

# **Project Report**

## **Phase 3**



### **Prepared By:**

Database Daredevils

Xin Sun

Ishan Ranpura

Shubham Agarwal

Yifei Xiao

Yubaihe Zhou

### **Advised By:**

Dr. Wang-Chien Lee

April 27, 2018

# Table of Contents

1. Introduction .....	4
2. Requirement Analysis .....	5
3. Conceptual Database Modeling .....	10
3.1. Changes Made in ER Diagram from Phase 1.....	10
3.2. Description of The Changes.....	10
4. Relational Database Design .....	13
4.1. Schema Definition .....	13
4.2. Schema Normalization and SQL Statements .....	14
4.2.1. Company Table .....	15
4.2.2. Individual Table .....	16
4.2.3. Items Table .....	17
4.2.4. Shopping_Cart Table .....	18
4.2.5. Feedback Table .....	19
4.2.6. User and Language Table .....	20
5. Technology Tools Deployed in Project .....	21
5.1 Additions.....	24
6. System Prototype .....	25
6.1 Overview .....	25
6.2 Transactions .....	28
7. Complete System Framework.....	33
8. Reflection.....	34
8.1 Analysis of Goals.....	34
9. Conclusion.....	35
Appendixes	
1. Appendix A. Link for the website demo.....	36

2. Appendix B. 3rd Normalization Tables and SQL statements.....	36
3. Appendix C. SQL tables generated by DataGrip .....	40
4. Appendix D: Photos of the Databases.....	41
5. Appendix E: Bibliography for technology survey.....	42
6. Appendix F: Progress Reports .....	43

## List of Figures

Figure 2.1 ER Diagram.....	6
Figure 2.2.3 Relationship between user categories and items.....	6
Figure 2.2.4 Categories - items.....	7
Figure 2.2.5 Changes from auction_price to bidding.....	7
Figure 2.2.6 Shopping_cart.....	8
Figure 3.1.1 Relational Schemas of Database.....	9
Figure 3.2.1-1 Relational Schema of company.....	10
Figure 3.2.1-2 3 <sup>rd</sup> normal form of company tables.....	10
Figure 3.2.2-1 Relational Schema of individual.....	11
Figure 3.2.2-2 3 <sup>rd</sup> normal form of individual tables.....	12
Figure 3.2.3-1 Relational Schemas of items.....	12
Figure 3.2.3-2 3 <sup>rd</sup> normal form of items tables.....	12
Figure 3.2.5-1 Relational Schemas of shopping_cart.....	14
Figure 3.2.5-2 3 <sup>rd</sup> normal form of shopping_cart tables.....	14
Figure 3.2.6 3 <sup>rd</sup> normal form of feedback table.....	15
Figure 3.2.7-1 Relational Schemas of user and language.....	16
Figure 3.2.7-2 3 <sup>rd</sup> normal form of user and language tables.....	16
Figure 4.1 MySQL Single Query Performance .....	19
Figure 4.2 The company table created by MySQL on Terminal in macOS.....	20
Figure 4.3 DataGrip Interface .....	21

# 1. Introduction

e-Auction is a startup company which aims to pursue the exploding opportunities of online auction business. This report will outline the milestones completed in Phase 1, Phase 2, Phase 3 and then list down goals for the future.

## About the project

This project is a multi-phased research project where Database Daredevils is going to explore the requirements needed for an *e-Auction* to launch in the e-business market. Through different phases in this project, Database Daredevils will analyze the requirements, come up with a conceptual database design and a system prototype for the website. The analysis made by the team provides detailed explanation about the website with three main objectives as below:



Features: Why should you use this website? The website contains great features like multi-language functionality and discounts to attract users to the website.

Great Deals: Did you win the auction and manage to get a great price on an item? That's not it! Users will also get additional discounts on the items they have already bought in the auctions.

Easy to use: Not tech-savvy? e-Auction website's easy to use interface allows anyone with a desire to take part in auction without getting confused on how to use great features available on the website.

## 2. Requirement Analysis

The following is the requirement analysis for storing and accessing database at ease for the website.

1. **Auction Items:** Since the crust of any e-auction website is the items available for sale and bidding, each item available in the database is sold by either by a company or an individual. It consists of the items available for bidding.
2. **Categories:** All the items available at e-Auction are categorized using a predefined classification tree. The root of the tree starts from categories (books, furniture, electronic) with subcategories under it comprising of multiple items. Items comprise of ID, name, quantity, etc. available at different warehouse locations. Maximum number of tree levels are set to 10.
3. **Registered Users:** The registered and verified individual users can sell or buy items through the bidding process or BuyItNow (fixed price) feature on the website. For registration purposes, we plan to put Email ID verification into application. Companies are special type of users, therefore, they do not have to provide any payment information like individual users. For the verification of special kind of users like company users, users will have to upload the federal registration documents where the registration number will serve as an ID.
4. **Companies:** Since companies can only sell products, they do not need to provide the credit card information. To make the process hassle-free for individual users, company need to assign a contact person to contact in case of any product issues. The assigned person is contacted in case of returns, special queries, and things like deals offered by e-Auction to the company.
5. **Rating:** To prevent scams and fraud over the e-auction website, buyers would rate other buyers on a scale of 1-5. Once the rating falls below 1 for the buyer, the buyer will be banned from taking part in any further e-auctions.
6. **Browsing:** Here, we have come up with a model which allows only verified individual users to

browse the website. Individual users and company users perform a browsing action by surfing among several available categories of items (e.g. phones, books etc.). Visitors are not allowed to browse the items through categories, users can browse only when they have signed in their accounts.

- 7. Searching:** Our effective user interface enables users to search for a particular item with the help of a search bar provided at the top of the webpage. An item can be searched by entering the keywords or conditions if the name is yet unknown. Example of searching keywords include 'computers', 'chairs', etc. To optimize the search results, we have enabled users to search like 'computers under \$25', 'Apple laptops', etc.
- 8. Bidding:** To get access to the bidding system, the user provides the verified login information. After successful verification, user proceeds with the bidding process. However, the constraint has been set which allows user to bid only during the live auction duration. In addition to that, the new bid price will automatically increase by 5 % after the user clicks the button "I Want It!". Initial bidding period is set by the user and at the end of the bidding process, the final winner's username will show up on the webpage and the winner will be announced through an email.
- 9. BuyItNow:** An alternate method to buy any product is by placing an order through the BuyItNow price directly. Usually, auction price is lesser than the BuyItNow price. In addition to that, when user tries to buy an item through the BuyItNow price, they can then see the inventory for the item, and have an option of buying more than one quantity of the item. The quantities of items available are mentioned in the inventory.
- 10. Auction Statistics:** The model has been built effectively considering easy collection of statistical data enhancing the efficiency of the statistics report. The required data is collected and generated bi-weekly which contains details like total selling amount and number of items sold during the period. It enables data analysis which can further improve the targeted customer base and lead to

increase in the profits.

- 11. Delivery:** To avoid scams, e-Auction serves as a third party which will connect buyer to the sellers.

Once the order has been placed by the user and approved by the seller, the item will be received from the seller to the e-auction website. After that, e-auction website places a charge on user's credit card. After the payment is approved, the e-auction website performs the further delivery process to the user's shipping address. This provides one extra layer of security in case of faulty item or a fraud.

- 12. Reports to Telemarketers:** To enable functionalities developed in auction statistics, e-auction website sends the user's information like income, date of birth, etc. to the telemarketer companies for a survey twice a week which is further analyzed by the telemarketing companies.



### **Additional Features**

Below listed are some additional features the team has suggested for the website.

- 13. FAQs:** This could enable each item to have few questions and answers about the item, which could make it easier for users to know the product better. Such information is provided by the seller of the product.
- 14. Feedback:** This feature is proposed with a purpose to update sellers with the comments on the items sold. Also, it will enable sellers to listen to the queries of the customers and enhance the product quality.
- 15. Shopping Cart and Coupon:** Buyers could put the items that they intend to purchase in BuyItNow price into this shopping cart, and if the buyer has won the bidding, the item will appear automatically in the buyer's shopping cart. The coupon will be fixed according to the various ranges of the total shopping cart amount. And based on the coupon, the shopping cart will generate the net amount

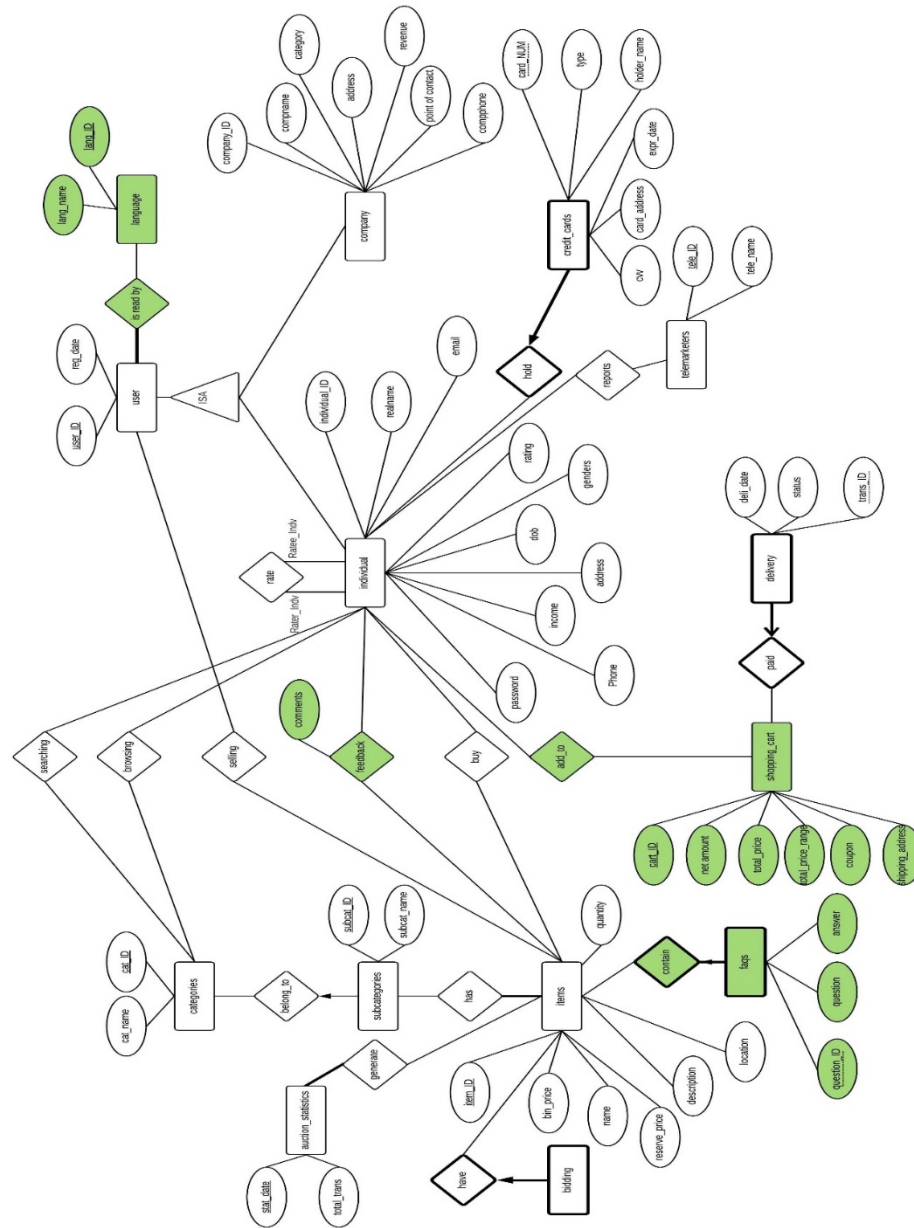
during the check-out process. Discount is provided by the e-Auction website after generating the coupons based on the data collected from the auction statistics report.

- 16. Multiple Languages:** An additional feature where users are provided with the language priorities comprising of English, Hindi, Chinese, etc. allowing them to effectively interact with the UI. Here, the website is translated to the desired language upon selection providing an ease of access.





## ER-diagram



*Figure 2.1 ER Diagram*

### 3. Conceptual Database Modeling

#### 3.1 Changes made in ER Diagram from Phase 1

There were a lot of changes made in Phase 2 ER diagram compared to Phase-1. The team gave a good consideration to the feedback provided and inculcated the changes to improve the ER diagram and thereby, the user experience. Some of the changes made range from adding new entity sets and primary keys to modifying the existing relationships to better suit the requirements of the database.

#### 3.2 Description of the changes

In this section, we will focus on the improvements we made for the Entity-Relationship model and thereby, highlighting the need for those changes. Later, we plan to discuss the results based on our observation due to the changes. Full updated version of ER model has been added to the appendices.

