERE username = ‘$username’ and password = ‘$password’”;
The user inserts admin as $username and a’ or ‘1’=’1 as $password, user will be able to login to the admin account without knowing the password because the SQL statement has been altered.
If we use PreparedStatement. The SQL query is pre-compiled with placeholders, and the user’s data is added later. If the user inserts admin and a’ or ‘1’=’1, the initial SQL query logic won’t be changed. Instead, the database will look for a user admin whose password is literally a’ or ‘1’=’1.
Here is an example that we implement preparedstatement in login process that prevent SQL injection.



```
$stmt = $conn->prepare('SELECT * FROM employee WHERE em_username = ? and em_password = ?');
$stmt->bind_param("ss",$username,$password);
$stmt->execute();
$stmt->store_result();
$result=$stmt->num_rows;
if ($result==1) {
    $_SESSION["username"] = $username;
    header("Location: home.php");
}else{
    header("Location: index.php?message=Wrong Username or Password");
}
```

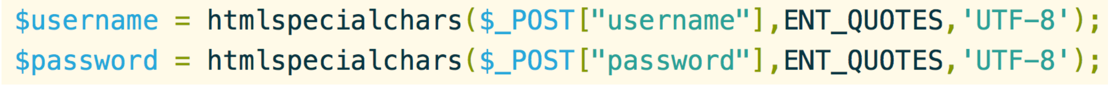
Here is another example that we implement preparedstatement in register process that prevent SQL injection.



```
$stmt = $conn->prepare("INSERT INTO employee(em_username, em_password) VALUES(?,?)");
$stmt->bind_param("ss",$username,$password);
$stmt->execute();
```

8.4 PHP’s htmlspecialchars() to prevent XSS injection

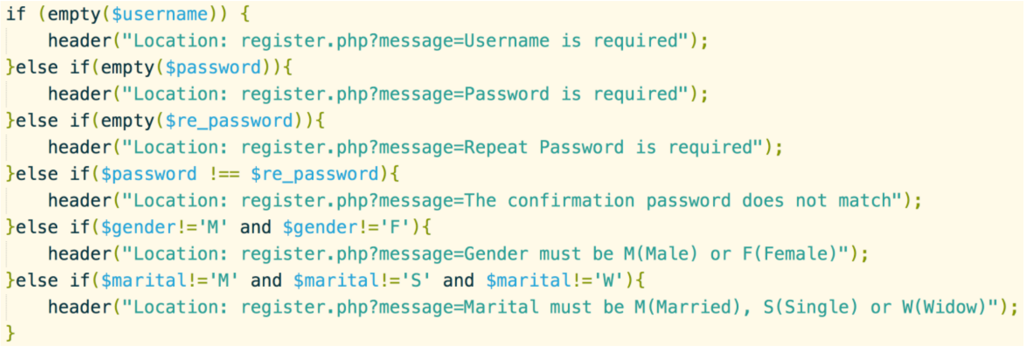
In the form or URL parameter, the malicious JavaScript code is written, which looks like ordinary text, but after being parsed by the browser, it becomes an executable JavaScript action, which is used for attacking. Using PHP’s htmlspecialchars() can convert some special characters like “””, “<”, ”>” into HTML entities. These characters have special meanings. After conversion, these HTML entities can only be recognized by Web browser. For example, a JavaScript will be just a String rather than an executable JavaScript after conversion. Here is an example that we implement htmlspecialchars() that prevent XSS injection.



```
$username = htmlspecialchars($_POST["username"],ENT_QUOTES,'UTF-8');
$password = htmlspecialchars($_POST["password"],ENT_QUOTES,'UTF-8');
```

8.5 System check inputs from users before concatenating them into query strings

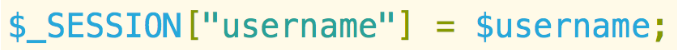
Here is an example that we check user inputs in register process.



```
if (empty($username)) {
    header("Location: register.php?message=Username is required");
}else if(empty($password)){
    header("Location: register.php?message=Password is required");
}else if(empty($re_password)){
    header("Location: register.php?message=Repeat Password is required");
}else if($password !== $re_password){
    header("Location: register.php?message=The confirmation password does not match");
}else if($gender!='M' and $gender!='F'){
    header("Location: register.php?message=Gender must be M(Male) or F(Female)");
}else if($marital!='M' and $marital!='S' and $marital!='W'){
    header("Location: register.php?message=Marital must be M(Married), S(Single) or W(Widow)");
}
```

