Test 1: Recursive File Structure

C:\

- Documents
  - Images
    - Image1.jpg
    - Image2.jpg
    - Image3.png
  - Works
    - Letter.doc
    - Accountant
      - Accounting.xls
      - AnnualReport.xls
- Program Files
  - Skype
    - Skype.exe
    - Readme.txt
  - Mysql
    - Mysql.exe
    - Mysql.com

This test will help us to explore your research, problem solving and coding capability. Above sample illustrates a file system on the computer. **Red** colour represents the folders (e.g. Documents) and **blue** colours represent the files (e.g. Image1.jpg). This is a sample file system structure and the solution should handle any type of recursive file system structure. For example when we create a new folder, subfolder or file in the system, the solution should able to handle the changes on file system.

Task:

1. Create a database design to able to store given file system as above.

2. Create a text file with above file system structure and read and insert into database. Inserting above structure into database manually will not be accepted.

3. Research for possible solutions and explain that why your solution is better fit than other possible solutions.

4. Create a web interface for the user so they can able to search any file or folder within database. Web search interface should only include an input box and search button. When a user click search button, result should be listed as below. For example if the user enter 'image' into the input box and the click search, the code able to list the results as below.
   C:\Documents\Images
   C:\Documents\Images\Image1.jpg
   C:\Documents\Images\Image1.jpg
   C:\Documents\Images\Image3.jpg

5. The code should be written with OO PHP using design patterns. Unit test is preferred but not required.

Test 2: Multi Form Validation

This test helps us to explore your knowledge of JavaScript and PHP with web form handling. The web form will have name, email and phone number of the contact. The user should able to add a new contact, validate and remove already added contact. You need to have following codes to meet test requirements.

- **Frontend Validation**
  Write an OO JavaScript class to validate all input elements on the web form. When user clicks validation, display the validation errors near the input fields.
- **Backend Validation**
  Write an OO PHP class to validate all input elements on the web form at server side. Assume that JavaScript is disabled and server side PHP validation is on place. Validation class should collect all validation errors and displays to the user all together. If there is no validation error on server side, save all contact data into a text file.
- **Validation Rules:**
  Name: Alphabets and space.
  Email: Able to validate all valid emails.
  Phone: Numbers only
- **Action Buttons**
  Add Contact: This button should insert an empty set of contact form (Name, Email, Phone) to the same web page.
  Validate: This button should trigger JavaScript validation for all input elements. Validation errors should be displayed under each input element.
  Remove: This button should remove selected set of contact form from web page and the text file which has been saved.
  Save: This button should submit the forms on the page to server side so PHP can handle form validation and save into the text file.