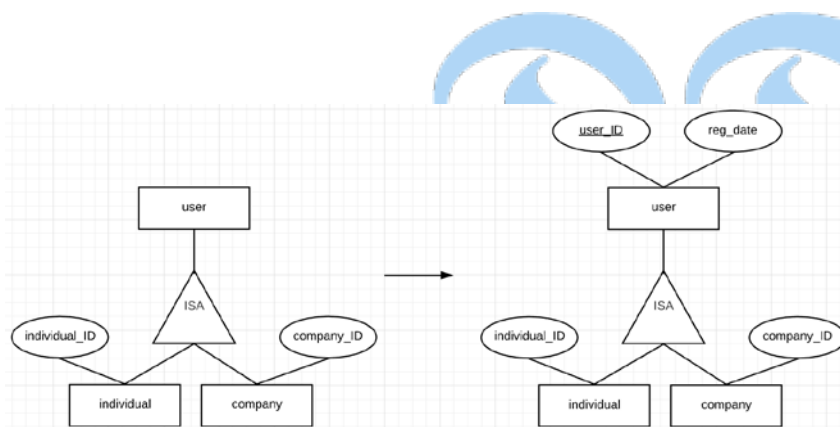
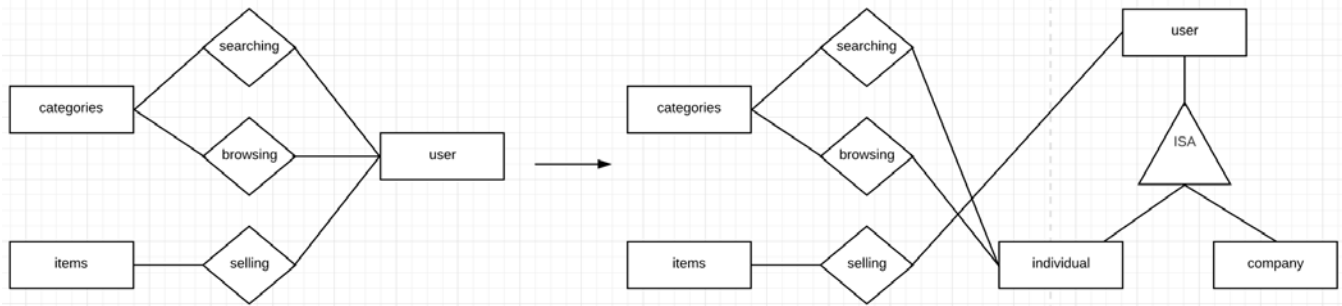


Figure 3.2.2 user ISA hierarchy

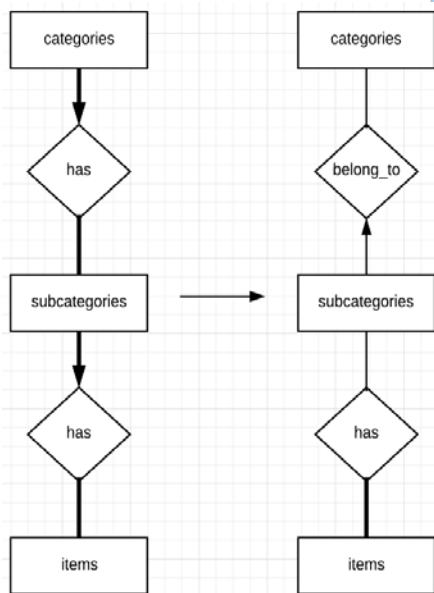
The **first thing** we changed was the 'user' entity set. The original 'user' entity set did not contain *user\_ID*, instead there were *indv\_ID* and *comp\_ID* which serve as a primary key of 'individual' and 'company' entity

set, respectively. However, we noticed that individual entity set has many relations between other entity sets, such as, 'items', 'shopping cart' ... so we add the *user\_ID* as a primary key and gave 'user' entity set an attribute called *reg\_date* which records the registration dates for each user. In addition to that, there were few relationships which involved both the individual and company entity sets and therefore, to reduce the redundant tables in the database, we decided to come up with the *user\_ID*.



**Figure 3.2.3** relationships between user and categories, items

**Next change** was ‘searching’ and ‘browsing’. Earlier, relation tables ‘searching’ and ‘browsing’ were connected with the ‘user’ entity set. However, since companies are ‘special’ type of users who only sell items on the website, we want to provide them with the limited access in contrast to the individual users. With that, companies get access to the features other than searching and browsing on the website. ‘Selling’ function is enabled for both the company and the individual users with the help of *user\_ID* as explained in the previous paragraph.



**Figure 3.2.4** categories - items

implying one subcategory can be linked to only one category.

**Next**, we decided to add the integrity constraints between ‘categories’, ‘subcategories’ and ‘items’. While creating the SQL tables for these entity sets, we experienced difficulties due to the oversetting of the not required integrity constraints. Since each item has to belong to at least one subcategory, we save the total participation between ‘items’ and ‘subcategories’. In addition to that, reducing the non-required constraints will enable the e-auction website to add few special items at its own discretion. To reduce the redundancy, we came up with a one to many key constraint between ‘subcategories’ and ‘categories’

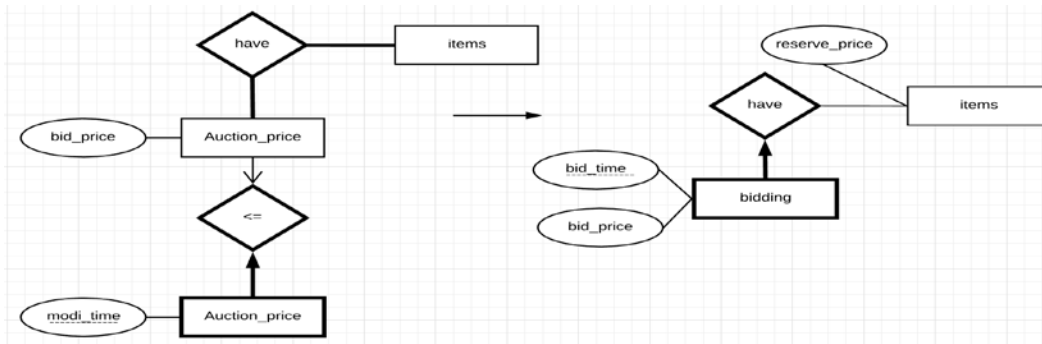


Figure 3.2.5 changes from auction\_price to bidding

**Further**, the major change was to redo the bidding system. In the previous version, we set auction price and reserve price as entity sets and gave them attribute *ending\_time* and *modi\_time*. We noticed that the reserve price should be decided by the seller and therefore, concluded to remove the ‘Reserve\_Price’ entity set and further, changed it to an attribute. Also the attribute ‘<=’ was ambiguous since it was not clear if the auction price be less than or equal to the reserve price or vice versa. To improvise the design, we dropped it and put it into the new ‘bidding’ entity set. Notice, if an item is deleted by a seller, then there should not be any bidding operation for that item. To enable that functionality, we set the ‘bidding’ entity set a weak entity set. To take care of the unique time during an e-auction, *bid\_time* has been made unique using a partial key due to weak entity set. Also, the *bid\_price* will be updated and only store the latest bidding price of the item.

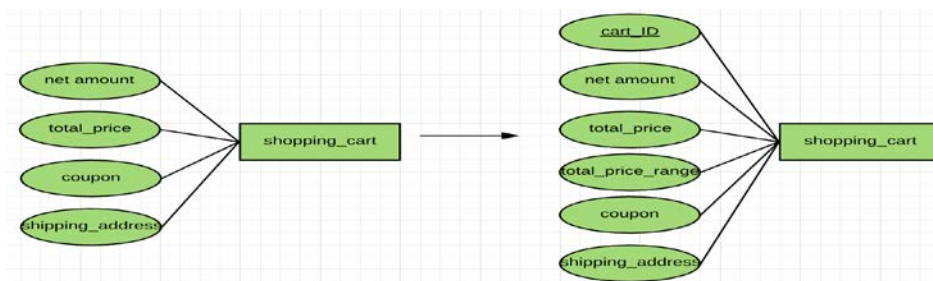


Figure 3.2.6 shopping\_cart

**Further**, we have reorganized our shopping\_cart. Here, we added a primary key, *cart\_ID* to the shopping cart entity set to uniquely identify the records. Along with that, since the coupon is generated based on the total price in the active cart and the challenge being the infinite values which the total price can possess, we added an additional attribute called *total\_price\_range*. Based on this range, the coupons are applied.

Thereafter, the calculation of the net amount is carried out based on the total price and the corresponding coupon.

**Moreover**, we decided to set ‘faqs’ entity set as a weak entity since, if the item gets deleted, there will be no corresponding questions and answers. We have set ‘delivery’ entity set as a weak entity by taking into consideration the same reason. If a shopping cart was removed i.e. doesn’t exist then, there should not exist the delivery functionality. Along with these modifications, we have added a partial key *question\_ID* to the ‘faqs’ entity. We noticed that in the previous ER diagram, we did not have a place to store ratings of each individual user, so we have added the *rating* attribute. Although there is always a scope of improvement but with the above modifications, we are confident of our design enabling us to generate SQL tables and normalize them to the highest normal form possible.

## 4. Relational Database Design

### 4.1 Schema Definition

A schema is the structure behind data organization. The definition of schema is the base for creating the SQL statements in the database. Database schema defines its entities and the relationship among each them. The followings are the relational schemas of the database developed based on the requirement analysis and ER design. Notice the arrows in the diagram represent foreign key constraints.

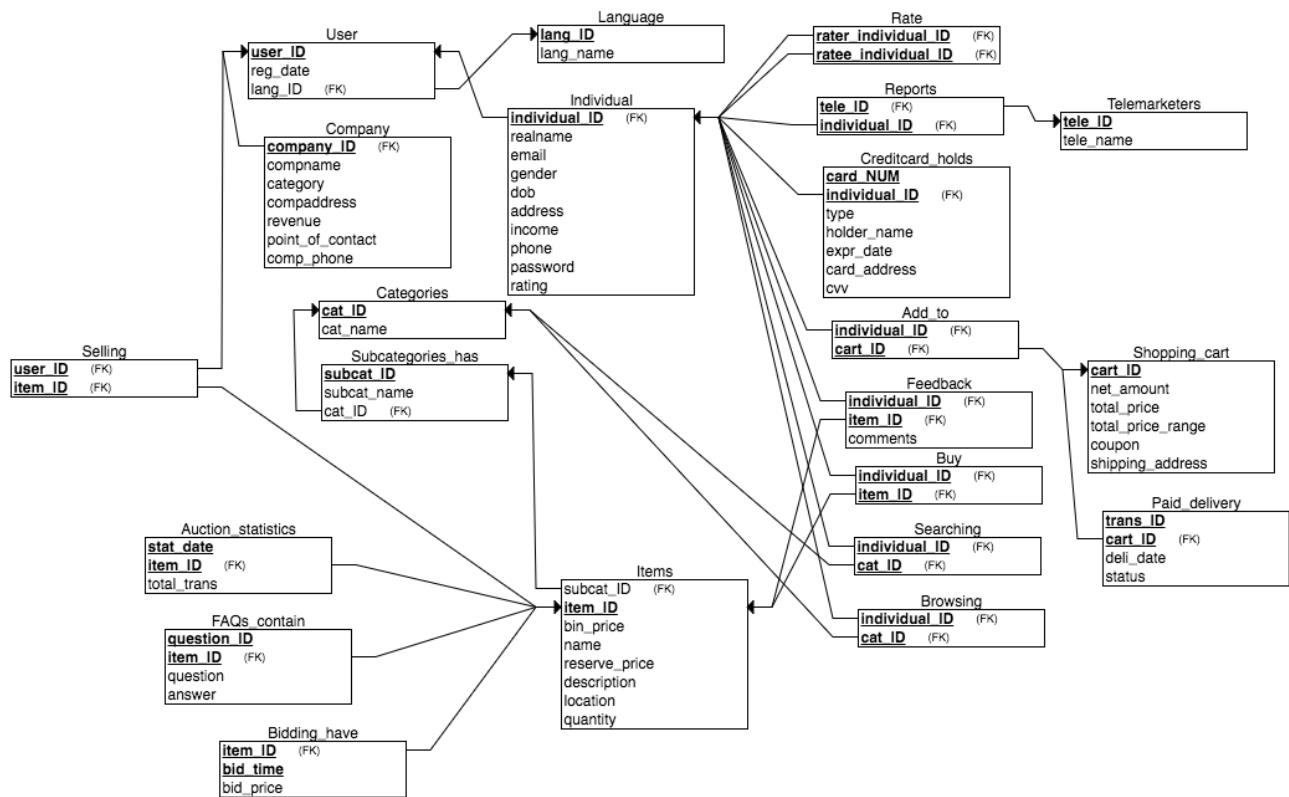


Figure 4.1.1 Relational Schemas of Database

The entity integrity constraint states that primary keys cannot be null and there must be a proper value in the primary key field. Here, the user\_ID is a unique id which is assigned to a user after registration. It is used to identify individual users in the table. The user\_ID should not be null and based on the ISA hierarchy. User is the root entity set of individual and company entity sets, so user\_ID is also the foreign key reference of both individual\_ID and company\_ID.

## 4.2 Schema Normalization and SQL Statements

There were several functional dependencies that our features specified. To make the best use of time and thereby, come out with the best possible optimum design, we have carried out normalization for all the entity sets and relationships. An important part of our design is ensuring that the 3rd Normal Form (3NF) is achieved through eliminating redundancy and non-atomic values. We need to refine the schema through normalization in order to reduce and control data redundancies to an acceptable level (at least in the 3rd NF).

In this project, we are working on implementation of the full e-auction website from scratch and so, dealing with more than 30 tables. Here, we will be discussing some of the significant entity sets, their tables, schemas, and the normalization process from 1st NF to 3rd NF.

#### 4.2.1 Company Table

Initially, we started with the schema design which derived from the ER diagram:

**Company**(company\_ID, compname, revenue, category, address, pointofcontact, compphone)

The primary key is *company\_ID*, and all other non-key attributes are dependent on the primary key - *company\_ID*, so the company table is in the 1st normal form. And since there is no non-key attribute functionally dependent on a part of the primary key, the company table is already in the 2nd NF.

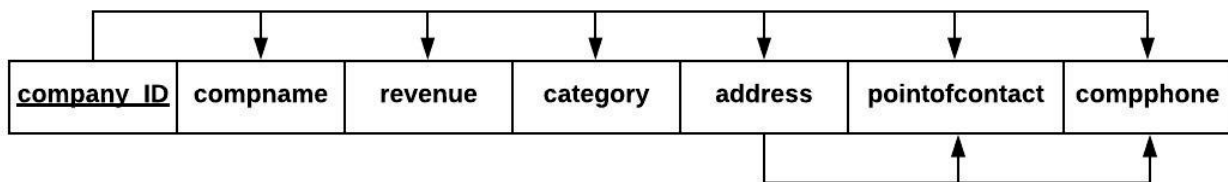


Figure 4.2.1-1 Relational Schema of company

However, since *pointofcontact* and *compphone* are also be determined by *address*, this is a **functional dependency**, which non-key attributes are determined by another non-key attribute, so we need to decompose it into 3rd normal form.



Figure 4.2.1-2 3<sup>rd</sup> normal form of company tables

The corresponding SQL statements are:

<pre>CREATE TABLE company(     company_ID varchar(20),     compname varchar(20),     revenue varchar(20),     category varchar(20),     address varchar(50),     PRIMARY KEY (company_ID),     FOREIGN KEY (company_ID) REFERENCES         luserid(user_ID));</pre>	<pre>CREATE TABLE address(     address varchar(50),     pointofcontact varchar(20),     compphone varchar(20),     PRIMARY KEY (address));</pre>
---	--

Moreover, for simplicity, since our model is already a full-fledged complex design of the e-Auction Website, we have assumed that the address is single-valued rather than being multi-valued in the real-life scenario.