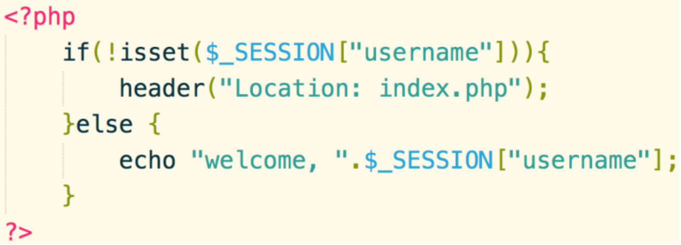
8.6 Keep state for a user session

If a user login successfully, the system will store username information in Web browser session.



```
$_SESSION["username"] = $username;
```

A user cannot access a web page using URL directly without login. There will be no session set. The system will redirect to index.php, which requires users to login first.



```
<?php
    if(!isset($_SESSION["username"])){
        header("Location: index.php");
    }else {
        echo "welcome, ".$_SESSION["username"];
    }
?>
```

9. Lesson learned (as individually and as team), detailing your reflections about project work, what have you learned, what went well and what did not. Constraints you faced, if any (e.g. time management, coordinating project with team member remotely etc.)

The final project was challenging for everyone in our group, and this is the first database with a complete web-based user interface we have ever made. We spent the last couple of weeks starting to learn PHP and how to connect the frontend to our backend database. In addition, we searched online documents and articles, and learned how to use prepared statements in the PHP files to against SQL injection to our database. We completed the project efficiently in a short time thanks to the precise division of work and mutual trust in our group. We keep everyone in our group in the same stage as new features are implemented in the frontend webpage. When we first designed the frontend, one of our team members could not open it, but with the help and advice of other team members, the problem was quickly solved. Through this project, we not only learned new techniques, but also made us realize the importance of teamwork.

Overall, the experience of front-end and back-end development using PHP amd MySQL is great. We have completed the basic requirements, login, registration, CURD and most of the security requirements. However, during the development, myself has made mistakes. I have learned similar but more basic demos. I know we need to start based on the requirements, but we got to codes too hurry. I should have had a more global view and made a plan for this development. Then our development will be more organized. Meanwhile, due to the tight schedule, we cannot go deep into every detail. In my future career life, I should be more careful and do my best to make every project perfect.

-Maohua Shen

This project gives me a more profound concept of how the technology is implemented in real-world situations, not only the tech skill. For example, in this project part 1, I need to figure out the relationship between customer, insurance policy, invoice, payment, and so on based on the finance process and the logical side. In project part 2, I notice setting up the application environment takes a long time, and different operating systems become a hindrance. This problem also happens during our present.I think it will be more efficient if we use a tool like ducker in a container so that every team member will have more time to focus on implementing the function.

— Melody Zhang

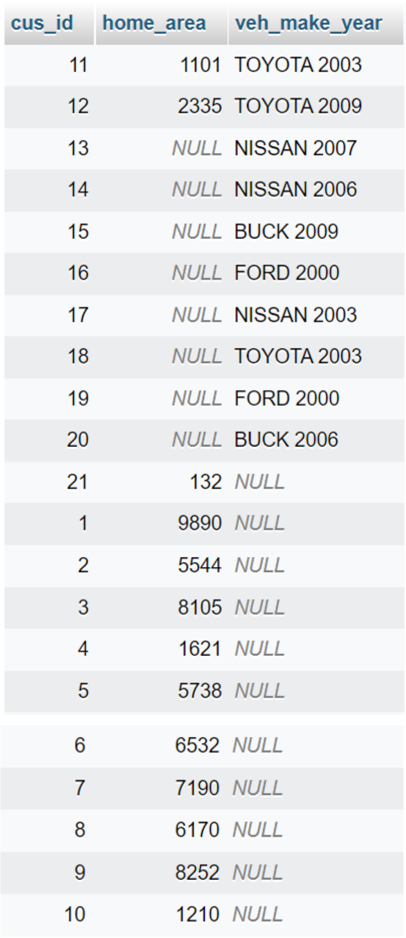
This class is my first time to learn about databases, and this project is my first time to build a frontend and connect it to a database. Before the final project, I have no idea how the data passes from users to the database. But now, I realized the frontend web page consists of multiple php files, and each one has different functionalities. For example, connection.php connects the frontend and backend, it’s like a bridge allowing data to get into the database. Since this is my first time implementing a database with web-based user interface, I did lots of research online and referred to a lot of videos and articles on how to build a frontend web page using PHP. Due to the lack of time, we did not realize all the features, such as, data visualization, Security check on password reset, to make our system perfect and more attractive to the users. But as the first frontend I've been involved in designing, I'm pretty happy with it. I believe this project provides valuable experience for my future study and work.

