**Instructions**

1. Please solve the problems below using html, css and javascript and send the code through github or a zip file.
2. This is intended to test your ability to
   1. Solve a practical problem you will encounter at work
   2. Work independently using any resources available to complete the problem
   3. Ability to write clean, modular code

**Problem 1**

Put the following text in a comma separated file in your local folder. (Typically it will be available on the web site)

Loan Number,Loan Purpose,Industry Type

100001,Tailoring,Services

100002,Petty Shops,Services

100003,Cow,Agricultural Inputs

100004,Cow,Agricultural Inputs

100005,Tailoring,Services

100006,Cow,Agricultural Inputs

100007,Vegetable Vending,Petty Trade

100008,Tailoring,Services

100009,Petty Shops,Services

100010,Grocery Store,Services

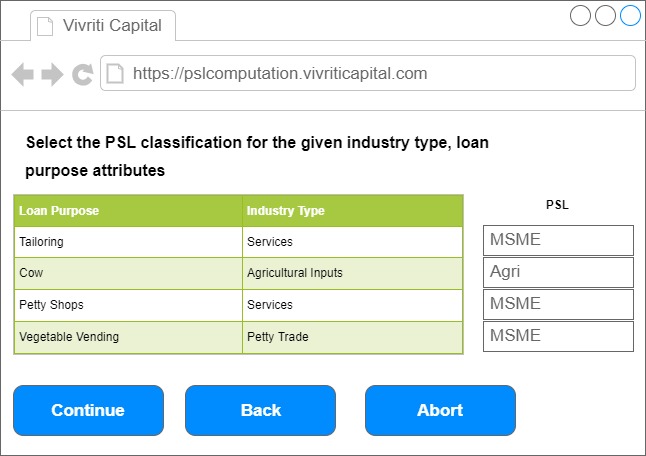
100011,Tailoring,Services

100012,Cow,Agricultural Inputs

100013,Vegetable Vending,Petty Trade

This file contains three columns the first is a loan number, the second is loan purpose and the third is industry type. Read this file in from the local folder, parse it and find all unique combinations of {loan purpose, industry type}. Then display it on a web page a mock up of which is shown below

You should provide a set of files (.html and javascript/jquery/angular/react.js) and one file index1.html and when I access the index1.html file, it should read the CSV file, and display the table as shown below. Not too worried about colors used etc. but the display should be a nice table as shown in the mockup. The third column in the mockup should consist of text fields. The user is expected to input a value for each row of this column. Hence, when I load the index1.html file, all the fields in this column should be empty and I should be able to input text values there.



**Problem 2**

For this problem, again create a set of html and javascript files. The entry point html file should be index2.html. When the file is loaded, it should just have one field **Number of Investors** to be entered by the user. The filed should accept only integer values between 1 and 3 including both. Once the user enters a value between 1 and 3, you should then create n+1 (so if user enters 2, three columns should be created). The columns should be named A1, A2, A3, etc. Then, the user is expected to enter percent values. Then n+1 th column should always be 100-sum of percentages of the other columns. You should validate the the total does not add up to more than 100%. Yield can be any value between 0% and 100% (validate). Across interest priority and principal priority, this is an ordering filed. So, if there are four columns, we should have all values 1 to 8. So, when user presses continue, you should ensure that all the values from 1 to 8 are present and each of them is present exactly once. Further, A1’s interest priority should always be 1 and this text field should be shown and filled with 1 but disabled. So, the user can effectively enter only 2 to 8. If a user repeats the same value, the code should reject the value and not allow him to enter.