#### 4.2.2 Individual Table

Initially, we started with the schema design which derived from the ER diagram:

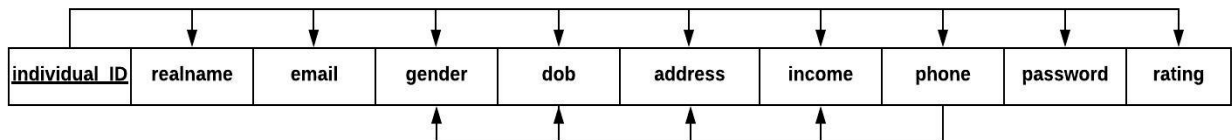


Figure 4.2.2-1 Relational Schema of individual

The primary key is *individual\_ID*, and all other non-key attributes are dependent on the primary key *individual\_ID*, so the **Individual** table is in the 1st normal form. And since there is no non-key attribute functionally dependent on a part of the primary key, the **Individual** table is already in the 2<sup>nd</sup> normal form.

Due to the rating system, individuals that buy items can rate each other, in order to store their rating, we included an attribute *rating*, which is determined by *individual\_ID* like other non-key attributes. However, since the individual might change his/her phone number in the future, and since the phone number is unique, the corresponding gender, dob, address, and income of the person with the phone number might change too. In addition, attributes *gender*, *dob*, *address* and *income* are also determined by *phone*, which



is a *functional dependency* of the case: non-key attributes are determined by another non-key attribute. So, we need to decompose it into 3rd normal form.

<u>individual_ID</u>	realname	email	password	rating	phone
----------------------	----------	-------	----------	--------	-------

<u>phone</u>	gender	dob	address	income
--------------	--------	-----	---------	--------

Figure 4.2.2-2 3<sup>rd</sup> normal form of individual tables

After normalization, the tables above are in the 3rd normal form and the corresponding SQL statements are:

<pre>CREATE TABLE individual(     individual_ID VARCHAR(20),     realname VARCHAR(25),     email VARCHAR(30),     password VARCHAR(20),     phone VARCHAR(20),     PRIMARY KEY (individual_ID),     FOREIGN KEY (individual_ID) REFERENCES         luserid(user_ID));</pre>	<pre>CREATE TABLE iphone(     phone VARCHAR(20),     gender VARCHAR(10),     dob DATE,     address VARCHAR(50),     income VARCHAR(20),     PRIMARY KEY (phone));</pre>
---	---

### 4.2.3 Items Table

Initially, we started with the schema design derived from the ER diagram:

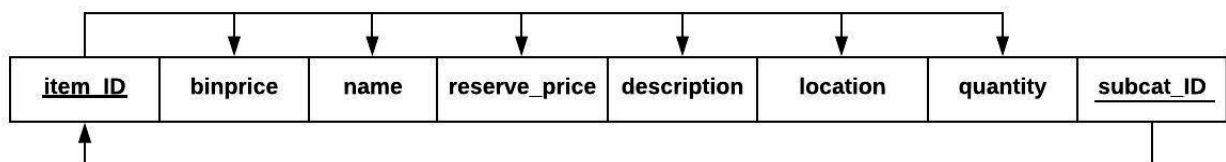


Figure 4.2.3-1 Relational Schema of items

Since 'items' entity set and 'has' relation set has a total participation integrity constraint, *subcat\_ID* determines *item\_ID*. *subcat\_ID* is a foreign key in the **items** table. We decomposed the schema as the following table (which is already in the 3rd normal form):

<u>item_ID</u>	binprice	name	reserveprice	description	location	quantity
----------------	----------	------	--------------	-------------	----------	----------

<u>subcat_ID</u>	<u>item_ID</u>
------------------	----------------

Figure 4.2.3-2 3<sup>rd</sup> normal form of item tables

In the table above, all non-key attributes are dependent only on the primary key - *item\_ID*, no non-key attributes are determined by another non-key attribute, so the **items** tables are already in the 3rd normal form.

The corresponding SQL statements are:

<pre>CREATE TABLE items(   item_ID VARCHAR(20),   binprice VARCHAR(20),   name VARCHAR(20),   reserveprice VARCHAR(20),   description VARCHAR(50),   location VARCHAR(50),   quantity INT(10),   PRIMARY KEY (item_ID));</pre>	<pre>CREATE TABLE itemlink(   subcat_ID VARCHAR(20),   item_ID VARCHAR(20),   PRIMARY KEY (subcat_ID, item_ID),   FOREIGN KEY (subcat_ID) REFERENCES subcategories(subcat_ID),   FOREIGN KEY (item_ID) REFERENCES items(item_ID));</pre>
--	--

#### 4.2.4 Shopping\_cart Table

The schema design derived from the ER diagram:

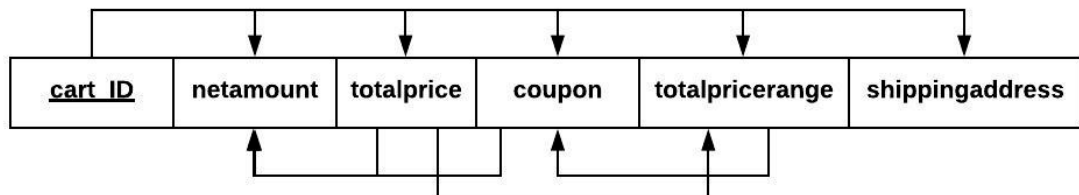


Figure 4.2.4-1 Relational Schema of shopping\_cart

In the above schema, all non-key attributes are dependent on the primary key - *cart\_ID*. And so, the **shoppingcart** table is in the 1st normal form. Moreover, since there is no non-key attribute functionally dependent on a part of the primary key, the **shoppingcart** table is already in the 2nd normal form.

In order to introduce our new feature - coupon, we added an attribute called *totalpricerange*. So, our final version of the **functional dependency** is: *totalpricerange* is determined by *totalprice*, *coupon* is

determined by *totalpricerange*, and finally, *netamount* is determined by both *totalpricerange* and *coupon*.

<u>cart_ID</u>	total_price	shipping_address
----------------	-------------	------------------

<u>total_price</u>	total_price_range
--------------------	-------------------

<u>total_price_range</u>	coupon
--------------------------	--------

<u>total_price</u>	<u>coupon</u>	net_amount
--------------------	---------------	------------

Figure 4.2.4-2 3<sup>rd</sup> normal form of shopping\_cart table

And now, after normalization, the above tables are in the 3rd normal form.

The corresponding SQL statements are:

<pre>CREATE TABLE totpricelink(     totalpricerng VARCHAR(50),     coupon VARCHAR(20),     PRIMARY KEY (totalpricerng));</pre>	<pre>CREATE TABLE netamt(     totalprice VARCHAR(20),     coupon VARCHAR(20),     netamount VARCHAR(20),     PRIMARY KEY (totalprice, coupon));</pre>
<pre>CREATE TABLE shoppingcart(     cart_ID VARCHAR(20),     totalprice VARCHAR(20),     shippingaddress VARCHAR(50),     PRIMARY KEY (cart_ID));</pre>	<pre>CREATE TABLE totprice(     totalprice VARCHAR(20),     totalpricerng VARCHAR(50),     PRIMARY KEY (totalprice));</pre>

#### 4.2.5 Feedback Table

We create this kind of relation table directly from the ER diagram. **Feedback** table has an attribute *comments*, which is not a primary key, and it is determined by both *individual\_ID* and *item\_ID*. So we just need to simply add the attribute into the table (which is already in the 3rd normal form):

<u>individual_ID</u>	<u>item_ID</u>	comments
----------------------	----------------	----------

Figure 4.2.5 3<sup>rd</sup> normal form of feedback table

The corresponding SQL statements are:

```
CREATE TABLE feedback(
    individual_ID VARCHAR(20),
    item_ID VARCHAR(20),
    comments VARCHAR(80),
    PRIMARY KEY (individual_ID, item_ID),
    FOREIGN KEY (item_ID) REFERENCES items(item_ID)
```

#### 4.2.6 User Tables and Language Tables

Before we changed the ER diagram, for the ISA hierarchy, we only created two tables - **individual** and **company**. We noticed that *individual\_ID* has too many relations with other entity sets and so we added the *user\_ID* and *regdate* in in the user entity set. So in this way, the union of *individual\_ID* and *company\_ID* will be the subsets of *user\_ID*.

After we made the changes on user entity set, we noticed that every user will have to have a type of language. So we combined user, isreadby and language together and got the following schema design:

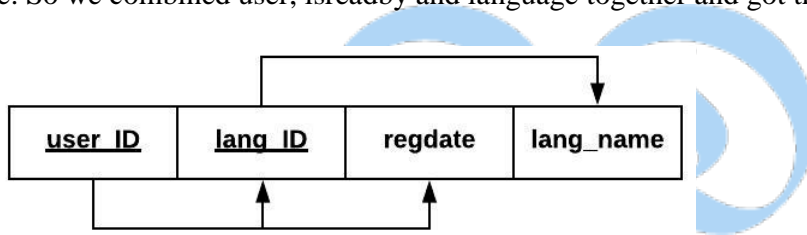


Figure 4.2.6-1 Relational Schema of user and language

In the above schema, non-key attribute *regdate* is dependent on the primary key - *user\_ID*, non-key attribute *lang\_name* is dependent on the primary key - *lang\_ID* and the primary key - *lang\_ID* is determined by primary key - *user\_ID*. So we decomposed the schema directly in the 3rd normal form. We have the table as following:

<u>user_ID</u>	reg_date
<u>lang_ID</u>	lang_name
<u>user_ID</u>	<u>lang_ID</u>

Figure 4.2.6-2 3<sup>rd</sup> normal form of user and language tables

The corresponding SQL statements are:

<pre>CREATE TABLE luserid(   user_ID VARCHAR(20),   regdate DATE,   PRIMARY KEY(user_ID));</pre>	<pre>CREATE TABLE language(   lang_ID VARCHAR(20),   langname VARCHAR(30),   PRIMARY KEY(lang_ID));</pre>
<pre>CREATE TABLE llangid(   user_ID VARCHAR(20),   lang_ID VARCHAR(20),   PRIMARY KEY (user_ID, lang_ID),   FOREIGN KEY (user_ID) REFERENCES luserid(user_ID),   FOREIGN KEY (lang_ID) REFERENCES language(lang_ID));</pre>	

We have 22 tables in total, but due to space limitation, we cannot explain all normalization in the main report. We choose and explain some representative tables. In fact, many tables are similar to each other. For example, creditcard, faqs, bidding, and delivery tables are all derived from weak entity sets, so the way it got normalized is similar to how we normalize the creditcard table. Also, reports, searching, selling, browsing, buy, addto, and rate tables are all relation sets that has no attribute inside, so the way we normalized this kind of tables is similar to how we deal with the feedback table. Auction\_statistics, categories, and subcategories tables are similar to what we did for the telemarketers table. The rest of tables not shown here can be found in Appendix A.

## 5. Technology tools deployed in the Project

Since the main focus for any website to be successful is the user experience, we have made a potential attempt to develop the interactive and user-friendly website. Although we carried out considerable amount of research, we kept exploring the latest technologies available and used in the web development industry. The learning which we gained during Phase 2 is worth noting, however, Phase 3 introduced us to the Bootstrap which eventually led to an effective user interface.

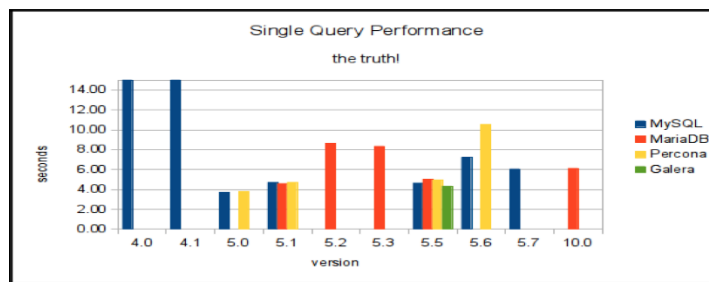
## MySQL



MySQL has been developed by Oracle Corporation since 1995. It has grown a large following. Some famous organizations like Twitter, YouTube, NASA and Walmart are using MySQL. MySQL stores data in tables and uses the SQL (that's why they get the name) to access the data. In MySQL, users must use schemas to define the database structure, requiring that all rows within a table have the same structure with values being represented by a specific data type. Also MySQL supports atomic transactions. One can have several operations with a transaction and can roll back as if you have a single operation. One advantage of MySQL is that the community in MySQL is much better than other database managements like NoSQL. This is because MySQL has been around for several years. [1][2]

### Reasons for MySQL

Our team compared different kinds of database management system. We excluded MS Access first. Although it is easy to use, it did not meet our requirements. There are too many constraints by using MS Access. Then we removed MongoDB. We should admit that as a new software, MongoDB has many things going for it. However, we needed to follow what we have learnt from class. So for each table we need to use schema to define the database structure. Also we need to use JOIN operation to check normalization. Then we compared MySQL to its peers. We found a table about MySQL single query performance. It compares MySQL with other Database systems. [6]



*Figure 5.1 MySQL Single Query Performance*

As we can see from the table, MySQL runs faster than other software. So we choose MySQL as our database management system. But we can only do MySQL on Terminal in macOS or Command prompt in Windows, the interface is not good enough. As we can see, the interface of MySQL in Terminal on macOS does not look pretty, and things may get messy when we add more and more tables in our database. We needed to find some IDE to run MySQL.

```
mysql> create table company(company_ID varchar(20), compname varchar(20), revenue varchar(20),
category varchar(20), primary key (company_ID));
Query OK, 0 rows affected (0.04 sec)

mysql> describe company;
```

Field	Type	Null	Key	Default	Extra
company_ID	varchar(20)	NO	PRI	NULL	
compname	varchar(20)	YES		NULL	
revenue	varchar(20)	YES		NULL	
category	varchar(20)	YES		NULL	

```
4 rows in set (0.00 sec)
```

*Figure 5.2 The company table created by MySQL on Terminal in macOS*

## **Reasons for DataGrip**

Then we found an IDE called “DataGrip”. This IDE is developed by JetBrains and supports MySQL perfectly. Furthermore, it has a smart text editor. We can create our tables much easier as shown in the picture. Also by using DataGrip, we are able to manage our tables clearly such as creating, editing and removing. What’s more, DataGrip enables us to explore SQL tables and their relationships on an insightful and visual diagram, which is extremely important for us. We need this diagram to help us analyze the whole project. (The whole diagram can be seen in Appendix D.) We are pretty sure that we will change our ER-diagram and SQL tables in next phase because more knowledge we learn; more questions will occur.

