Project 1 Milestone 1

1. Project details: Name of your project, your team, and team members. Please include Team ID and Member ID from the google sheet.

Name of Project: Online Video Game Store
Team: Make Database Great Again (14)
Team members: Thanh Kha (70), Yutong He(89), Zetian Xiao (90), Linxiao Bai (40)

2. Problem Statement: Describe the problem that your proposed database system will solve. Why do you need a database instead of an excel file?

The data we need to store will contain information of everything that an online shopping interface for users' and an employee's information would need to maintain their accounts and store products. Our data contains varieties of different products; such as video games, consoles, discs games, accessories, users, and many other information. We need a database in order to update the availability of items (stock), update the price of products, create functional trigger reward programs (such as points for every dollar spent), etc.

We would want to use a database instead of an excel because a database features more tools to manage the information stored in the database. Compared to Excel, the database system provides a faster querying, updating, inserting, and deleting of data. The features of the database also allows us to place constraints on specific data to keep integrity and consistency. It will also provide flexibility based on the privilege levels of certain users that can only access certain information in the database, making it more secured:

- Users will be able access the data of what products they can purchase, cart they can add to and remove from, and manage their own user information that includes shipping address, payment methods, and more.
- Employees are given the privilege to manage and update the products, customers, and store information.
- Managers would also be given the higher privilege to manage employment information such as basic information of salary and leave balance.

A database is also more resilient than excel in terms of backup mechanisms and management of larger data. Databases features capabilities of keeping data more organized and easier to maintain through relations. The relationships of different data (tables) will represent one another in order to provide only the relevant data needed at the time.

3. Target user: Who will use your database? Who will administer the database? You are encouraged to give a real-life scenario.

Store Manager:

The database administrator. The store manager will be responsible for authorizing the access to the database, monitoring the use of database, providing help if other user need, and will be the primary contact when external resources are needed.

Note that the store manager is the only user who can add and change employees' information.

Customer:

These users will be able to see and change their own information and only check details of games, publisher, and their own transaction.

Customer Service and Sell Force:

They can check but not change their own information, all customers' personal and transaction information, and other employee's' name, department, phone number. They will also be able to see but not modify the product and publisher information.

Purchasing Personnel:

Aside from the same access authority as customer service and sell force, these users can also add new data tuple into and update product and publisher relations.

4. List of Relations: Identify at least 4 relations(tables) that you need to maintain for this database. Provide schema for each of these relations.

Customers(id: int, name: varchar(20), phone: int, email: varchar(20), address: varchar(30), level: int, points: int, username: varchar(20), password: varchar(20), privilege.lvl: int)

id->unique id for each customer
(name,phone,email,address): basic information about the customer
(username,password): For customer to login (Username is unique string)
level: integer that indicates the level of the customer. Each level is corresponding to a different discount.
(points): every dollar the customer spend in the store can reward them 10 points. After reaching a certain number of points, the customer can get to a new level with higher discount.
Customer relation: store information about the customer
(privilege.lvl) a role indicator of user's privilege to access functionality of the system.

Employees(id: int, name: varchar(20), salary: float, Dept.No: int, phone: int)

id -> unique id for each employee
(name,salary,phone) -> basic information about the employee
Dept.No -> the id of which department the employee works for
Employee relation: store information about the employee

Departments(Dept.No: int, name: varchar(20), phone: int)

Dept.No-> unique id for each department
(name,phone)-> basic information about the department
Department relation: store department detail

Product(id: int, Com.No: int, price: money, stock: int, sales: int, rating: float)
id-> unique id for each product
Com.No-> the id for the manufacturer company for the product
(price,stock,sales)-> describes the price, remains in stock and total sales
rating-> customers' rating of the product, calculated by taking the mean of the ratings in all transactions containing the product
Product relation: store the information

Game(id: int, name: varchar(20), platform: varchar(20), genre varchar(20))
id->unique id for each game
(name,platform,genre)-> describes name of the game, which platform it can run, and which genre it belongs to(action, horror.etc)
Game relation: Store the details about the game.

Console(id: int, name: varchar(20), platform: varchar(20), color: varchar(20), memory: varchar(10))
id->unique id for each console
(name,platform,color,memory)-> describes the name of the Console, which platform it can support, and its color and memory
Console relation: store the detail about the console

Accessory(id: int, name: varchar(20), category: varchar(20))
id->unique id for each accessory
(name, category)->describes the name and category of the accessory(Gamepad, hard-drive .etc)
Accessory relation: store the detail about the Accessory

Company(id: int, name: varchar(20), country: varchar(20))
id->unique id for each company
(name,country)->describes the name of the Company and which country it belongs to.
Company relation: store the details about the Company

Transaction(Tid: int, CusID: int, ProID: int, rating: float)
Tid-> unique id for transaction
(CusID, ProID) -> describes which customer buy what product.
Rating: customer's rating for the transaction
Transaction relation: store the detail about the transaction customers make.

