Class: <CMIS485 6380 Web Database Development (2152) >  
Student Name: David Walters  
Instructor: <Dr. Alla Webb>  
Assignment #: Final Project  
Description: <This project consists of an extensive web application that allows the create, read, update and delete of records in an Oracle Database, written in ASP.NET with C#.>  
Due Date :<3/8/2015>  
I pledge that I have completed the programming assignment independently.  
I have not copied the code from a student or any source.  
I have not given my code to any student.  
Sign here: <David Walters>

**Set Up:**To run the application, simply navigate to URL on the Virtual Machine.

To open the application in Microsoft Visual Studio, extract the .Zip folder to a location of your choosing, select Open Webpage from the Visual Studio window, and then choose the extracted folder. Please note you will need to change the ConnectionString parameters to connect to any local database running on your machine.

**Technology:**The web application is written in HTML 5 with ASP.NET and C#. The front end user application is styled and created using Bootstrap styling with some additional JavaScript features. The back end is created using Oracle 11g Database Management System. The application was managed and written in the Visual Studio 2013 IDE.

**Security:**The application is inherently secure because it harnesses ASP.NET’s security features. ASP.NET, particularly with Visual Studio 2013, uses safe parameters in it’s SQL database calls. The IIS server will interpret all of the SQL commands on the server before outputting the static HTML file, thus the user will never see what is actually called the SQL database, only a plain HTML representation of the data.

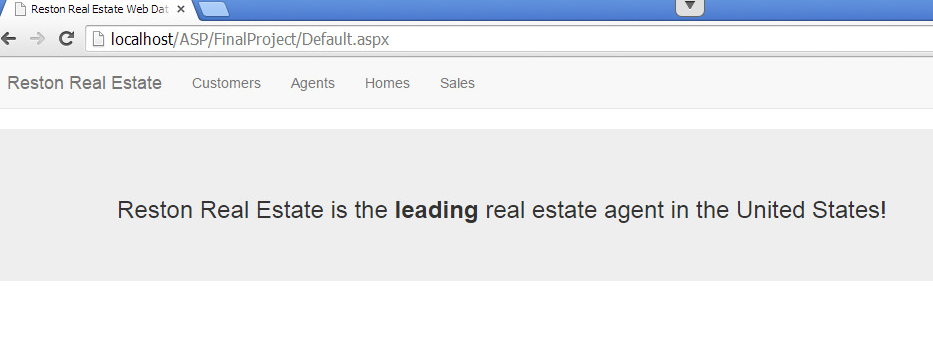
In addition to ASP.NET’s inherent security, I chose to validate the user inputted data on the server side rather than the client side. The built in ASP.NET validators will handle all input on the client side using JavaScript. This makes the user experience smoother, but also opens up for a skilled person to circumvent the validation. By validating the data on the server-side using CustomValidators, the client will never have an opportunity to see the code that is used to validate their input, and thus will have a serious disadvantage at ‘tricking’ the system.

**Lessons Learned and Reflection:**The most valuable lesson I learned from this project is the reusability of code. I found that since many of the pages, including Homes.aspx, Sales.aspx, Customers.aspx, and Agents.aspx, had the same functionality. Reusing the code across all the pages saved me hours of work and I found a standard way of adapting the code to fit each scenario.

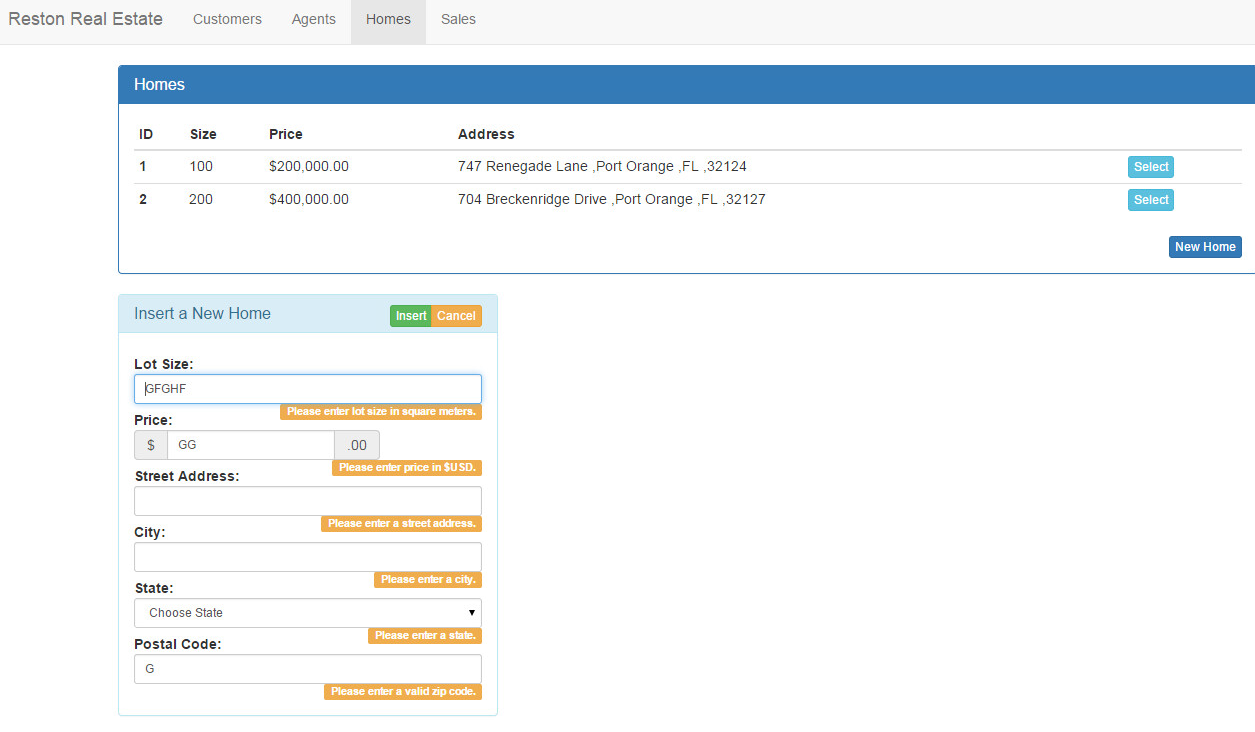
In the future, I would learn to do more extensive validation on the server side and work on data manipulation from the server to display to the user. I would harness more of C#’s object oriented power and not just use basic ‘CodeBehind’ techniques. I can see how ASP.NET with C# can be extended to power large scale applications.