## **Front-End - Reasons for HTML5**

For the front end, the team has decided to use HTML5. Many web page related technologies which would previously have required 3rd party plug-ins and scripting to specify related technologies are readily specified by HTML5. The web page is able to take all commonly included content varieties and interactivity into the markup language and the browser for granted due to the HTML5 standard.

## 5.1 Additions

### **Bootstrap**

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. The advantage of using Bootstrap is that it comes handy and easy to integrate with the developed code. So, anyone with basic knowledge about HTML and JS can quickly learn to use it. It has mobile first approach which makes the website extremely appealing to look at in the mobile browsers. Moreover, it is compatible with all the modern browsers like Safari, and Firefox.



*Figure 5.3 Bootstrap*



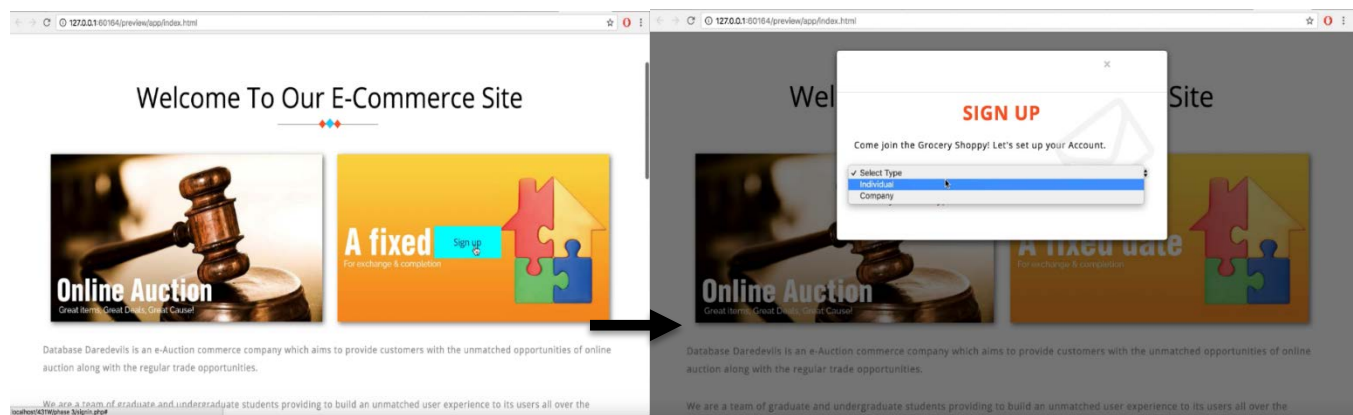


## 6. System Prototype

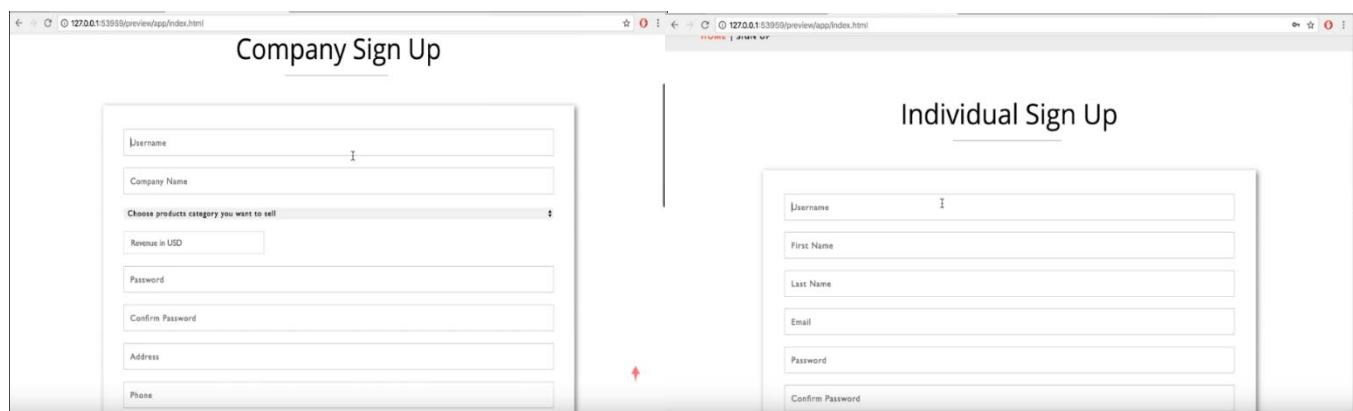
### 6.1 Overview

We take this section to provide an overview on the multi-functional website. We take this opportunity to showcase the working of different components at the backend and frontend which provide a great experience to the user to use the website. Additionally, in **Appendix A**, we included a YouTube link for the website demonstration and a shared google drive link for all the codes.

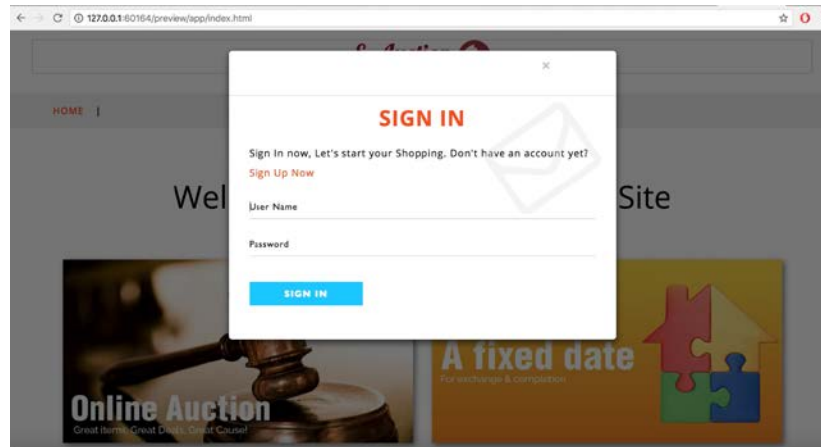
- On the website homepage, a new user can sign up as an individual or as a company.



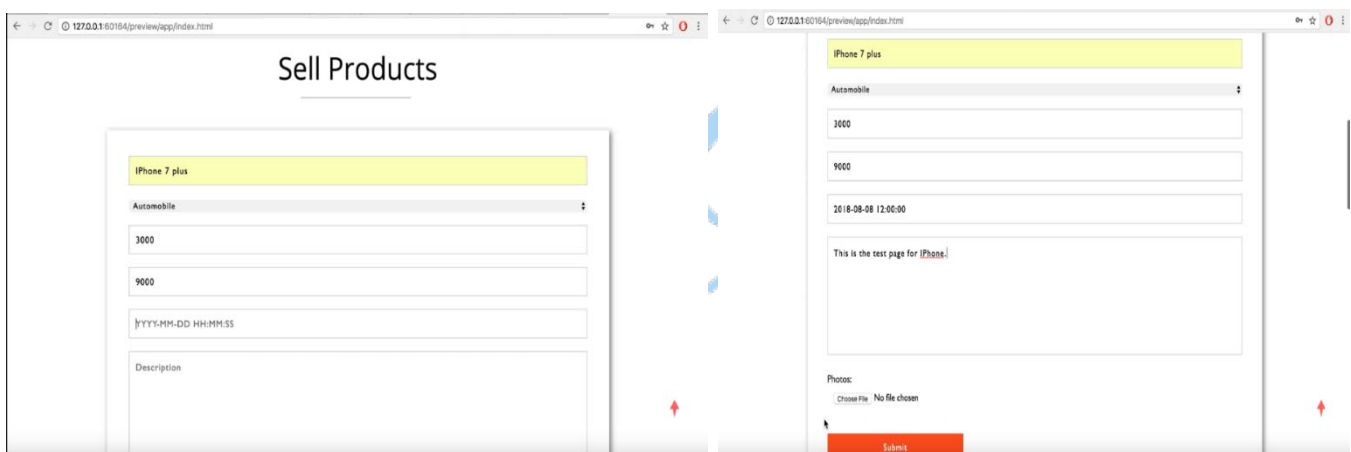
- Since the requirements of companies are different, sign up details required for a company are different with the individual user.



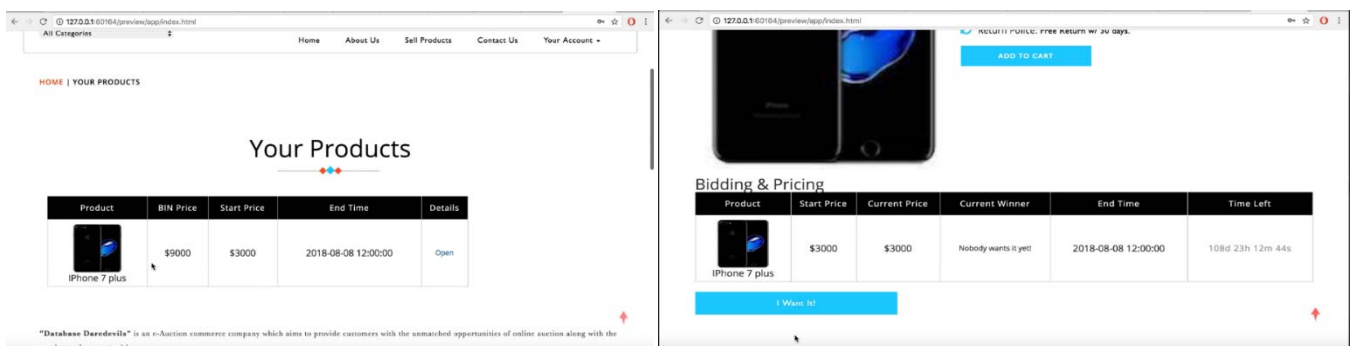
- Once the users registration is successful, they are able to access the website using Sign In page.



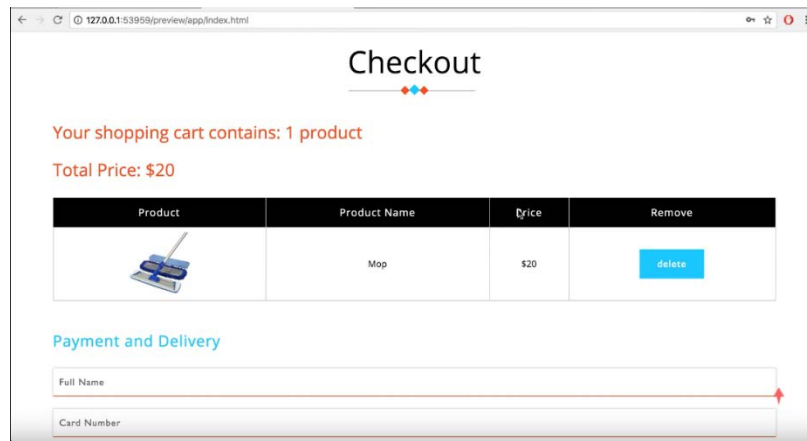
- When a user wants to sell products, they are provided with a Sell Product page where they enter the item details.



- To enable sellers keep up with the process, the Your Products page has been developed.




- After the bidding process is complete and the user wins an auction or opts for the 'Buy It Now' price, the item gets enabled for the check-out process.



Checkout

Your shopping cart contains: 1 product

Total Price: \$20

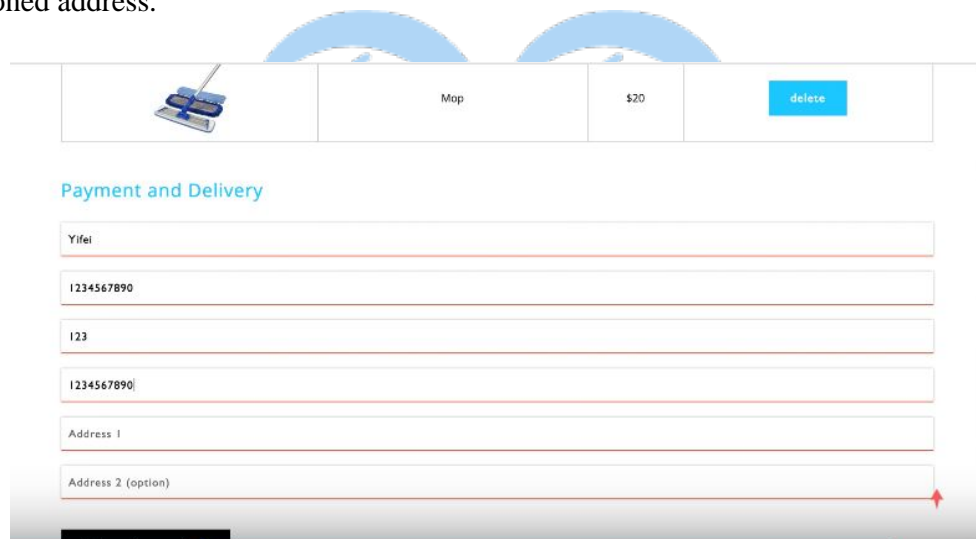
Product	Product Name	Price	Remove
	Mop	\$20	<button>delete</button>

Payment and Delivery

Full Name

Card Number


- After successful addition to the cart, user enters the payment information and delivery attempt is made to the mentioned address.



