**Client Side Dynamic Forms Renderer**

Develop an HTML5 dynamic form renderer that is driven by metadata that is fetched from REST API. The metadata contains a list of fields to be rendered by the form. The form renders a field in either of the two modes:

1) **Edit Mode** which is used to collect input data from user using familiar input controls (text box, drop down, check box, date time picker, custom, etc…),

2) **View Mode** which is used to display information.

The high-level flow is as follows:

1. A page containing the dynamic form is loaded
2. Components responsible for rendering html fields are loaded from server
3. The renderer fetches form’s metadata from server based on its name or ID
4. The renderer parses the metadata and renders the result in the browser

The main requirement of this renderer is to be customizable by developers

1. The ability for a developer to write a custom component (HTML5, JS, CSS3) that handles rendering a specific field type
2. The ability to deploy the component to server component repository
3. The component is loaded by the dynamic form when rendering a field

**Technical Specs**

1. It should fetch form metadata based on an ID or name
2. It should parse the JSON
3. It should render each field corresponding to the result of the parsed JSON
4. If A field is a custom component it should be fetch from the server by its name or ID and rendered accordingly.
5. It should add any validation, whether on a single field or multiple field according to the metadata
6. It should add the form actions (submit, cancel, save…etc.) as described in the metadata

**Example Usage**

A business analyst builds a registration form in the backend by describing each field, its validation, etc. Provided with this document two metadata files:

1. metadata\_view.json has the metadata for fields in readonly mode
2. metadata\_edit.json has the metadata for fields in edit mode.

***Note: This backend is not in the scope of this document.***

**Metadata Description**

**Properties**: an array containing a collection of properties to render in the form.

**Actions:** an array containing a collection of actions that are rendered as buttons.

The attributes for each property depends on the type of property itself:

* “label”: contains the text for input control label
* “id”: a unique identifier of the fields
* “tooltip”: tooltip for control
* “type”: identifies the type of the field
* “viewMode”: identifies the view mode of the field. Values are: readonly and edit
* “required”: indicates whether the field is required
* “choices”: available only for fields of type MultiChoice and contains a list of legal choices.
* “length”: available only for fields of type Text and defines the max length for text field
* “minValue”: available only for fields of type Integer
* “maxValue” available only for fields of type Integer