I chose to work in ASP.NET because I found Visual Studio to be the easiest to work with. With GridView and FormView functionality built in with minimal use for code, I was able to focus on the bigger picture. I found implementing the same functionality in Java Servlets and PHP to be more time consuming, however I do appreciate how low level one can get with Java EE. I have experience developing in Java EE, so another advantage of choosing ASP.NET was for exposure and experience.

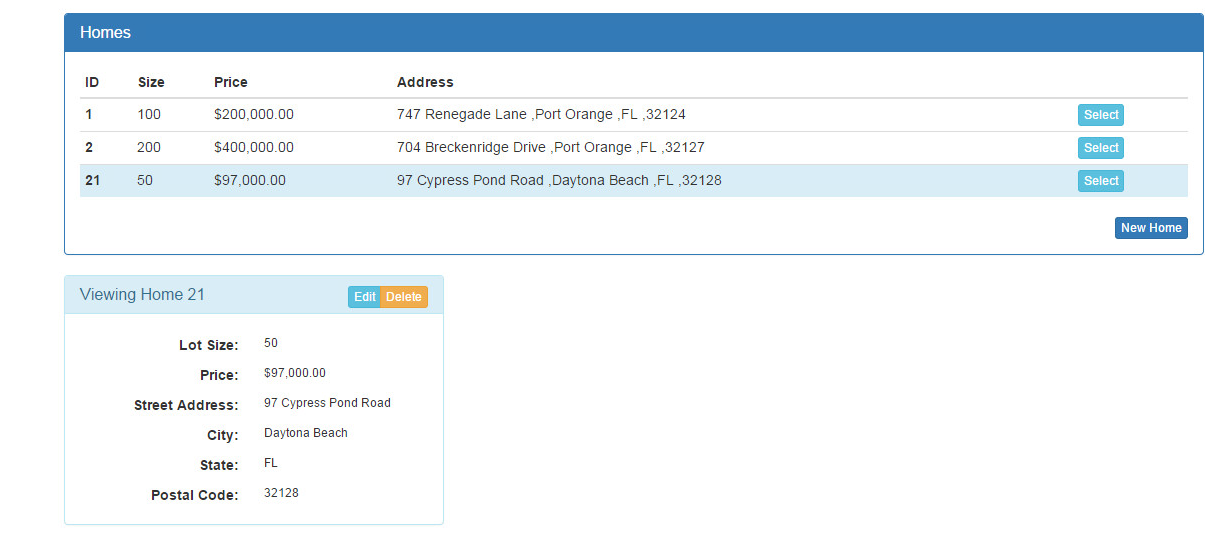
**Testing and Proof of Functionality Running on VM:**

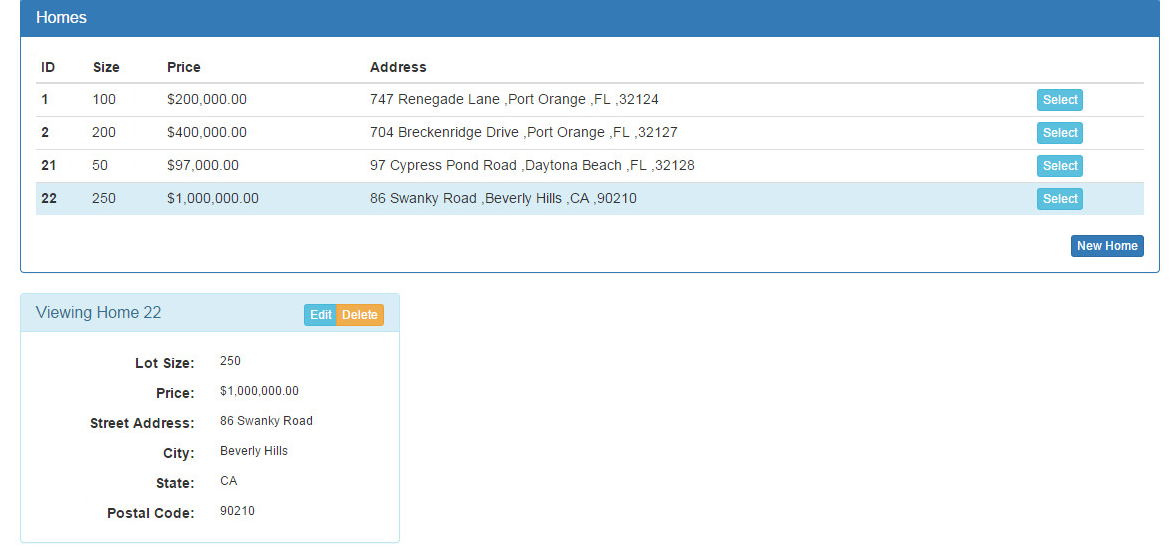
****

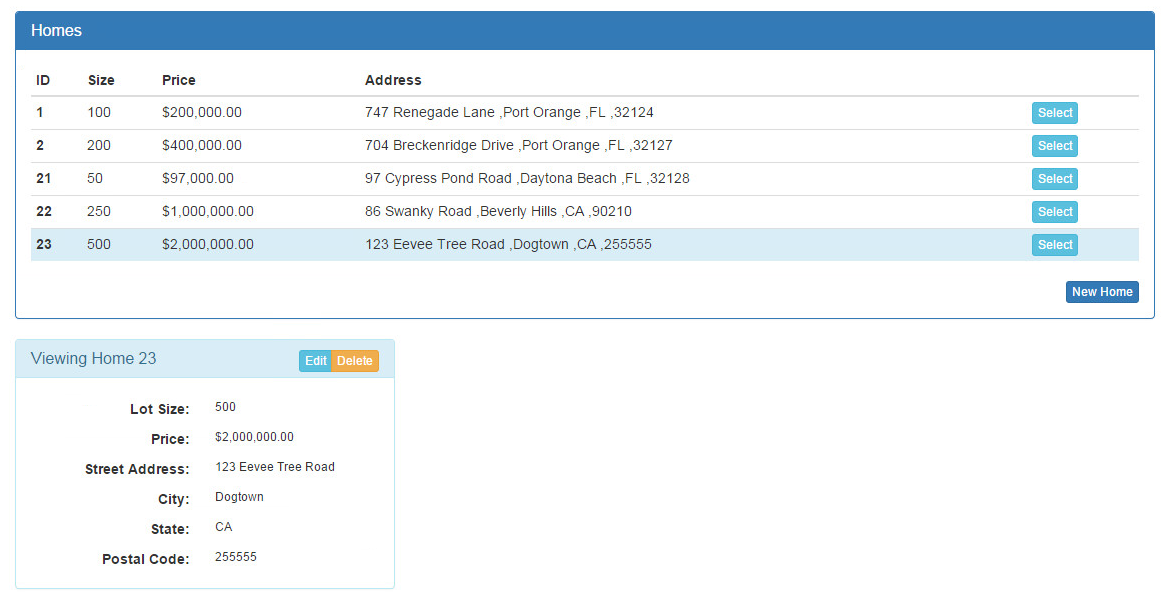
**Add three homes (note bad data for validation testing):**