Checkout

Your shopping cart contains: 1 product

Total Price: \$20

Product	Product Name	Price	Remove
	Mop	\$20	<button>delete</button>

Payment and Delivery

Full Name

Card Number

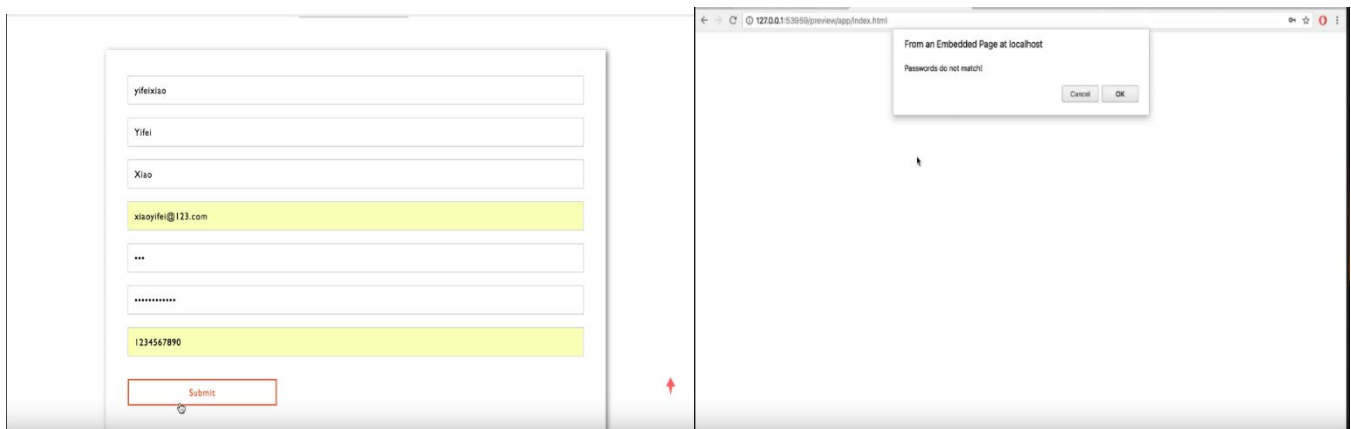
Address 1

Address 2 (option)

## 6.2 Transactions

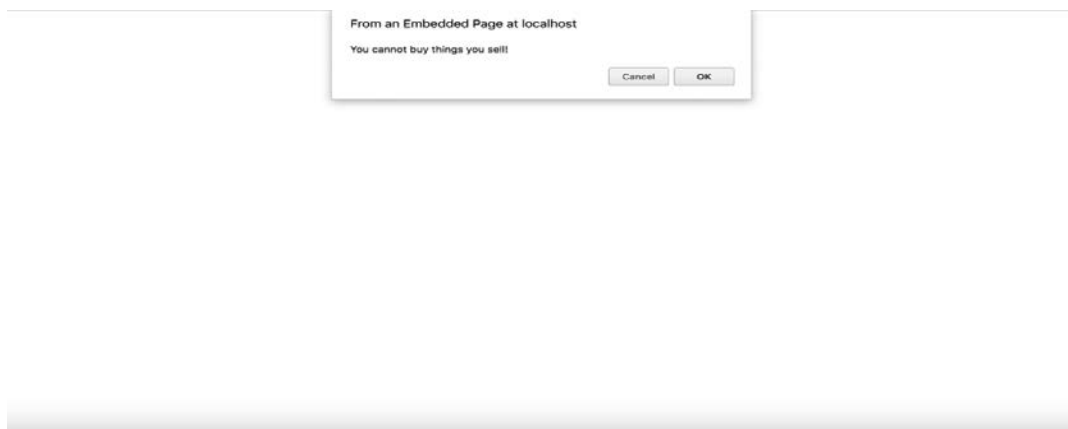
### 1. RegisterUser

To transact with the website, the users can either register as a company or an individual. Given companies and individual possess different requirements, both have different sign up forms. It is mandatory for the company to have a certain revenue making sure reputable firms gets partnered with the website. To keep the functioning more-realistic, we have put constraints on various fields of the sign-up forms, like the password is not visible and encrypted too. This attempt makes sure that authentic buyers and sellers get attached to the e-auction.



### 2. AddAuctionItem

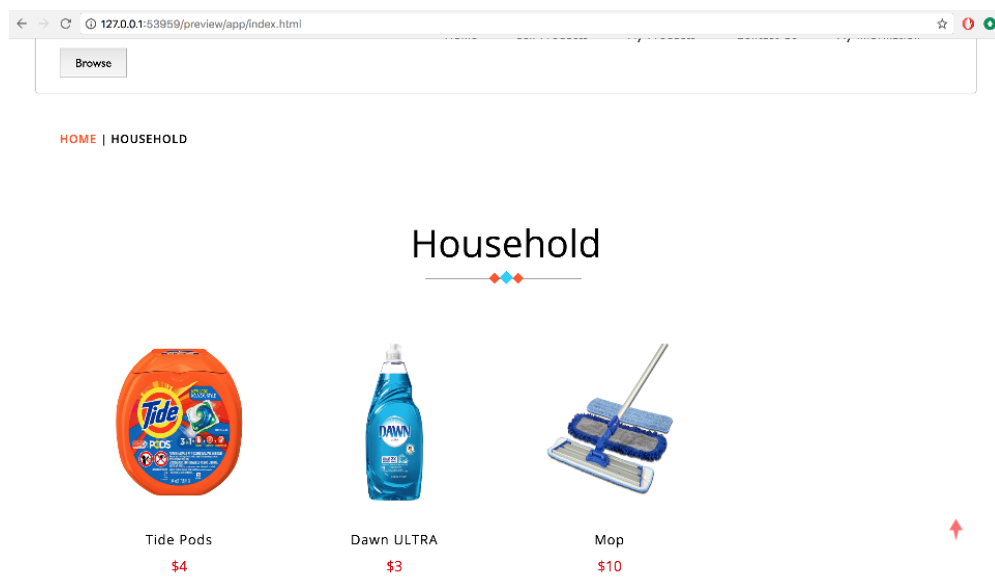
To make the selling transaction, users enter the details about the items such as item name, description, end of auction date, time, and picture. To make the bidding process transparent, we have gone one step ahead by



disabling the seller to bid on his/her own added item. Team has taken an extra care to make sure that the seller of the item cannot bid on the item in an auction.

### 3. BrowsingItems

To develop an effective browsing experience, we have come up with categories and subcategories on the website. For e.g.: If the user is looking for some product for the house, but doesn't have any specific product name in the mind, they can simply click on the 'household' items category which will allow them to browse for various items under the category and subcategory.



### 4. SearchingItems

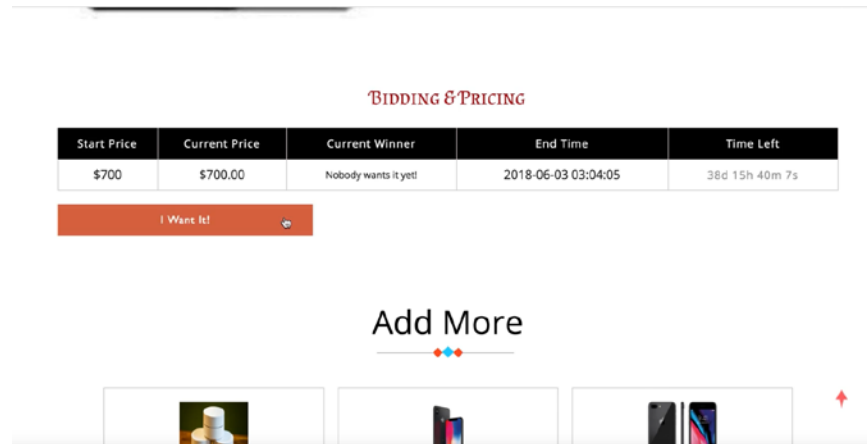
We have enabled searching in our model. Our search bar enables users to search for the items by entering few keywords or keywords with conditions (e.g., iPhone). As a search result, a list of items satisfying the search



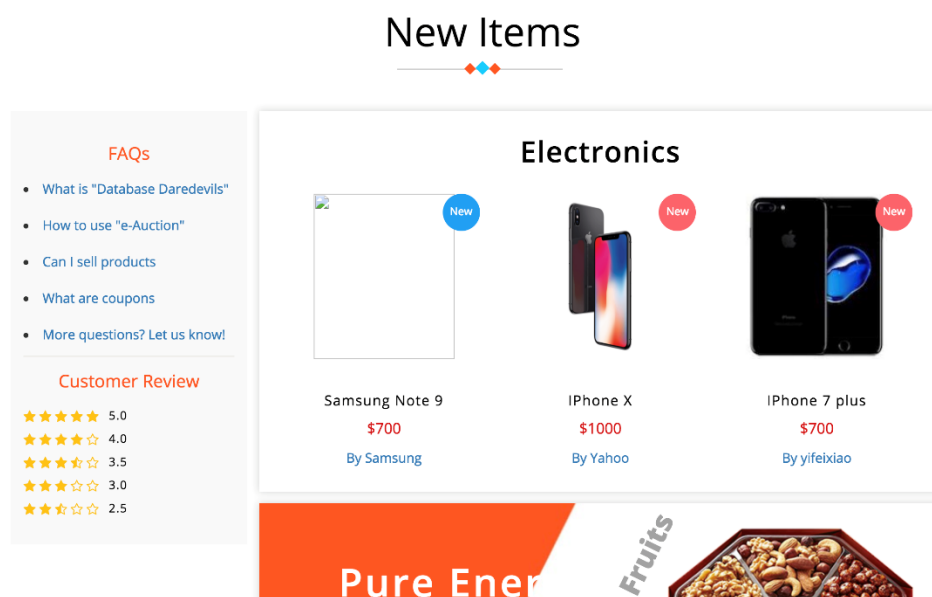
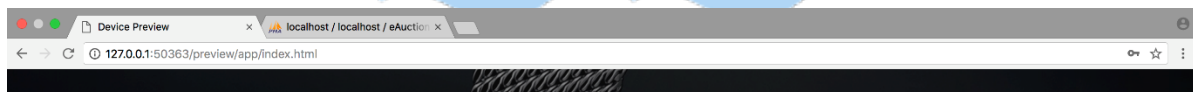
criteria gets returned to the user as shown in the image.

## 5. BidItem

Our website allows easy bidding process. When a user clicks on the 'Bid Now' button and makes a bid for a product, he gets added to the bidding process. To make higher profit and potential higher bid, the bid works in a 5% increment for every consecutive bid made by the user.



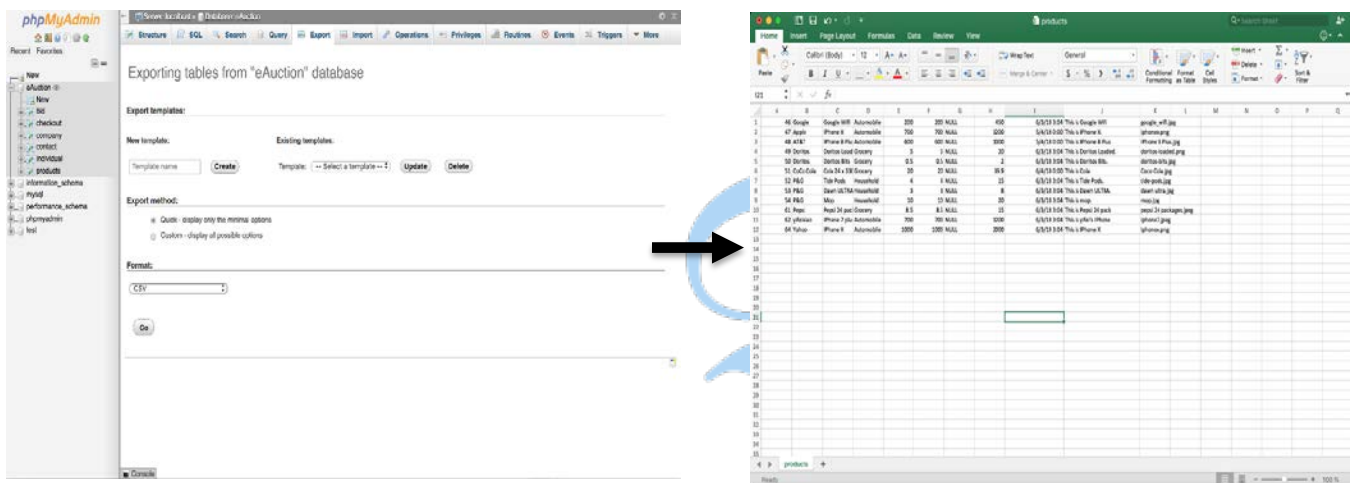
## 6. LeaveRatings: The user can leave ratings to items and other users.



7. **TerminateAuction:** This is executed at the end of each day to terminate all auctions that finished during the day. It does not use a web interface so a normal Java application or a C with embedded SQL application can be used. The website administrators can manually do it at the end of the day as well.

## 8. TeleMarketingReport

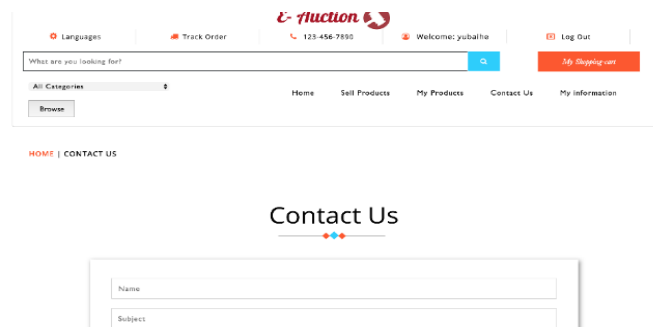
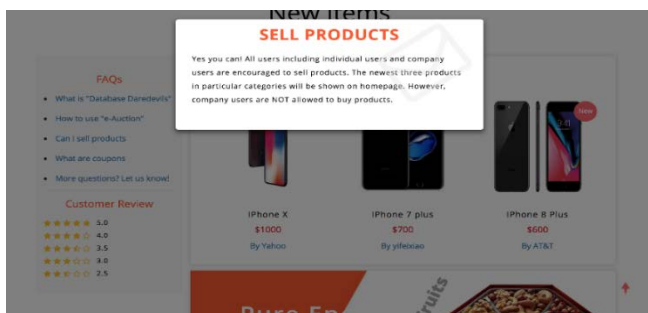
Everything on the e-auction website is connected with the database at the back-end. Phymyadmin (backend server) provides an interactive GUI which allows to generate the Data Analysis report as per the bi-weekly requirement of the telemarketers. Different formats including JSON, Excel, etc. are available for data storage and analysis.