5. Web-interface: Provide rough sketches of the web-interfaces that you will design. Web interfaces are mainly 'form' where a user can provide input, and based on the input, the tables are updated and/or query results are produced. This is the user interface (UI) of your project.

Customer Interface:

Cart

Xbox One  **Unit Price**: $300.00 **Qty**: 2 **Total** $600.00

Quantity 2
Current: 2

Back Next

Checkout

Shipping Address

Street Address

Street Address Line 2

City

State / Province

Postal / Zip Code

Please Select
Country

Shipping Method

- Fedex
- Fedex Express
- U.S Priority

Payment Method

Total: $600.00 USD

Cancel Submit

Account Recovery

E-mail

Back Submit

Account Settings

John Smith

First Name Last Name

JSmith@yahoo.com

585-275-1111

Area Code - Phone Number

21 Elmwood Ave,
Rochester,
NY 14620

Street Address

Street Address Line 2

City State / Province

Postal / Zip Code Please Select Country

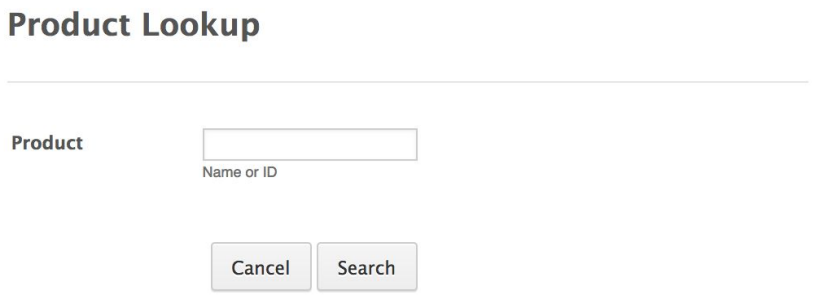
Cancel Submit

Order History

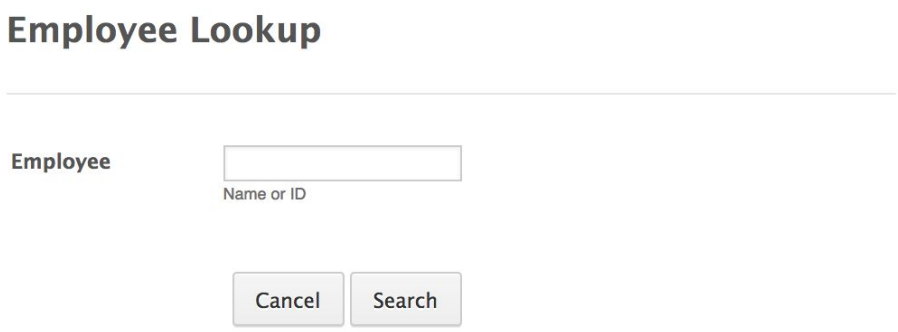
Date	Order No	Item	Tracking Number	Progress
2/4/2017	1234567	Xbox One		Processing

Cancel Order

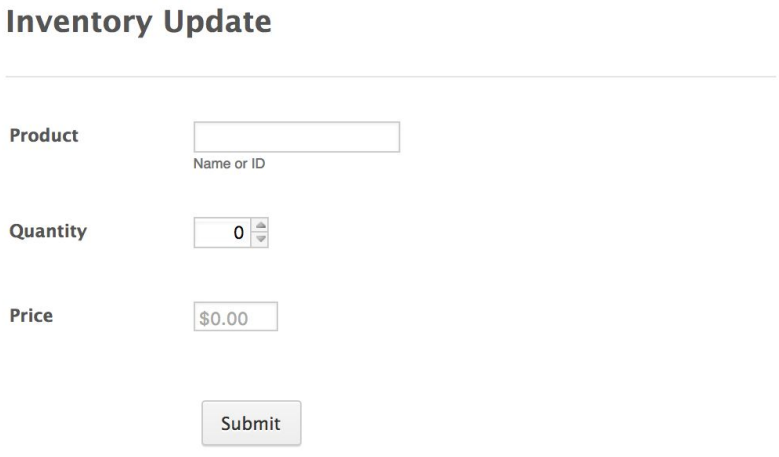
Date	Order No	Item	Tracking Number	Progress
1/27/2017	4583837	Wii	Mkj4N8d99qndC	Shipped



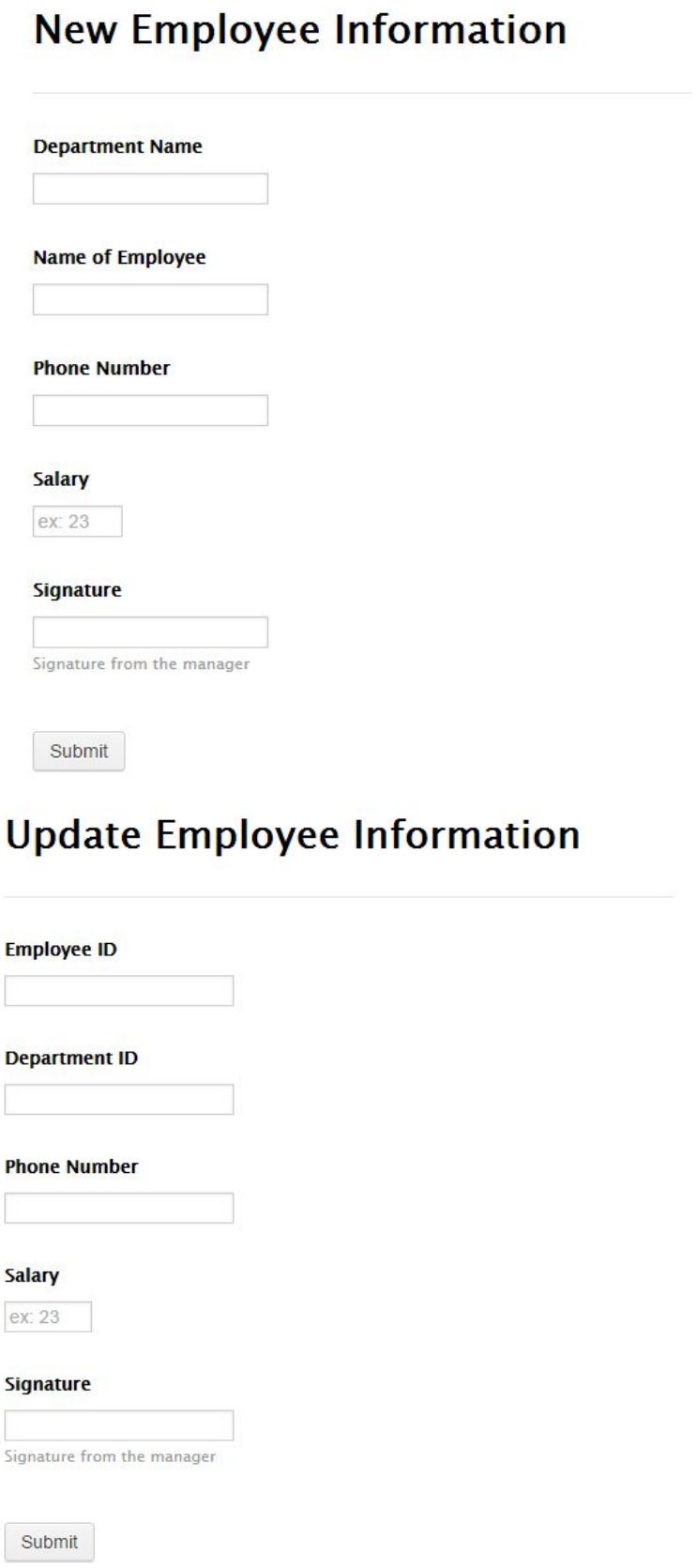
Employee Interface:



Stock Interface:



Manager Interface:



6. Data: How will you populate your relations? You can get the data from external sources or you can create your own data.

We are going to combine the external data and our own data to build our database.
Creating our own data:

1. When hiring employees, the store manager would fill out the employee table to insert a new tuple into employee relation.
2. Store managers can update data in employee relation including salary, phone number, and addresses through the update employee information form.
3. When the customers register an account at our online store, they will fill in the signup form to insert a new tuple into customer relation.
4. When buying products, the customers will fill in the transaction tables (cart and checkout) to insert new tuples into the transaction relation.
5. When purchase department purchases the new products into the stock, purchasing personnels will either add the new product and company to the database or update the stock of the existing game.

External Sources:

Because we would like to use the real world games as our product, we are going to get the product data(like publisher , Category, name) from external source such as official webpage.