****

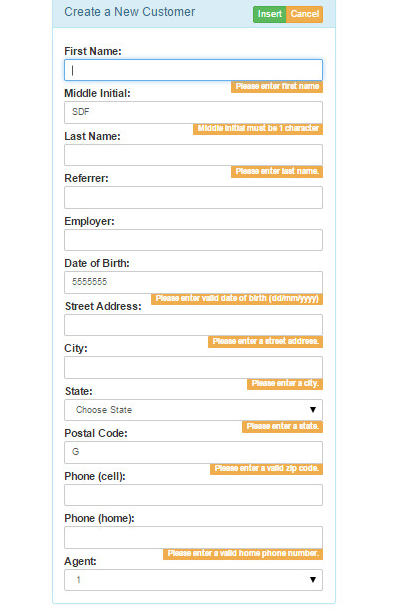
**With good data:  
(1)**

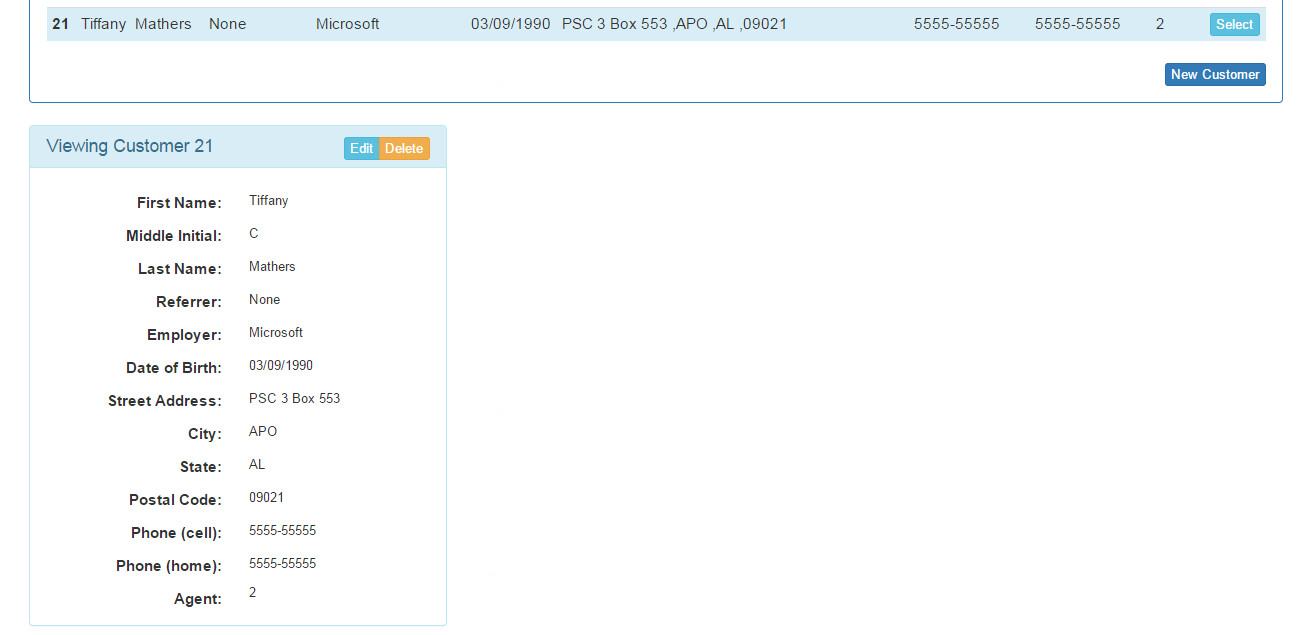
****

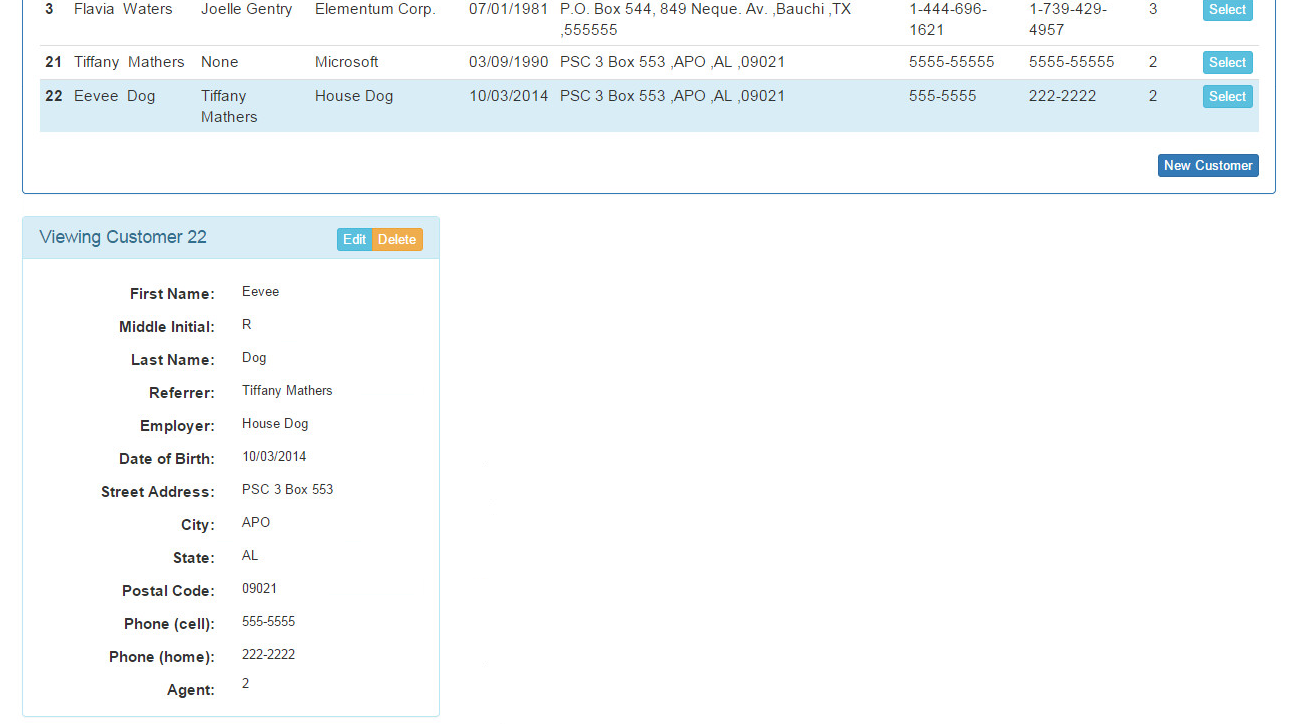