## 9. New Additional Features

### a. FAQs

Frequently Asked Questions are questions that a user might have about the website or a certain product available during bidding and selling process. Although the user is supplied with many FAQs, still to provide



an benevolent customer service, users can click on ‘Contact Us’ button and submit the unanswered question to the website and the seller.

### b. Shopping Cart and Coupons

Once the bidding and selling process is complete, item goes to the shopping cart of the user. When there is an on-going coupon and promotion, the discount is applied to the final price of the items in the shopping cart before the checkout and delivery process

## Checkout


Your shopping cart contains: 1 product

Total Price: \$2000

Discount Rate: 10%

---

**Final Price: \$1800**

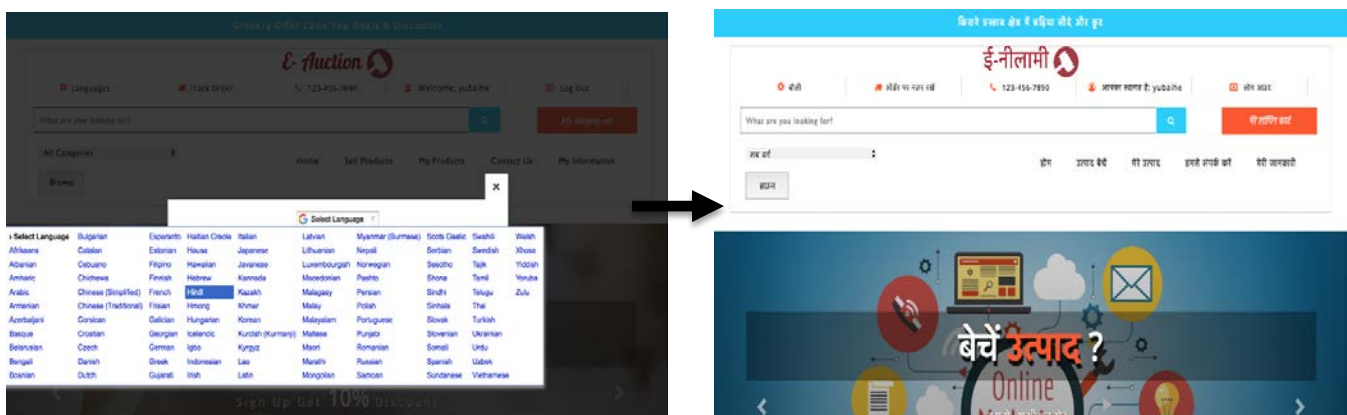
Product	Product Name	Price	Remove
	iPhone X	\$2000	<div style="background-color: #007bff; color: white; padding: 5px 10px; border-radius: 4px;">delete</div>

↑

Payment and Delivery

### c. Multiple Languages

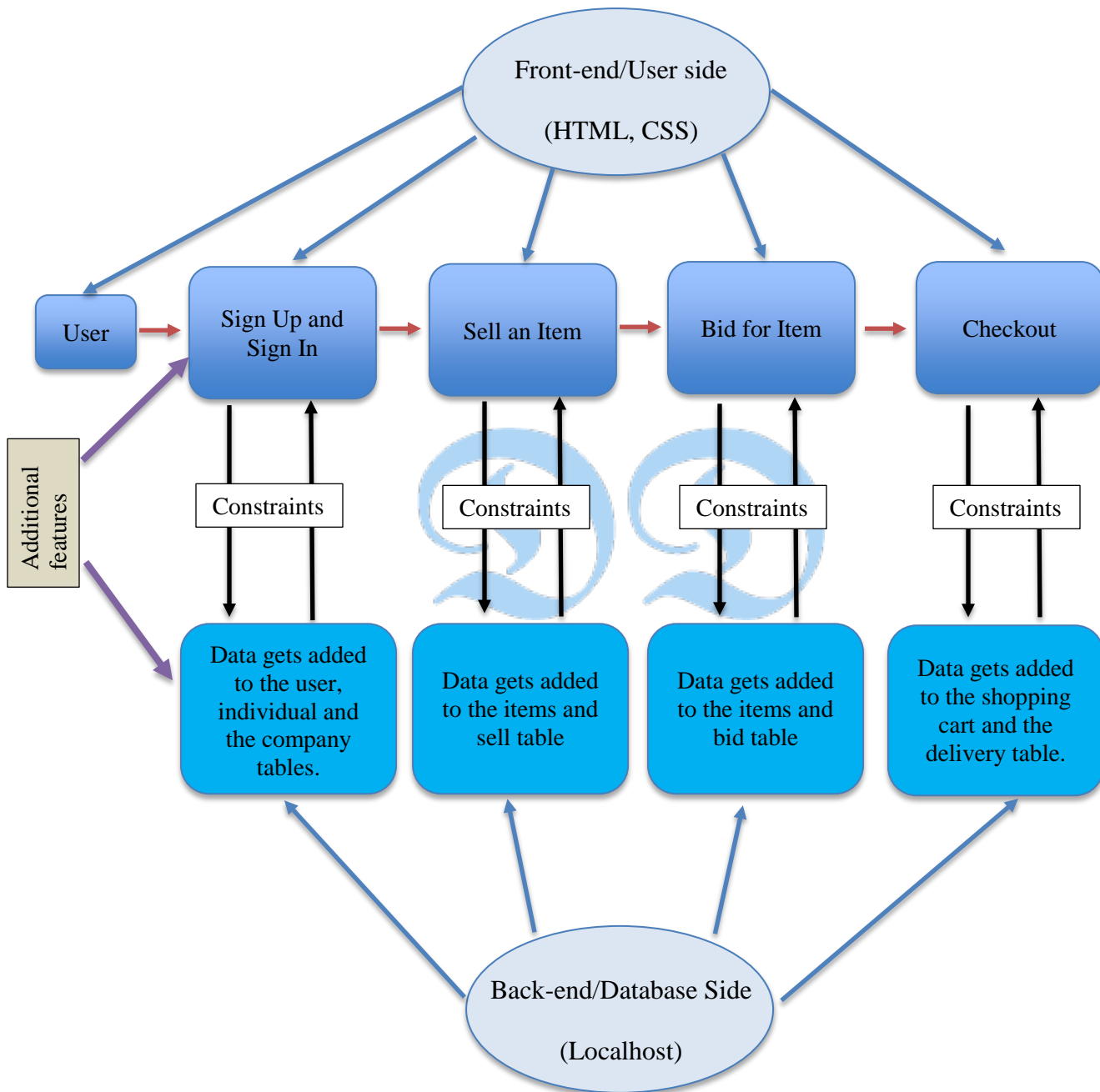
To make the website globally accessible, the website is viewable in more than 100 languages. Here, we have used the Google APIs which enables translation to multiple languages.





## 7. Complete System Framework

Below is an illustration of how the front-end interacts with the back-end in the system. The users interact with the front-end while the system stores the data in the back-end.



## 8. Reflection

The primary factor from Phase 1 to Phase 3 have been interactive and easy to use GUI for our users. In the project, we have made sure that the front-end steals the show being attractive and couples all the features promised during initial phases of the project.

We are happy to share with you that we were not able to satisfy our goals but also exceed them with promising results and quality. Since the journey for an ambitious project is never , we learnt wonderful lessons which has enhanced our knowledge in front-end, back-end, and middleware designs manifold.

- Communication

Since the team comprises of students with different majors, we used technology available for communication like GroupMe. GroupMe is available on both the PC and mobile and therefore, communicating and keeping up with the progress of the project was convenient.

- Collaboration

We used the very handy Google Drive for collaboration purposes instead of GitHub. The team found it just as easier to collaborate through Google Drive.

### 8.1 Analysis of Goals

The team has come a long way from the beginning of the semester with the help of the professor and the TAs. The team consists of a single Computer Science major who has taken higher levels of programming classes at Penn State. In fact, no one in the team had previous experience of Bootstrap, HTML, and PHP. However, with strong determination and a desire to excel, we were successfully able to create an easy to use website with all the features promised in the requirement analysis.

The only feature the team could not complete was the ‘delivery’ feature. To mark the good delivery experience for the user, we have planned to partner with FedEx and equivalent delivery service provider to generate unique order numbers for tracking the packages. This requirement is more towards the business side than the technical

one. Once a partnership deal is penned with a delivery service, it would be optimal to inculcate the feature. Since there are a lot of successful online e-auction website right now, such as Amazon, eBay, and so on, it will be hard for us to attract many users. So, one of the probable ideas we can implement in the future is to focus our market solely on the Penn State campus! Teachers and students will be our users. They can sell used textbooks, used Nittany notes, and used electronic device etc. This is extremely useful specially for graduating students as they can sell their used products on our website once they graduate. We will communicate with Penn State and the corresponding departments later in the future!

Overall, the team is proud of what we learnt and what we achieved in the semester in the form of collaborative process. We believe that e-Auction website has turned out to be great!

## **9. Conclusion**

Database Daredevils has summarized the requirement analysis, ER diagram, conceptual design and finally, the system prototype of the project through an interactive report and appendices. The team has focused on incorporating the modifications as suggested in the Phase 1 and Phase 2 for the betterment of Phase 3. In fact, we learnt about more technologies in Phase 3 and decided to use Bootstrap to enhance the user experience. We have displayed all the features of the website along with their constraints through the eyes of a user with the help of images, videos, and live demonstration. We hope that we are able to meet your expectations and satisfy all the requirements as outlined in the project description both in terms of quality and timeline. We appreciate you for taking your time to go through our report. Team Database Daredevils wishes you the best!

## Appendix A: Links for the Website Demo

Link for the YouTube Video: <https://www.youtube.com/watch?v=mVUkFELknKg>

Link for all the codes will be shared with you through google drive:

<https://drive.google.com/drive/folders/1uQbhK56-ILNL56bi9ahXMqkx3rbfqIf0?usp=sharing>

## Appendix B: 3rd Normalization Tables and SQL statements

The following tables are all in 3<sup>rd</sup> normal form.

### 1. FAQs Tables

<u>question_ID</u>	question	answer
<u>item_ID</u>	<u>question_ID</u>	

```
CREATE TABLE faq(  
  question_ID VARCHAR(20),  
  question VARCHAR(100),  
  answer VARCHAR(100),  
  PRIMARY KEY (question_ID);
```

```
CREATE TABLE faqlink(  
  item_ID VARCHAR(20),  
  question_ID VARCHAR(20),  
  PRIMARY KEY (item_ID, question_ID),  
  FOREIGN KEY (item_ID) REFERENCES items(item_ID)  
    ON DELETE CASCADE,  
  FOREIGN KEY (question_ID) REFERENCES faq(question_ID)  
    ON DELETE CASCADE);
```

### 2. Bidding Tables

<u>bid time</u>	bid_price
<u>item_ID</u>	<u>bid time</u>

```
CREATE TABLE bidlink(  
  item_ID VARCHAR(20),  
  bidtime DATETIME,  
  PRIMARY KEY (item_ID, bidtime),  
  FOREIGN KEY (item_ID) REFERENCES items(item_ID)  
    ON DELETE CASCADE,  
  FOREIGN KEY (bidtime) REFERENCES bidding(bidtime)  
    ON DELETE CASCADE);
```

```
Create TABLE bidding(  
  bidtime DATETIME,  
  bidprice VARCHAR(10),  
  PRIMARY KEY (bidtime));
```

### 3. Delivery Tables

<u>trans_ID</u>	deli_date	status
<u>cart_ID</u>	<u>trans_ID</u>	

```
CREATE TABLE paidelivery(
  trans_ID VARCHAR(20),
  delidate DATE,
  status VARCHAR(50),
  PRIMARY KEY (trans_ID));
```

```
CREATE TABLE translink(
  trans_ID VARCHAR(20),
  cart_ID VARCHAR(20),
  PRIMARY KEY (trans_ID, cart_ID),
  FOREIGN KEY (trans_ID) REFERENCES paidelivery(trans_ID)
    ON DELETE CASCADE,
  FOREIGN KEY (cart_ID) REFERENCES shoppingcart(cart_ID)
    ON DELETE CASCADE);
```

#### 4. Reports Tables

<u>tele_ID</u>	<u>individual_ID</u>
----------------	----------------------

```
CREATE TABLE reports(
  tele_ID VARCHAR(20),
  individual_ID VARCHAR(20),
  PRIMARY KEY (tele_ID, individual_ID),
  FOREIGN KEY (individual_ID) REFERENCES individual(individual_ID),
  FOREIGN KEY (tele_ID) REFERENCES telemarketers(tele_ID));
```

#### 5. Searching Tables

<u>individual_ID</u>	<u>cat_ID</u>
----------------------	---------------

```
CREATE TABLE searching(
  cat_ID VARCHAR(20),
  individual_ID VARCHAR(20),
  PRIMARY KEY (cat_ID, individual_ID),
  FOREIGN KEY (cat_ID) REFERENCES categories(cat_ID),
  FOREIGN KEY (individual_ID) REFERENCES individual(individual_ID));
```

#### 6. Browsing Tables

<u>individual_ID</u>	<u>cat_ID</u>
----------------------	---------------

```
CREATE TABLE browsing(
  cat_ID VARCHAR(20),
  individual_ID VARCHAR(20),
  PRIMARY KEY (cat_ID, individual_ID),
  FOREIGN KEY (cat_ID) REFERENCES categories(cat_ID),
  FOREIGN KEY (individual_ID) REFERENCES individual(individual_ID));
```

## 7. Selling Tables

user\_ID

item\_ID

```
CREATE TABLE selling(
  item_ID VARCHAR(20),
  user_ID VARCHAR(20),
  PRIMARY KEY (item_ID, user_ID),
  FOREIGN KEY (item_ID) REFERENCES items(item_ID),
  FOREIGN KEY (user_ID) references luserid(user_ID));
```

## 8. Buy Tables

individual\_ID

item\_ID

```
CREATE TABLE buy(
  item_ID VARCHAR(20),
  individual_ID VARCHAR(20),
  PRIMARY KEY(item_ID, individual_ID),
  FOREIGN KEY (item_ID) REFERENCES items(item_ID),
  FOREIGN KEY (individual_ID) REFERENCES individual(individual_ID));
```

