**ISP Term Project**

***Due Date: Sunday, December 13th***

**Objective:** To exercise all of the technologies that we have learned in class.

**Requirements:** Write a User Management application that exercises all of the behaviors of CRUD, (C)reate, (R)ead, (U)pdate, and (D)elete. You will use HTML, CSS and jQuery client-side technologies that talk to a PHP or Ruby based server-side component. You will use MySQL as the RDBMS to store all of the user data. You will use AJAX to perform pre-fill operations and Form submission for telling the server which CRUD functionality to execute.

* Create a new web application that contains four pages:
  + Cover Page
  + User Listing Page
  + Add/Edit User Page
* Your Cover page should contain the following:
  + Page title of ‘*Term Project*’
  + your name in a *span* tag with an id of ***name***
  + the time and date it is finished in a paragraph tag with an id of ***date***
  + h1 tag with the content of ‘Term Project’, id of ***assignment***
  + h2 tag with the content of ‘Internet Systems Programming’, id of ***class-title***
  + You must have a button that takes you to a listing of the users, id of ***user-list***
* The user data that you will need to enter is: (if not denoted, then use input type of ***text***)
  + User ID (name=”***userid***”)
  + First Name (name=”***fname***”)
  + Last Name (name = “***lname***”)
  + Email (name=”***email***”)
  + Phone (name=”***phone***”)
  + Street (name=”***street***”)
  + City (name=”***city***”)
  + State (name=”***state***”)
  + Zip (name=”***zip***”)
  + Date Added (name=”***add-date***”)
  + Male/Female (name=”***sex***”, input type radio button)
* Your second page, ‘User Listing’ page will have all the user data as defined in the above step
  + You must also have an ‘Add New User’ button, id of ***new-user***, that will take you to a screen where you will be able to fill in user data.
  + You will also need an ‘Edit User’ button, id of ***edit-user + UserId***, that will take you to a screen where you will be able to update user data. (Must be same screen as ‘Add New User’ page.
* The ‘Add New User’ and ‘Edit User’ forms must have validation rules as follows:
  + User ID is present
  + First name and last name must have the first letter as capital
  + Email address is present and of the proper form (asdf@asdf.abc)
  + Phone is present and of the proper form (123-123-1234)
  + Street must be present
  + City must have the first letter capitalized
  + State must be 2 character state, capitalized, validate that it is a real state
  + Zip must be only digits, limited to 6 digits
  + Date Added is present and of the proper form (01/01/1971), manual entry of date
  + Validation is done when the user clicks on a *Submit* button and by the change/blur event. If an error is found, the input field that contains the invalid value(s) will change the border of the input field to color ‘red’. The auto grader will look for the border color with the keyword of ‘red’. If anything else is used, this will be marked incorrect, so please follow directions.
  + Two ideas for fulfilling the AJAX requirement:
    - When the user enters a two letter state code, make an AJAX call to the server to get the full state name and display it in a label next to the state edit box.
    - When the user enters a zip code, make an AJAX call to the server to get the city/ies that are located within that zip code for the user to select from or if only one, prefill the city field.
  + For new customers, on submit of the form, all of the user data must be added to the database
* The ‘Add New User’ and ‘Edit User’ forms:
  + Must have a Title on the top of the page:
    - Id of ***page-title***
    - ‘Add New User’ or ‘Edit User’ based on what mode you are in
  + Must have a submit button, input type ‘***submit***’ and clear button, input type ‘***reset***’
  + On submission an HTTP POST will be sent to the server with either the command to update or insert a new record based on what mode your are in.
  + Go back to the ‘User Listing’ page after successful submission of the data
  + Gracefully handle errors from the server
  + On error of the HTTP POST, show the user an error message.
* The ‘Edit User’ form:
  + Populate the data for the particular user.
    - The id will be passed in from the ‘User Listing’ page as part of the URL and will be needed to find the correct user
    - Do not get the data from the ‘User Listing’ page
  + Must have a ‘Delete User’ button, id of ***delete-user***, that will delete the current user being edited.
    - Since you share this page with the ‘Add New User’ page, this button must be hidden when in ‘Add New User’ mode.
    - This button will send a request to the server to remove this users record from the database
* The data will need to be stored in the MySQL database.
  + Use at least two tables to store the data. (Ex: table for street, city, state, zip and a second table for the other fields)
  + Define a primary key for both tables. You can use the UserId field as the primary key if you desire or auto gen the primary key. Do remember that the User Id must also be unique
  + There must also be a way to connect the two tables (Foreign Keys)
  + Validation rules must be used on the server to ensure that the data is good before storing in the database
  + Validation rules must be used to ensure that the same User ID is not used more than once, it must be unique.

**Grading:** This is your term project. You must work on this independently, no teams. Please take care to fill in all of the details as listed above. There is an auto-grader that will look for the specific id’s and name’s for the elements. Please follow the instructions closely. This has a total score of 25 points but is weighted much more heavily (see syllabus for weight) than the rest of your assignments. No points will be given if code does not run at all.

**Submitting your program:** Use the Pausch server to submit the assignment.

1. Create the new folder TermProject in Assignments.
2. Upload your files (\*.\*) to TermProject.