— Haoyu Wang

10. Business analysis with 6 SQLs using your project data.

10.1 Q1)

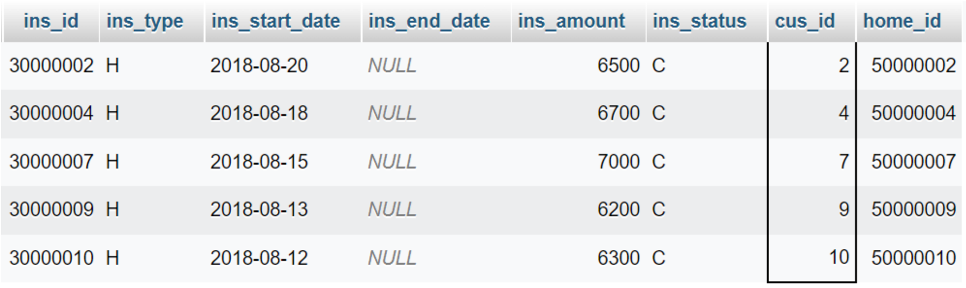
```
select c.cus_id, h.home_area, a.veh_make_year
from customer c
LEFT join home h on h.cus_id=c.cus_id
LEFT join vehicle a on a.cus_id=c.cus_id;
```



cus_id	home_area	veh_make_year
11	1101	TOYOTA 2003
12	2335	TOYOTA 2009
13	NULL	NISSAN 2007
14	NULL	NISSAN 2006
15	NULL	BUCK 2009
16	NULL	FORD 2000
17	NULL	NISSAN 2003
18	NULL	TOYOTA 2003
19	NULL	FORD 2000
20	NULL	BUCK 2006
21	132	NULL
1	9890	NULL
2	5544	NULL
3	8105	NULL
4	1621	NULL
5	5738	NULL
6	6532	NULL
7	7190	NULL
8	6170	NULL
9	8252	NULL
10	1210	NULL

This query shows if a customer has a house or vehicle. So company can recommend appropriate insurance policies for the customers.

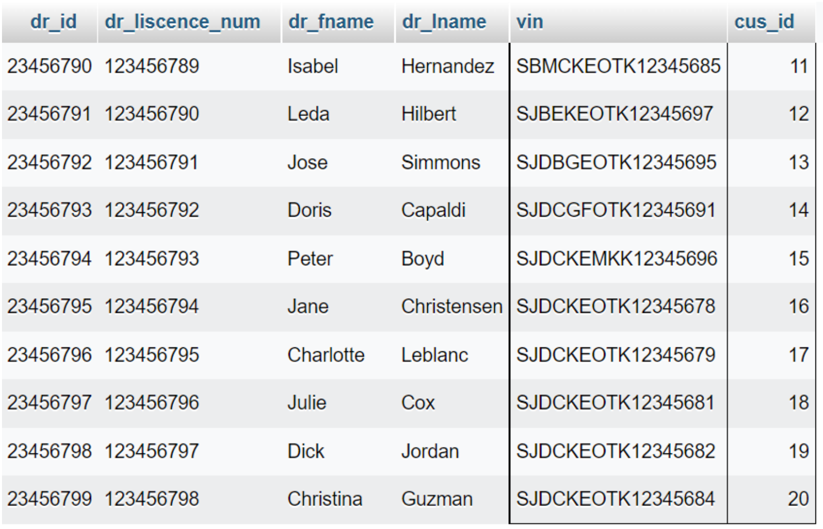
10.2
select * from home_policy
where cus_id in (select cus_id from customer where cus_gender='F');



ins_id	ins_type	ins_start_date	ins_end_date	ins_amount	ins_status	cus_id	home_id
30000002	H	2018-08-20	NULL	6500	C	2	50000002
30000004	H	2018-08-18	NULL	6700	C	4	50000004
30000007	H	2018-08-15	NULL	7000	C	7	50000007
30000009	H	2018-08-13	NULL	6200	C	9	50000009
30000010	H	2018-08-12	NULL	6300	C	10	50000010