## 9. Addto Tables

individual\_ID

cart\_ID

```
CREATE TABLE addto(
  cart_ID VARCHAR(20),
  individual_ID VARCHAR(20),
  PRIMARY KEY (cart_ID, individual_ID),
  FOREIGN KEY (cart_ID) REFERENCES shoppingcart(cart_ID),
  FOREIGN KEY (individual_ID) REFERENCES individual(individual_ID));
```

```
CREATE TABLE rate(
    rater_individual_ID VARCHAR(20),
    ratee_individual_ID VARCHAR(20),
    rating VARCHAR(5),
    PRIMARY KEY(rater_individual_ID, ratee_individual_ID),
    FOREIGN KEY (rater_individual_ID) REFERENCES
    individual(individual_ID),
    FOREIGN KEY (ratee_individual_ID) REFERENCES
    individual(individual_ID));
```

## 10. Rate Tables

<u>rater_individual_ID</u>	<u>individual_ID</u>
<u>ratee_individual_ID</u>	<u>individual_ID</u>

## 11. Auction\_statistics\_generate Tables

<u>stat_date</u>	total_trans
<u>item_ID</u>	<u>stat_date</u>

```
CREATE TABLE auction(
    statdate DATE,
    totaltrans VARCHAR(20),
    PRIMARY KEY (statdate));
```

```
CREATE TABLE auclink(
    item_ID VARCHAR(20),
    statdate DATE,
    PRIMARY KEY (item_ID, statdate),
    FOREIGN KEY (item_ID) REFERENCES items(item_ID),
    FOREIGN KEY (statdate) REFERENCES auction(statdate));
```

## 12. Categories Tables

<u>cat_ID</u>	<u>cat_name</u>
---------------	-----------------

```
CREATE TABLE categories(
    cat_ID VARCHAR(20),
    catname VARCHAR(50),
    PRIMARY KEY (cat_ID));
```

## 13. Subcategories Tables

<u>subcat_ID</u>	subcat_name
<u>cat_ID</u>	<u>subcat_ID</u>

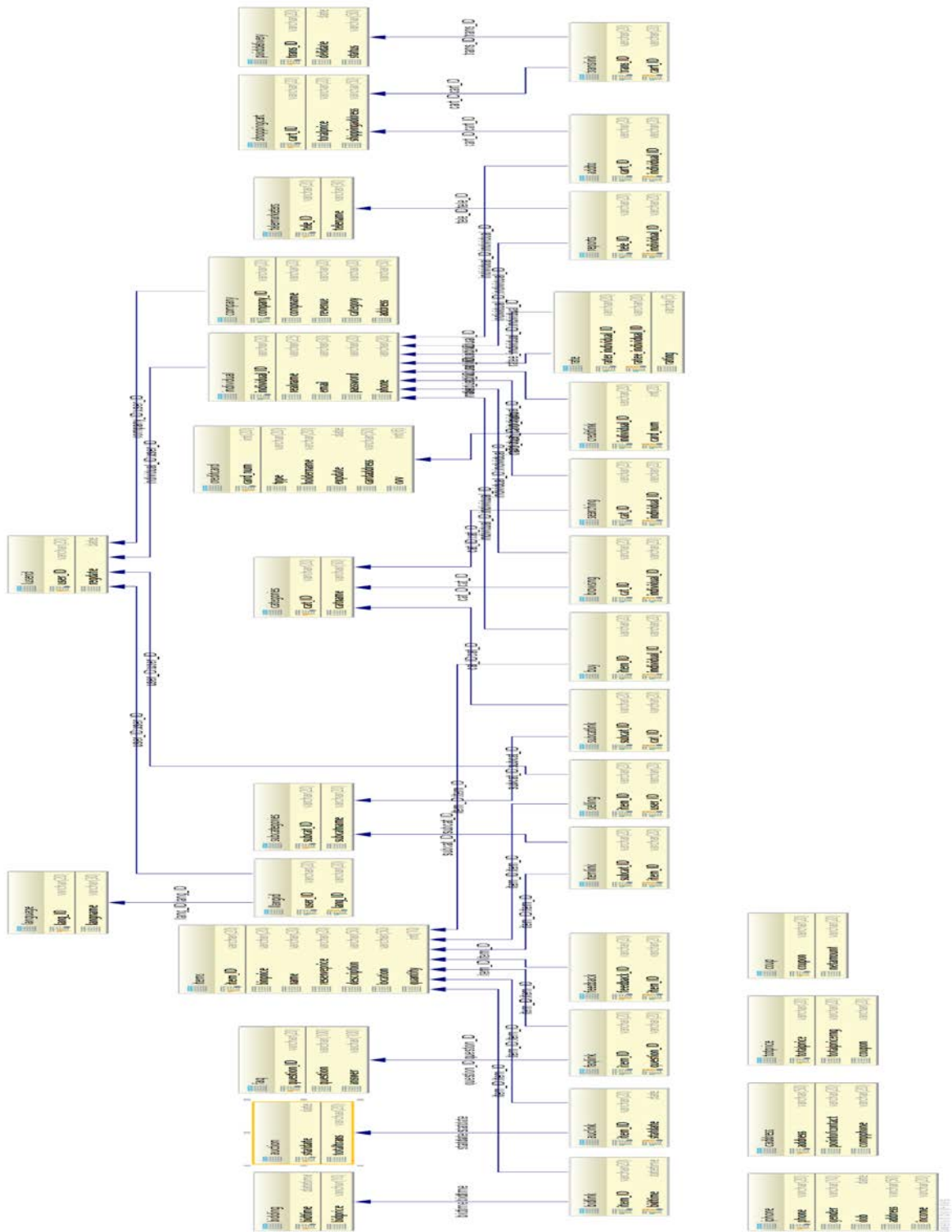
```
CREATE TABLE subcategories(
    subcat_ID VARCHAR(20),
    subcatname VARCHAR(20),
    PRIMARY KEY (subcat_ID));
```

```
CREATE TABLE subcatlink(  
  subcat_ID VARCHAR(20),  
  cat_ID VARCHAR(20),  
  PRIMARY KEY (subcat_ID, cat_ID),  
  FOREIGN KEY (subcat_ID) REFERENCES subcategories(subcat_ID),  
  FOREIGN KEY (cat_ID) REFERENCES categories(cat_ID));
```





## Appendix C: SQL tables generated by DataGrip



## Appendix D: Photos of the Databases

Table	Action	Rows	Type	Collation	Size	Overhead
addto	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
auclink	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
auction	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
bidding	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 Kib	-
bidlink	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	32 Kib	-
browsing	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
buy	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
caddress	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
categories	Browse Structure Search Insert Empty Drop	10	InnoDB	latin1_swedish_ci	16 Kib	-
company	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
creditcard	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
creditlink	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
faq	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
faqlink	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
feedback	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
individual	Browse Structure Search Insert Empty Drop	11	InnoDB	latin1_swedish_ci	16 Kib	-
iphone	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
itemlink	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
items	Browse Structure Search Insert Empty Drop	9	InnoDB	latin1_swedish_ci	16 Kib	-
language	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
llangid	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
luserid	Browse Structure Search Insert Empty Drop	11	InnoDB	latin1_swedish_ci	16 Kib	-
netamt	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 Kib	-
paiddelivery	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 Kib	-
rate	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	32 Kib	-
reports	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
searching	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
selling	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	32 Kib	-
Console Inqcart	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 Kib	-

Device Preview localhost / localhost / eAuction x

localhost/phpmyadmin/db\_structure.php?server=1&db=eAuction+Phase2&token=278a26e1586082f3b1a5e66110b77d38

Server: localhost Database: eAuction Phase2

Structure SQL Search Query Export Import Operations Privileges Routines Events Triggers More

language language llangid luserid netamt paiddelivery rate reports searching selling shoppingcart subcategories subcatlink telemarketers totpice

35 tables Sum 67 InnoDB latin1\_swedish\_ci 816 Kib 0 B

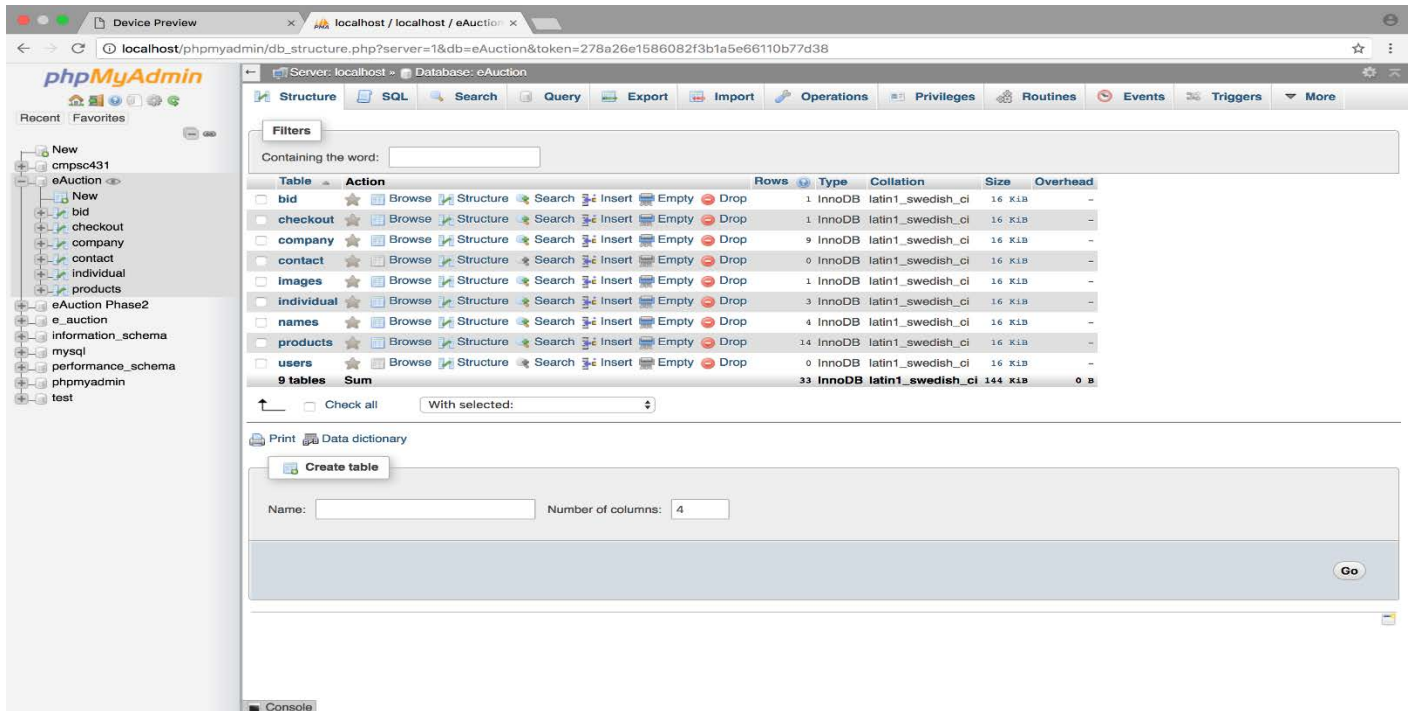
Check all With selected:

Create table

Name: Number of columns: 4

Go

Console



## Appendix E: Bibliography for technology survey

- [1] Richards, J. (2015, June 24). Comparing MongoDB with MySQL. Retrieved from <https://www.analyticbridge.datasciencecentral.com/profiles/blogs/comparing-mongodb-with-mysql>
- [2] Sarig, M. (2017, November 27). MongoDB vs MySQL: the differences explained. Retrieved from <https://blog.panoply.io/mongodb-and-mysql>
- [3] Karwin B. (2016, October 24). The difference between MySQL and MariaDB. Retrieved from <https://www.quora.com/What-is-the-difference-between-MySQL-and-MariaDB>
- [4] Sarig, M. (2017, March 28). MariaDB vs MySQL: In-Depth Comparison. Retrieved from <https://blog.panoply.io/a-comparative-vmariadb-vs-mysql>
- [5] Johnson H. (2016, December 13). How do MySQL and Microsoft Access differ. Retrieved from <https://www.quora.com/How-do-MySQL-and-Microsoft-Access-differ>
- [6] S. (2013, December 13). MySQL single query performance - the truth! Retrieved from <http://www.fromdual.com/mysql-single-query-performance-the-truth>



