**Frontend as a Service**

**Overview:**

Frontend as a Service (engine) for rapid UI/UX development. Base on the predefined **templates and components** **UI/UX easily implementable**. The main purpose of this engine make application lightweight so that base on customer need it can easily configurable. Base on meta configuration fronted development will be rapid.

This engine consumes endpoints from Object Lifecycle Management, Tree Engine, Property Extender, Rule Engine, and Rule Engine.

After process static, dynamic meta it can populate view, with helps of defined templates and components.

**Templates:**

Template is need to handle actions and render meta for components. Template also reusable but most cases recommended, create new template for individual UI Design.

**Components**:

Components are predefined view which are reusable in templates. When we provide meta into a configurable manner then those templates are populating with its nested components.

**How Template and component Design:**

Components will be always reusable. But template might be individual or reusable UI & UX.

Here, recommendation will be repetitive UI & UX break into small components so that those design reusable and reusability will be increased.

Top Navbar

Side Nav-Bar

Master View

Master Page

**Master Page:**

After user login into system based on use authorization, UI master page will be populated with Top Navbar, Master view and Side Navbar.

Top Navbar: In Top navbar dynamically available those service whose are permit to access for that user.

Side Navbar: In Side Navbar base on user access submenus will be populate from selected service.

Master View: Base on the selected menu item load template meta then process it, and render component’s from meta then finally generated view.

All the component’s render into a template. Those templates render on Master View.

Meta Configuration: It is just a defination how actually template populated view, which component are load, where from it load necessary data, which action will be performed, how basic validation will be, and how can share data to other view those things need to configured through meta.

High Level Architecture:

**Fronted Engine**

**Browser**

Process & Render - Business and System meta Information. Based on user authorization it populates features and services

Browser Process HTML, CSS, JavaScript

Render View

**Auth Engine**

c

**Tree Engine**

c

**Rule Engine**

c

**Property Extender**

c

Internal Architecture:

**Menu Tree**

**Application Panels**

Communicating through APIs

**Master View**

**Base Layout**

**Tab Table Reusable Grid View from Resource Libraries**

**Collections**

**|-Functions\_1**

**| |-sub-fun\_1**

**| |-sub-fun\_2**

**|-Functions\_2**

**| |-sub-fun\_1**

**| |… more**

**|… more**

Base on client subscriptions

Engine Populated available features & services as predefined configuration

**Tree Engine Service**

**Auth Engine Service**

|  |  |  |
| --- | --- | --- |
| **RMS Application** | | **Access layer** |
| **Operational Panel** | **Collections (Remittance, Beneficiary...more)** | **Base On Authorization** |
| **Admin Panel** | **OLM and Operational Panel** | **Base On Authorization** |