**(2)  
**

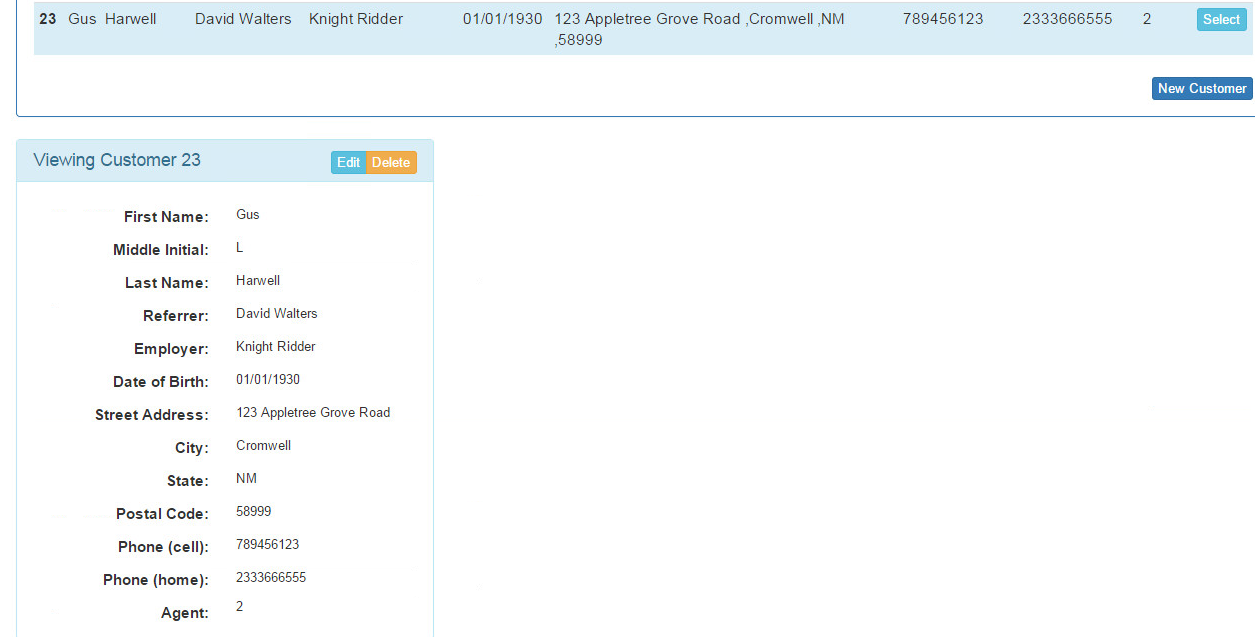
**(3)  
**

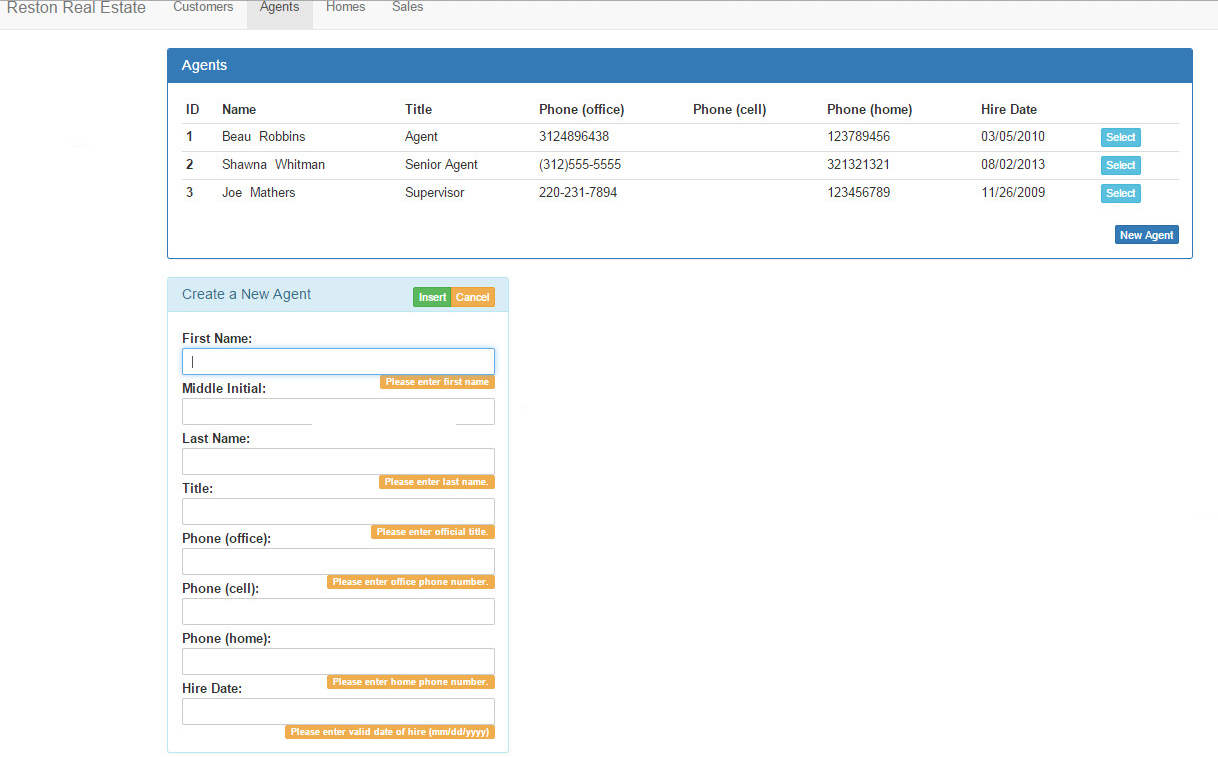
**Add 3 customers (note bad data for validation testing):**

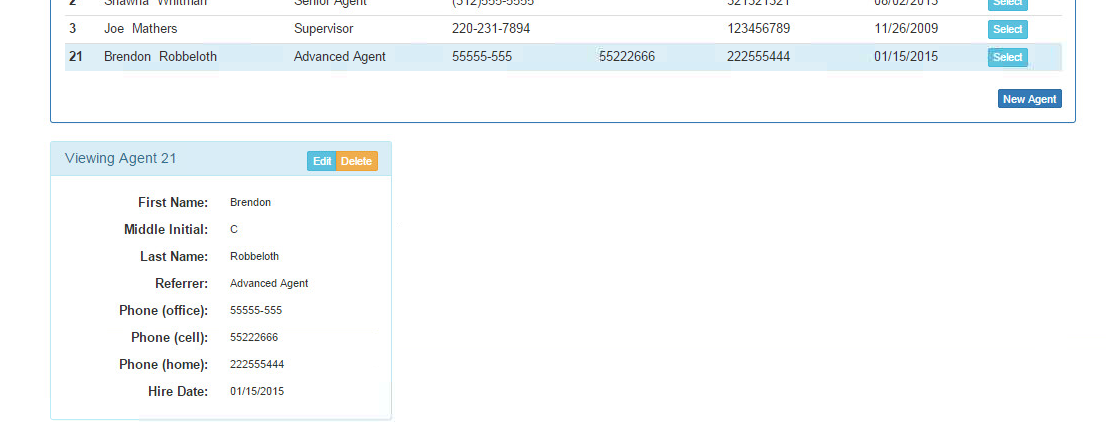
****

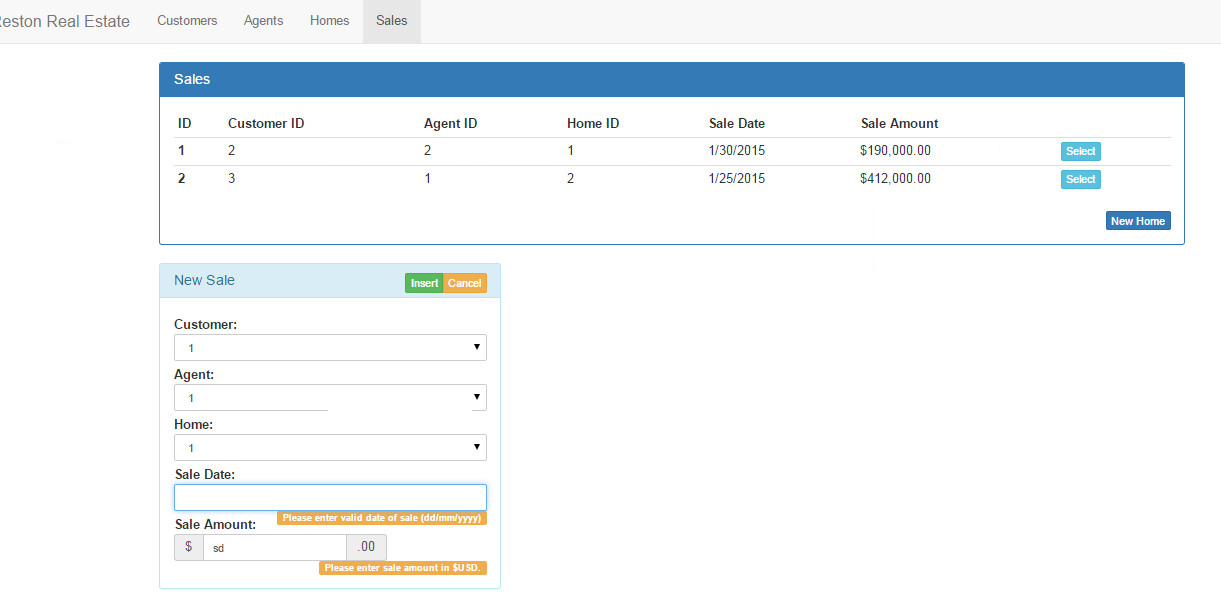
**With good data:  
(1)  
**

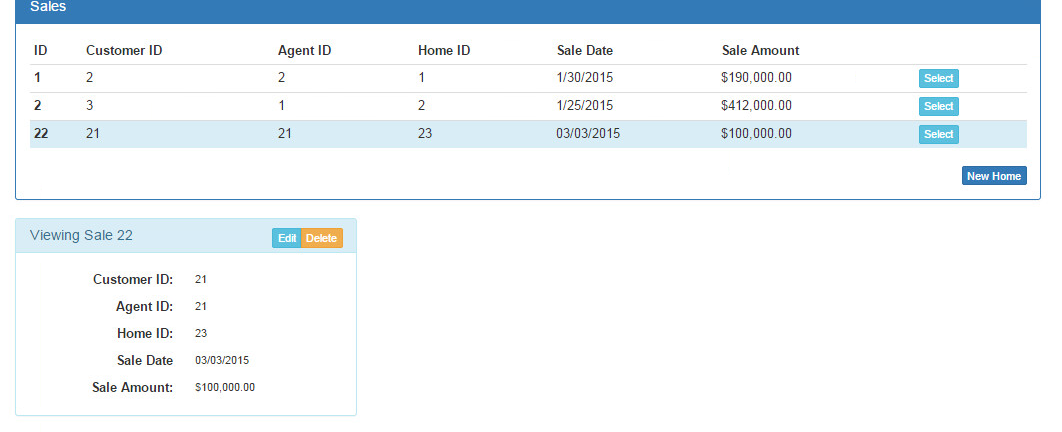