## Appendix F: Progress Reports

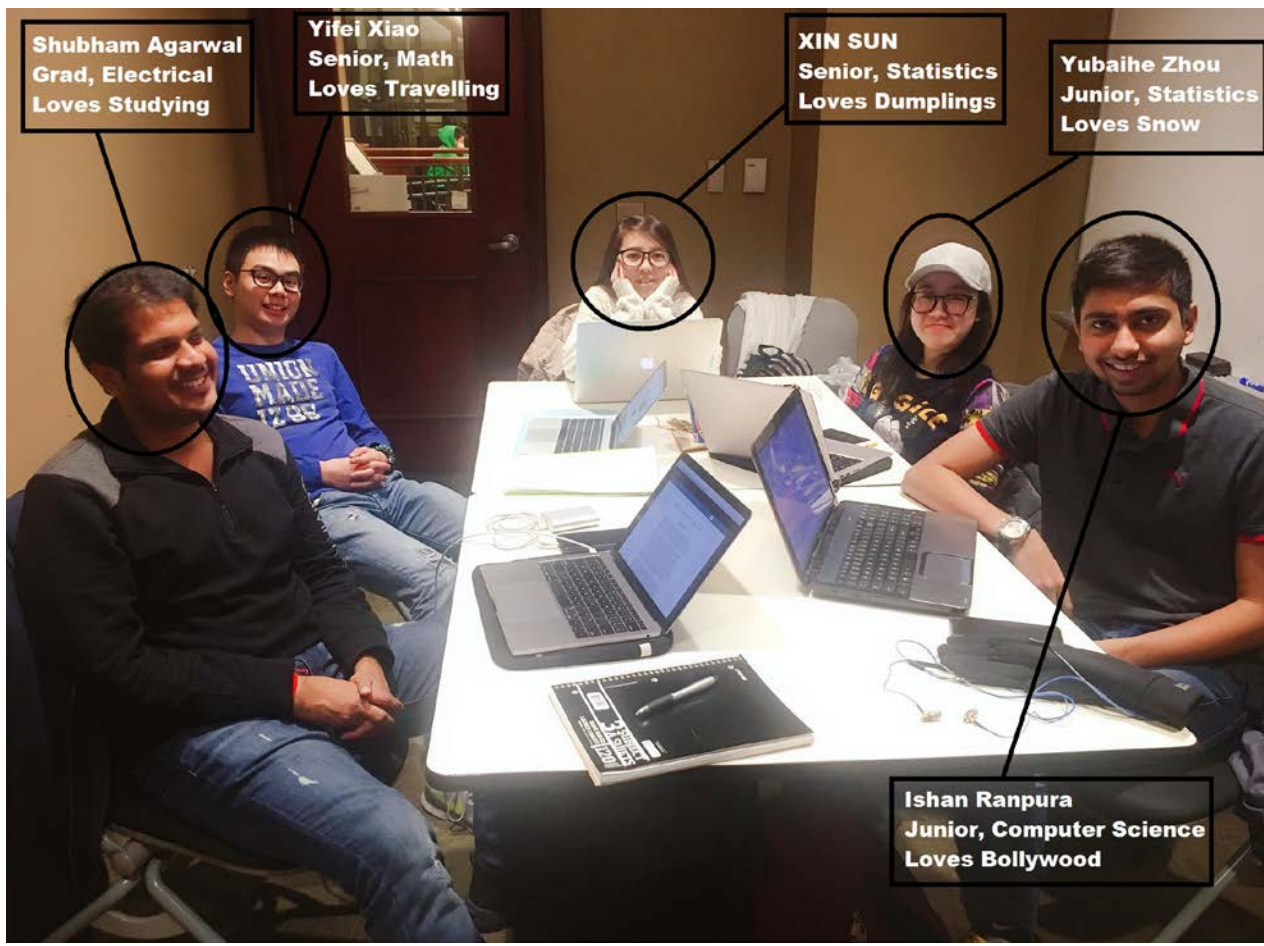
**Team Name:** Database Daredevils

**Members:** Ishan Ranpura, Xin Sun, Yifei Xiao, Yubaihe Zhou, Shubham Agarwal

**Reporting Period:** January 17, 2018 – April 27, 2018

### Progress Report

#### Meet the Team:



## Group Report

<u>Group members who attended the meeting</u>	<u>Description</u>	<u>Date/Time</u>	<u>Duration (Hours)</u>
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou	This was the first time everyone was meeting each other. We introduced ourselves, talked about our strengths and what we could bring into the team. We went over project guidelines and discussed what we needed to do going forward. We also discussed what unique features we could add into our website. Next time we are going to clarify them and define how we are gonna operate them.	1/17/18 6:30 pm - 7:45 pm	1.25 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the second meeting. In this meeting, we completed the progress report, discussed further about the additional features we were going to add to the website. We brainstormed ideas on how to make the relationship-entity model diagram. We went over all the features provided by the project description, including additional features, drew a rough relationship-entity diagram from it and wrote questions we had for the professor that we will ask him.	1/21/18 3:00 pm - 6:00 pm	3 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the third meeting. In this meeting, we started drawing the ER diagram, using Lucidchart on the laptop. We tried our best to include every entity in the ER diagram, find the relationships between each entity set and their corresponding attributes, also including our new features. We later made the main structure for the project report. And next time we will start writing the project report.	1/28/18 5:00 pm - 9:00 pm	4 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the fourth meeting. In this meeting, we started writing the introduction and explanations for all the features in the requirement analysis.	2/1/18 6:00 pm - 9:15 pm	3.25 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou	This was the fifth meeting. In this meeting, we finished 90% of the whole report, including finishing description of ER diagram and modifying requirement analysis. Each team member	2/3/18 7:00 pm - 11:45 pm	4.75 hours

Shubham Agarwal	was assigned to do part of work, and we double checked the whole progress to make sure everything looked fine.		
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the last meeting. In this meeting, we were finishing writing the conclusion of the project report, and deciding what we going to include for the appendix. All of us went through the whole project report and proofread the report.	2/4/18 3:00 pm – 6:00 pm	3 hours

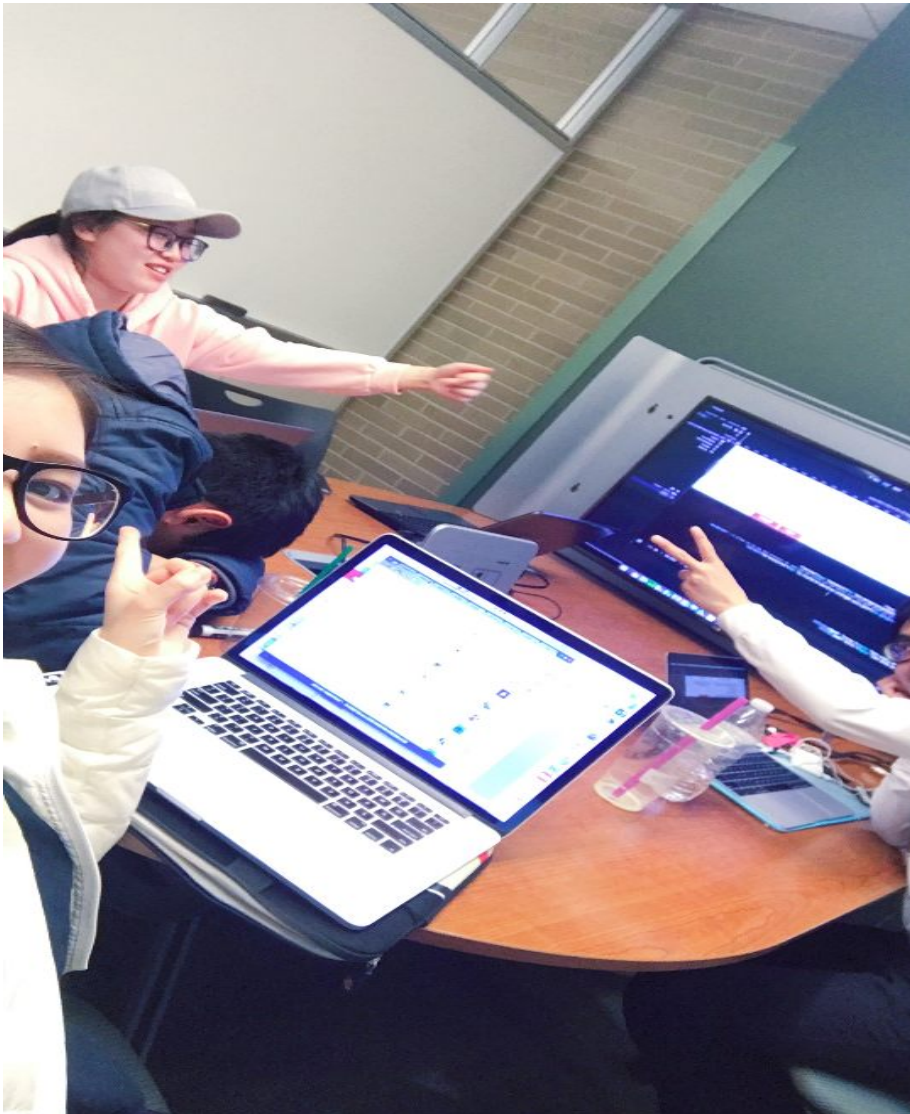
<u>Group members who attended the meeting</u>	<u>Description</u>	<u>Date/Time</u>	<u>Duration (Hours)</u>
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the first meeting after midterm and we started working on project phase 2. In this meeting we went over the phase 2 description and discussed what we should do going forward. We also discussed the feedback from phase 1 report and our ER Diagram since phase 2 is the extension and application of the ideas in phase 1. To make things easier, we wrote SQL statements to create the corresponding relations based on the ER Diagram. Next time we are going to clarify them and define the relational schemas.	2/18/18 Sun 4:00 pm - 7:00 pm	3 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the second meeting. During this meeting, we found a lot of places that can be improved in our relational model. We went over each entities step by step and added some new relations. Then, we transformed ER components (i.e., entity set, relationship set, constraints, etc) to relational schema and wrote questions we had such as the primary keys of weak entities for the professor and would bring it to the office hour. We set up the next meeting time and we will try to finish the definition of schemas and start to do the normalization.	2/22/18 Thu 6:00 pm - 10:00 pm	4 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the third meeting. In this meeting, we finally finished the part of schema definition and started working on schema normalization in order to reduce redundancy to at least the 3rd Normal Form. We specified functional dependencies and refined the schema through	2/24/18 Sat 6:00 pm - 11:30 pm	5.5 hours

	normalization. We also made the main structure for the 2nd project report. And next time we will finish schema normalization start writing the project report.		
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the fourth meeting. In this meeting, we focused on schema normalization and finished 90% of this part, but there was always room for improvement. Our team raised plenty of questions and we would bring it to the class on Monday. We also decided to use DataGrip to execute queries.	2/25/18 Sun 3:00 pm - 9:00 pm	6 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the fifth meeting. In this meeting, we finished schema normalization and drew the relational schema diagram to provide an overview of the entire database. After that, we started to write the project report. We also created all of the necessary tables defining primary keys and foreign keys as well as the sample database (including users, items, and categories) by using Datagrip. Each team member was assigned to do part of work, and we double checked the whole progress to make sure everything looked fine. Next time we will finish the 1st draft of the report and go over it together.	2/26/18 Mon 5:00 pm - 10:00 pm	5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the sixth meeting. In this meeting, we were finishing writing the 1st draft of the project report, and deciding what we going to include for the appendix. All of us went through the whole project report and pointed out the uncertain issues in each part. We also started working on our presentation slides. Next time we have to finish the final version of the report and hopefully practice a little for our presentation on Friday.	2/28/18 Wed 5:00 pm - 11:30 pm	6.5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was our last meeting. This was the last day before the presentation, so we decided to meet one last time and do some practice. In this meeting, we were finishing writing the project report, and discussed how to present to the class. All of us went through the whole project report and proofread the report.	3/01/18 Thu 3:30 pm - 11:30 pm	8 hours

Group members who attended the meeting	Description	Date/Time	Duration (Hours)
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the first meeting after midterm 2 and we started working on the phase 3 of the project. In this meeting we went over the phase 3 description and discussed what we should do. We also read through the feedback from phase 2 report and discussed about what we should improve on our phase 3 report. After we completed reading the phase 3 description, we realized that we need to do some research on what software we will use to create our website i.e. check the level of difficulties to connect the database and check what kind of features they have.	03/16/18 Friday 1:30 pm – 3:30 pm	2 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was our second meeting after midterm 2, and we discussed what kind of software we found that were good to use. We first all agreed to use Adobe Dreamweaver to create our HTML pages and CSS. After we decided to use Dreamweaver, we discussed how many website pages we will create. And then we distributed pages to each team member and let them work on those at home.	03/22/18 Thursday 6:30 pm – 9:30 pm	3 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was our third meeting after midterm 2, and we gathered everybody's website pages and watched some videos on how to connect our website with our local database. And then we worked on the CSS.	04/01/18 Sunday 3:00 pm – 6:00 pm	3 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the fourth meeting. We found that we could use Bootstrap to create the CSS for the website. Bootstrap provided us framework to create a really good UI. So, our team started watching tutorials on YouTube.	04/02/18 Mon 5:00 pm - 7:30 pm	2.5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was our fifth meeting. The team started coding the HTML pages again to make the UI better this time using Bootstrap. Meanwhile, on the PHP front, we were able to connect the database with the login and sign up page.	04/03/18 Tue 3:30 pm - 06:30 pm	3 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the sixth meeting for phase 3. We made a list of things to-do for the project and started working on the HTML and the PHP part of the project. Everything was done simultaneously and we started picking out tasks from the list and started completing them one-by-one.	04/04/18 Wed 4:00 pm - 8:00 pm	4 hours



Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the seventh meeting. We worked on the HTML and the PHP parts more and completed some work on the pages.	04/07/18 Sat 6:00 pm - 10:00 pm	4 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the eighth meeting. Like the previous meeting, we were just going down the list and kept on completing the task.	04/08/18 Sun 6:00 pm - 9:30 pm	3.5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the ninth meeting. We still had more than 70% of the project left with not enough time left to do. So, we decided to increase the team meeting time to give more time to the project.	04/12/18 Thu 6:00 pm - 10:30 pm	4.5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the tenth meeting. A little bit of panic had started setting in around this time. We had a lot of things left to do but did not want to compromise on the quality of the project just to complete all the features.	04/15/18 Sun 5:00 pm - 10:00 pm	5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the eleventh meeting. We knew that we were racing against time at this point to complete the project. We also made a structure for the final report in this meeting.	04/19/18 Thu 5:00 pm - 9:30 pm	4.5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the twelfth meeting. We worked more on the HTML, PHP and the final report in this meeting.	4/22/18 Sun 3:30 pm - 8:30 pm	5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the thirteenth meeting. We worked on everything we had been working on before. Additionally, we allotted everyone parts for the presentation.	4/23/18 Sun 3:30 pm - 8:30 pm	5 hours
Ishan Ranpura Xin Sun Yifei Xiao Yubaihe Zhou Shubham Agarwal	This was the fourteenth meeting in Phase 3. This was the last day before the presentation, so we decided to finalize everything for the presentation and the demonstration. We test run everything on the website, made a list of possible questions and prepared answers for them. We practiced our presentation for the project as well.	4/25/18 Sun 7:30 pm - 11:30 pm	4 hours



*Figure.1 Team's reaction after reading the project description*



Figure 2: Group discussing strategies.



Figure.3 Final meeting

Name	Contribution
Xin Sun	The team dedicated more than 50 hours for this phase and contributed equally to everything. We provided ideas, talked about them, and rejected or accepted some of ideas. All of us provided a lot of important contributions for the parts of the project. We fought and almost cried during the completion of the project and everyone on the team hopes that to get a reflection of their effort on the grades for this report.
Yubaihe Zhou	
Yifei Xiao	
Shubham Agarwal	
Ishan Ranpura	

Group Sign:

Ishan Ranpura:

Xin Sun:

Yifei Xiao:

Yubaihe Zhou:

Shubham Agarwal:

