# Coding Standards and Guidelines

## HTML

1. Place all HTML files under WebContent folder.
2. HTML5 semantic tags has to be used wherever applicable. Avoid using div tags for header, footer, article and navigation bar.
3. Document Type should be defined as

<!DOCTYPE html>

1. Form fields must always include a <label> element with a "for" attribute matching the "id".

<label for="age">Age</label>

<input type="age" id="age" name="age" value="" />

1. Each form field should have "name" attribute.
2. All tag names and attributes must be written in lowercase.
3. Inline style should be never used. Style should be in an external CSS and should be linked to the document.
4. Image files should be placed in WebContent/images folder.
5. Add one more tab on the next line after opening a tag
6. Reduce one tab on line when tag is closed

## CSS

1. CSS files names should be style.css
2. CSS files should be placed in WebContent/style folder.
3. Feel free to change the Background Color, Font, Font Size and Font Color of the screen. But it should be applied consistently across screens using CSS. Following are the element in the screen that can be targeted for style change:
   1. Navigation Bar
   2. Page Heading
   3. Form Elements
   4. Messages
4. Reuse styles wherever applicable. Potential areas are Application Name, Page Title, Form Fields and Buttons
5. Selector names should be in lower case with words separated by hyphen.

.page-title {

margin: 20px;

}

1. Selector class names should be relevant to the purpose of the element where the style is applied. For example if common style needs to be applied for all text boxes in an application.
2. Properties should be followed by a colon and a space
3. Properties and values should be in lower case
4. Use hexadecimal code for color definition

Example:

.toggle {

font-family: Arial, Helvetica, sans-serif;

font-size:20px;

padding:10px;

text-align: center;

height:20%;

background-color: black;

color: white;

}

## Spring

1. Never use System.out.println(), use logger.debug() instead
2. Include start and end logs in each method
3. Include debug logs for data retrieval and flow
4. Each Rest Controller class should have RequestMapping that maps to the particular domain entity
5. The URL definition should be in all lower case with words separated with hyphen.
6. Method level URL definitions in rest controller should contain only the parameters and should not contain any other URL definition.
7. Apply the right http method based on the operation performed. POST for creation, PUT for updation, GET for reading data and DELETE for removing data.
8. Do not include any instances level variables in controller apart from service auto wiring. Remember that in production environment based on volume of usage, there is a good possibility a rest controller method might be invoked in parallel, having instance variables with user specific information will overwrite one over the over and might result in bad user experience.