**(2)  
**

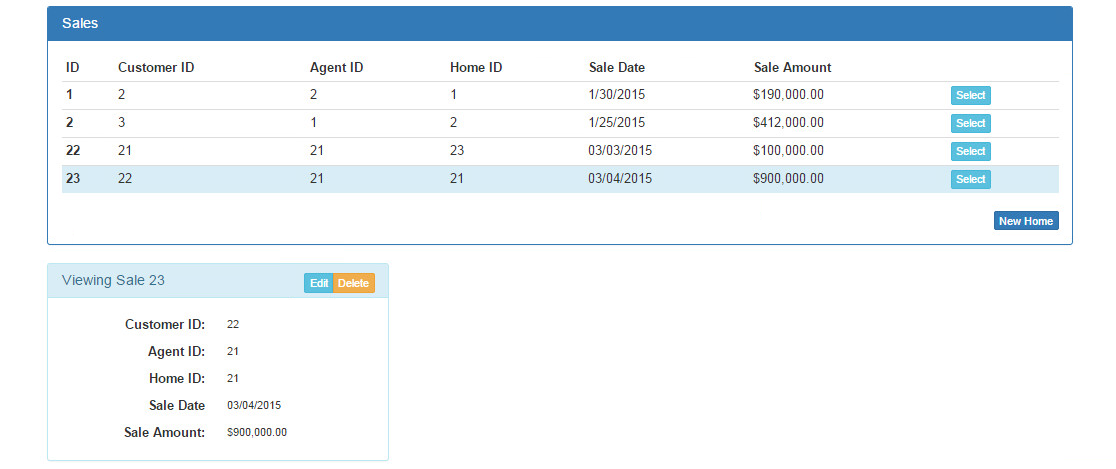
**(3)**

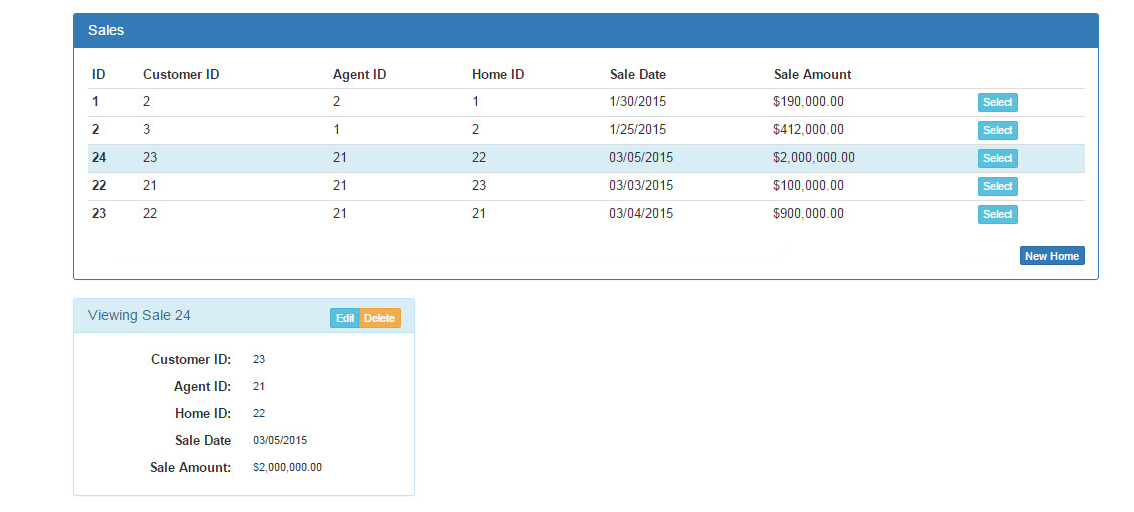
**Add 1 agent (note bad data for validation testing):  
**

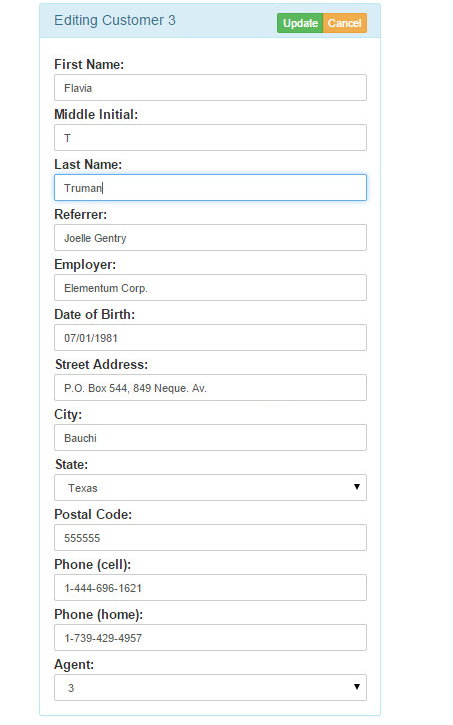
**With good data:  
**

**Add 3 sales (note bad data for validation testing):  
**

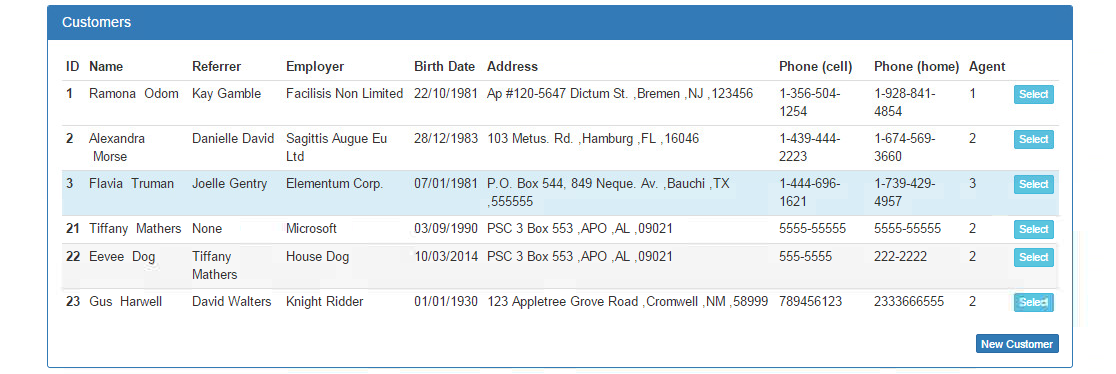
**With good data:  
(1)  
**

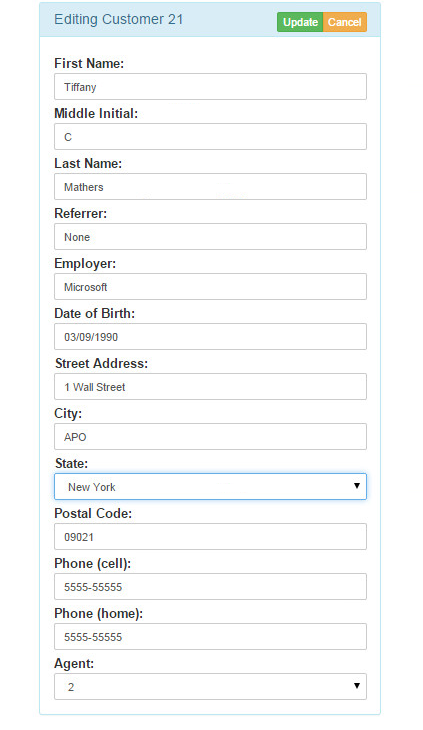
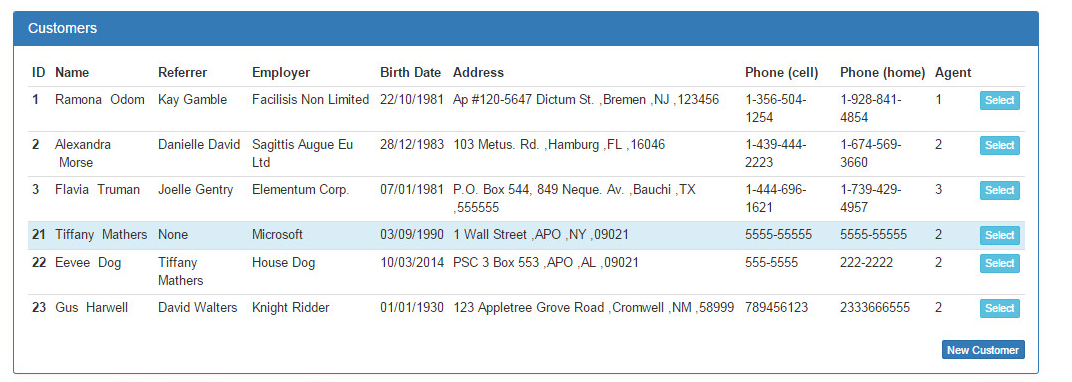
**(2)  
**

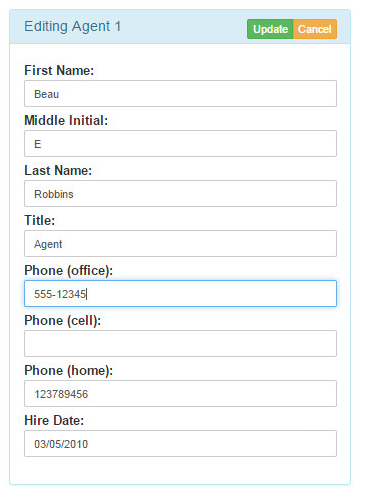
**(3)  
**

**Update customer’s last name (data):  
**

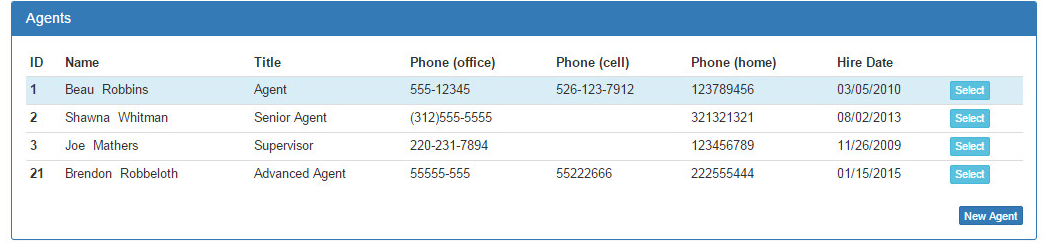
**Update customer’s last name (proof):**

****

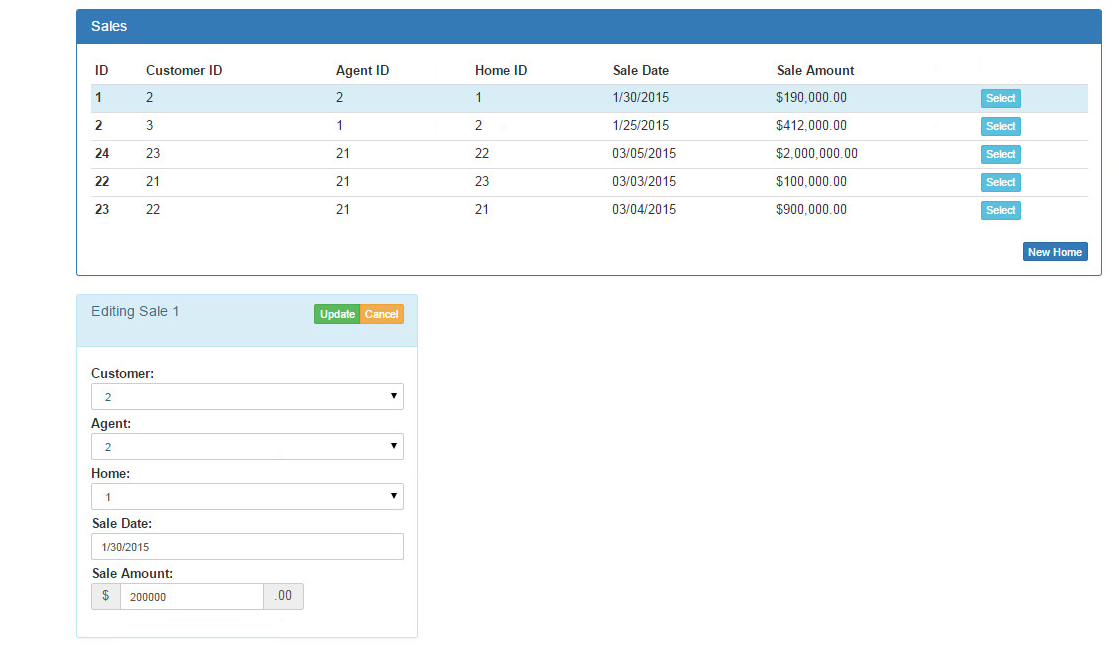
**Update customer’s address (data):  
  