This query shows how many home insurance customers are female. With the result, the company can calculate the proportion of female in the home insurance customers.

10.3
SELECT * FROM driver d
WHERE d.cus_id = ANY (SELECT v.cus_id FROM vehicle v WHERE v.vin =
d.vin);

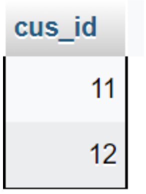


dr_id	dr_liscence_num	dr_fname	dr_lname	vin	cus_id
23456790	123456789	Isabel	Hernandez	SBMCKEOTK12345685	11
23456791	123456790	Leda	Hilbert	SJBEKEOTK12345697	12
23456792	123456791	Jose	Simmons	SJDBGEOTK12345695	13
23456793	123456792	Doris	Capaldi	SJDCGFOTK12345691	14
23456794	123456793	Peter	Boyd	SJDCKEMKK12345696	15
23456795	123456794	Jane	Christensen	SJDCKEOTK12345678	16
23456796	123456795	Charlotte	Leblanc	SJDCKEOTK12345679	17
23456797	123456796	Julie	Cox	SJDCKEOTK12345681	18
23456798	123456797	Dick	Jordan	SJDCKEOTK12345682	19
23456799	123456798	Christina	Guzman	SJDCKEOTK12345684	20

The query returns the information of customers who have both driver license and a vehicle.

10.4

select cus_id from home_policy
intersect
select cus_id from auto_policy;

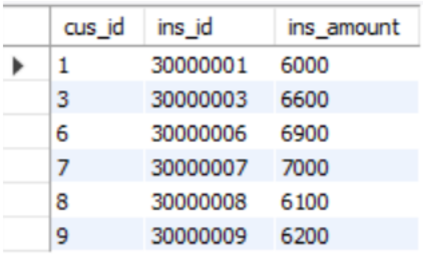


cus_id
11
12

This query intersects home_policy and auto_policy table and extract customer id who have both home and auto insurance in the company.

10.5

WITH
  cte1 AS (SELECT * FROM home h where h.home_area>=6000)
SELECT p.cus_id, p.ins_id, p.ins_amount FROM home_policy p
JOIN cte1
on cte1.cus_id = p.cus_id;



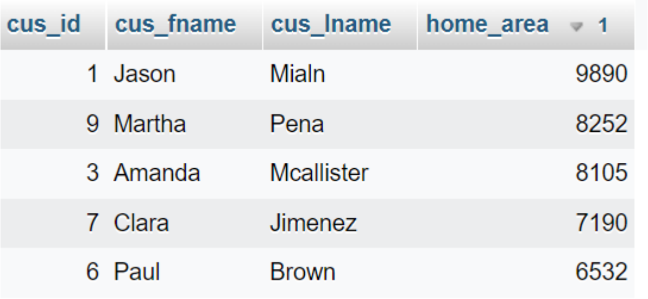
cus_id	ins_id	ins_amount
1	30000001	6000
3	30000003	6600
6	30000006	6900
7	30000007	7000
8	30000008	6100
9	30000009	6200

This query returns customer id, insurance id and amount for customer whose home area are larger or equal to 6000.

10.6)

SELECT c.cus_id, c.cus_fname, c.cus_lname, h.home_area
FROM customer c

```
join home h
on h.cus_id=c.cus_id
ORDER BY h.home_area DESC
LIMIT 5;
```



cus_id	cus_fname	cus_lname	home_area ▼ 1
1	Jason	Mialn	9890
9	Martha	Pena	8252
3	Amanda	Mcallister	8105
7	Clara	Jimenez	7190
6	Paul	Brown	6532

This query shows the id and names of the customers who have the largest home in top 5.