**User Interface**

**Link to Documentation of kefeD :**

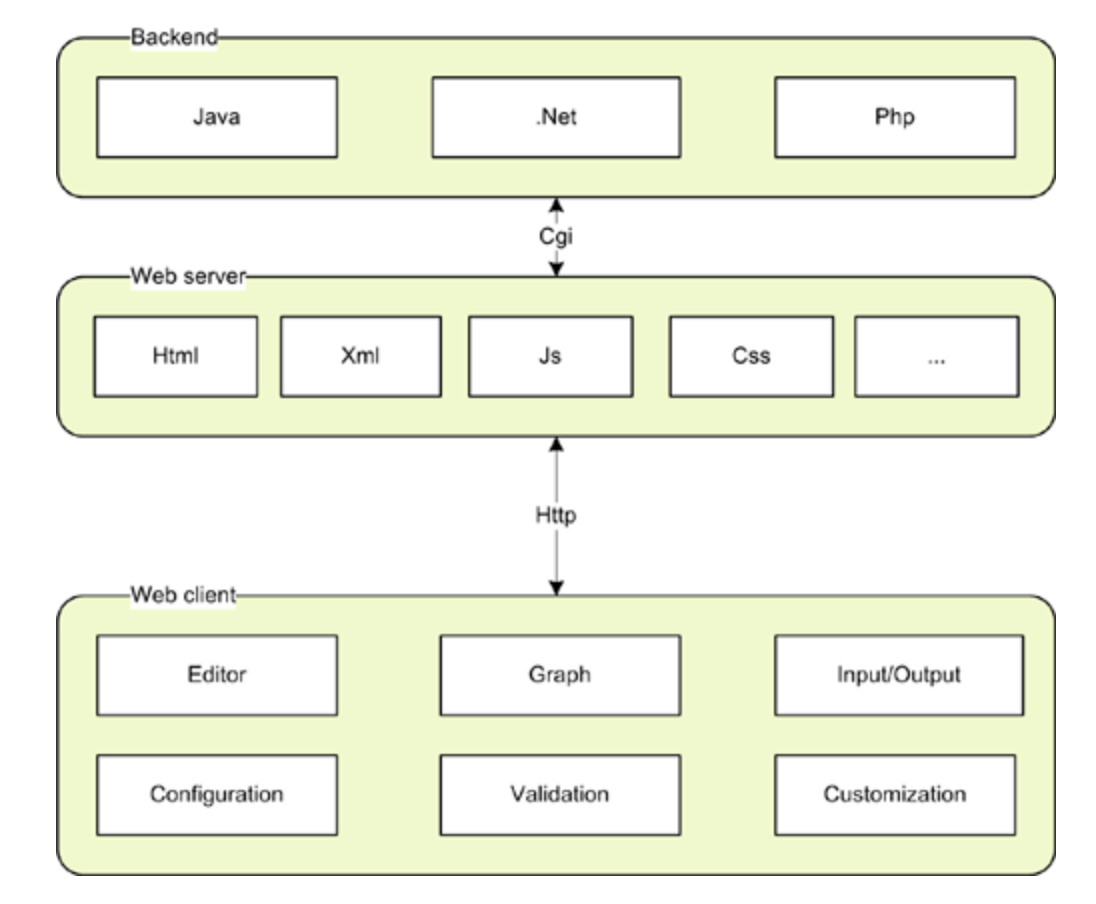
[**https://github.com/SciKnowEngine/kefedProject/wiki**](https://github.com/SciKnowEngine/kefedProject/wiki)

**Link to code:**

[**https://github.com/SciKnowEngine/kefed.io**](https://github.com/SciKnowEngine/kefed.io)

**mxGraph Library**

mxGraph is a JavaScript component that provides features aimed at applications that display interactive diagrams and graphs. mxGraph provides all the commonly required functionality to draw, interact with and associate a context with a diagram. The demo is provided on <https://www.draw.io> . The flow with existing system goes as



**mxGraph and Kefed**

Knowledge engineering project aimed to extract information from biomedical ontological database using semantic modelling and knowledge graphs. mxGraph is integrated with kefeD system to provide biomedical practitioners a tool to create and update a new model.

**Structure of mxGraph with KefeD**

mxGraph files start with init.js. init.js includes all web js file imports. The javascript files imported are named after the functionality they provide in mxGraph toolset. Eg. EditorUI.js and Editor.js are javascript functionality associated with diagram editor. ToolBar.js controls functions with toolbar. MenuBar.js with menu bar and respective modules.

Initialization of System

1. Index.html in demo webApp folder is loaded via SpringBoot.
2. Index.html will load up EditorUI.js.
3. Editor UI JavaScript file is loaded and setting all default values of width, height positions as per user screen.
4. Respective files are initiated and values are substituted as per current screen.
5. Updated changes are written as comments in respective files.

**kefeD BackEnd**

kefeD backend is written in Spring/Java and integrated with Elastic Search. The system uses Spring Boot default spring security features integrated with UI and performs business logic transformation of converting it to data model of kefeD and saving it into System.

**KefeD.js**

This is main file in UI which converts mxGraph XML model into kefeD Data model. This file is integrated with index.html. Respective comments are written in js file. The file will parse XML as per the components used up in model and map them in kefeD data model given on Wiki Page.

**Code Explanation (Important Pieces)**

**Index.html**

this.editor.graph.addListener(mxEvent.CLICK,mxUtils.bind(this function(evt){});

This is main function handling connection between mxGraph XML and kefeD model. This function takes XML on every change of input and upgrades the kefeD model as required. This will call kefeD.js by using functions:

1. changePropertyObjectValue

It will parse XML and update components values in kefeD model in kefeD js. This model will then use obtained javascript object to save into Database.

1. updatePropertyDialog

This will update/fetch the property values of corresponding component from kefeD.js and replace it into property dialog. After replacing values in propertyDialog, it will update corresponding kefeD model in kefeD.js by polymer two way binding data.

**EditorUI.js**

EditorUi.prototype.createSaveCancelbar

This function is part of Saving and Cancelling the entire model. This is not part of mxGraph and been custom developed and integrated with mxGraph.

Other files containing this customization –

Index.html

Editor.js

EditorUI.js

Graph.js

Menubar.js

Custom comments are written in files.

**EditorUi.prototype.saveFile**

This function will recalculate new kefeD model before saving to remove any inconsistencies.

**Default.xml**

This file contains all information about components like Process, Measurements and forks used on draw io panel.

The main part:

<add as="parameter">

    <add as="shape" value="image"/>

    <add as="perimeter" value="rectanglePerimeter"/>

    <add as="fontSize" value="11"/>

    <add as="align" value="left"/>

    <add as="verticalLabelPosition" value="bottom"/>

    <add as="verticalAlign" value="top"/>

    <add as="width" value="10px"/>

    <add as="image" value="ui/grapheditor/www/images/parameter.gif"/>

</add>

The <add> tag of mxGraph will upload defaults into EditorUI and then into graph.js There are custom <add> tags added for each respective component of kefeD model in the same file.

The file will be then loaded into **SideBar.js with** this.createVertexTemplateEntry function where its dimensions, height, width and other values are set.

**\*\*Polymer UI**

Polymer dashboard is updated in dashboard/app folder. The folder has bower.json file which contains polymer components used in UI dashboard.

System runs as:

1. Bower.json will be created/updated using bower command.
2. The polymer components will be loaded into bower\_components.
3. System will then fetch out kefeD model details from backend via RestController.java and store it in kefed Object.The Object is created from kefed.js using structure described in wiki.