Update customer’s address (proof):  
**

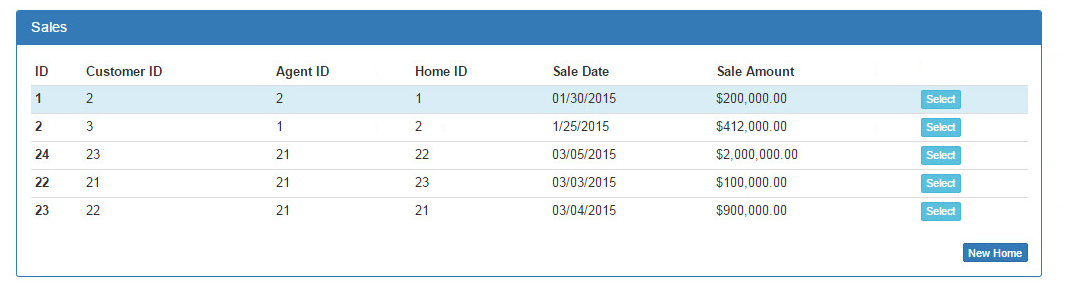
**Update agent’s phone number (data):  
**

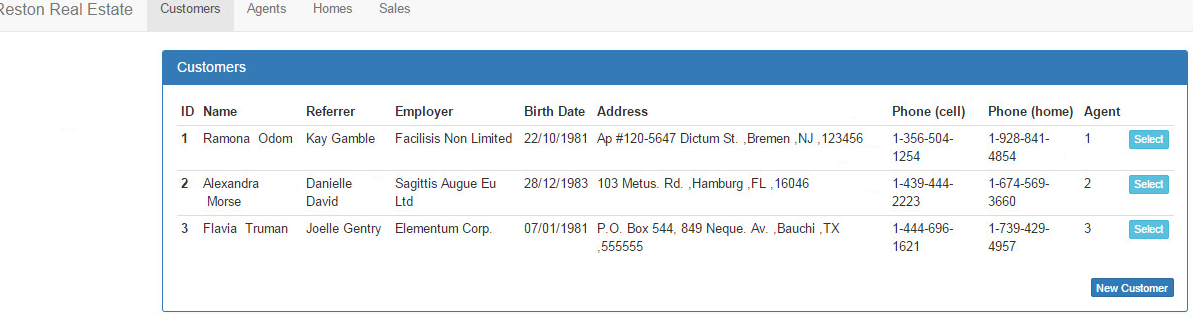
**Update agent’s phone number (proof):**

****

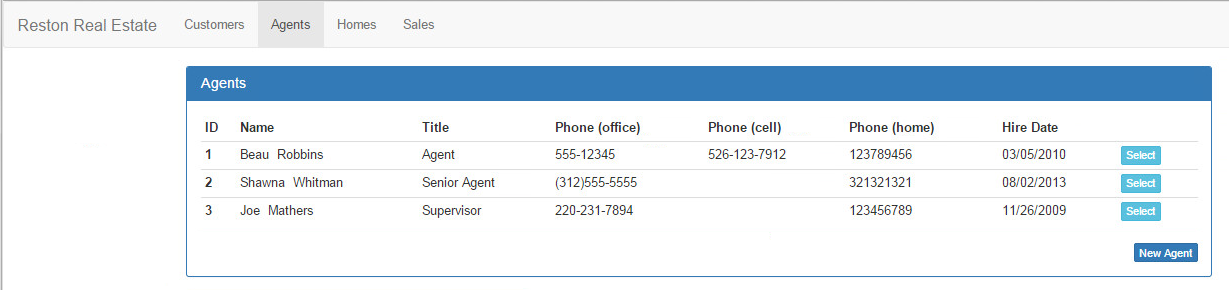
**Update sale amount of home (data):**

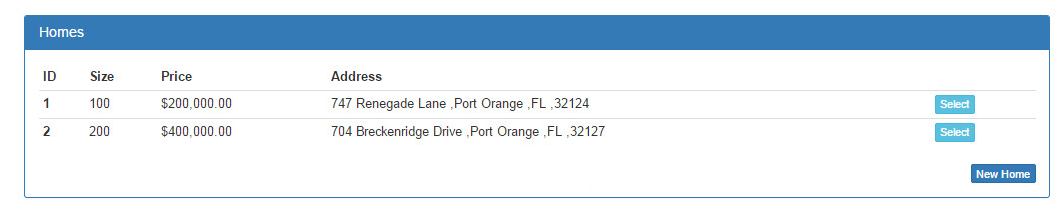
****

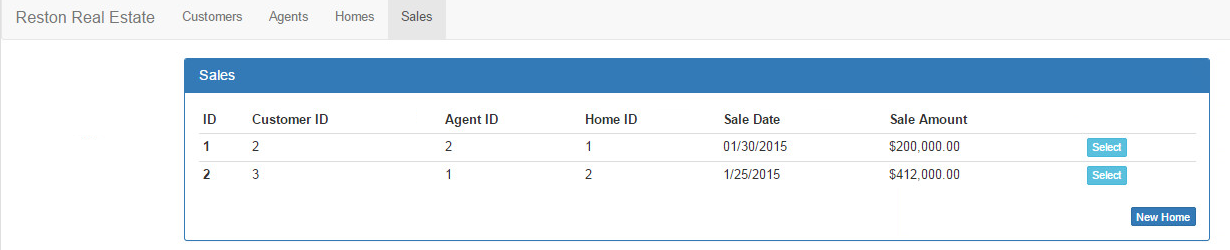
**Update sale amount of home (proof):  
**

**Delete all new data (Customers):  
**

**Delete all new data (Agents):**

****

**Delete all new data (Homes):  
**

**Delete all new data (Sales):  
**