**User Guide**

This User Guide is intended to give a brief overview of the “Y Process Suite”.

**Purpose**

The purpose of Y Process Suite is to model and view business processes in the standardized [BPMN 2.0](https://www.omg.org/spec/BPMN/2.0/PDF) notation.

**Intended audience**

Typically, Y Process Suite will be used by the owners of small and medium sized companies, who need to develop and/or document their business processes.

**Why Y?**

While there already are applications available, which can be used for the same purpose, none of them is developed in Switzerland and has the functionality of Y Process Suite. These are the main differentiating features:

* Free to use
* Works in the browser / no installation necessary
* All diagram elements can be linked to other diagrams
* All diagram elements can be colored in unique colors

**1 How to use Y Process Suite**

This section of the user guide will give an explanation on how to use the application, including any key features and functionality.

**1.1 Prerequisites**

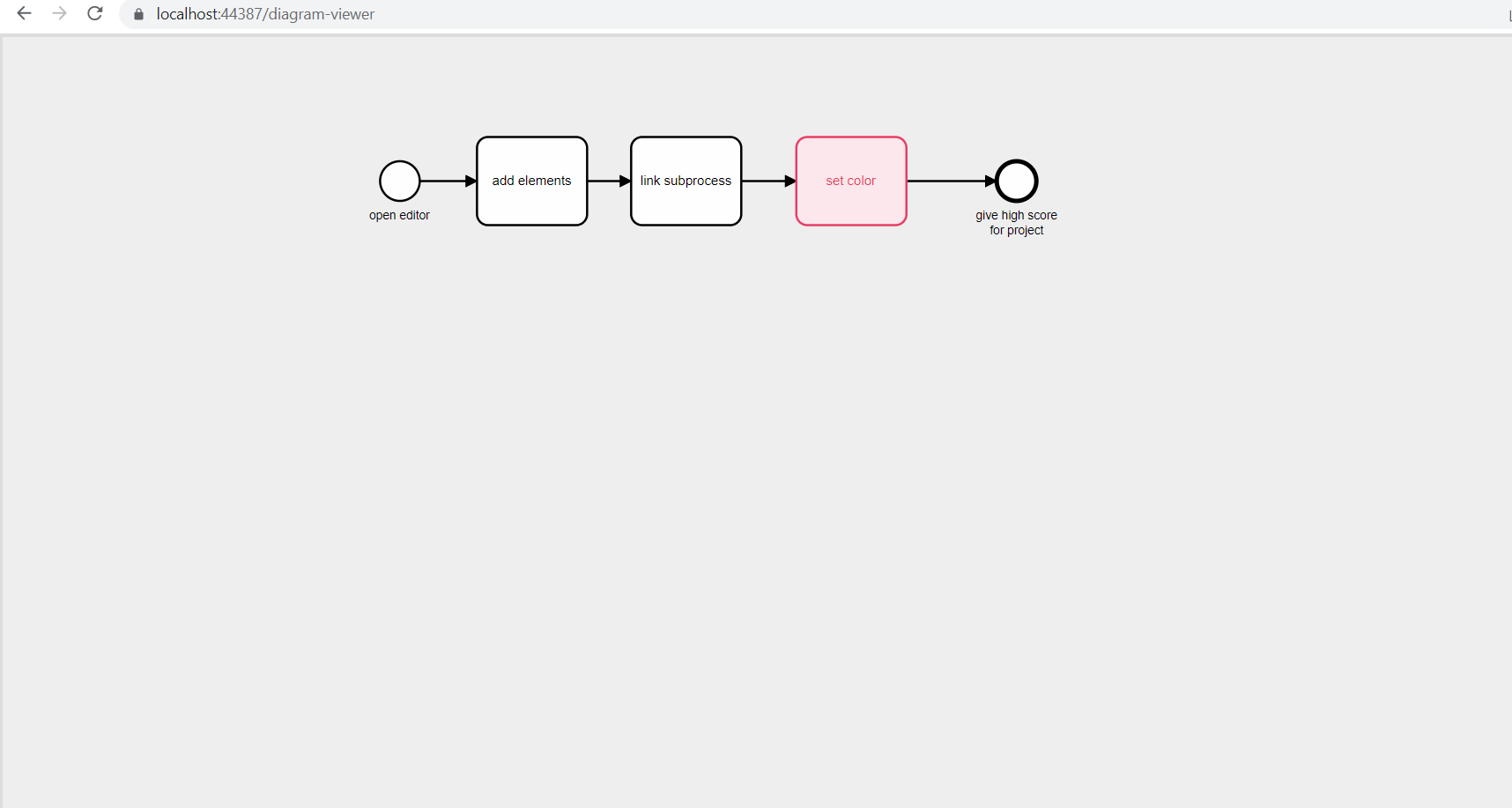
Currently, the full application with the functionality described below, needs to be built and run locally. This process is described in the technical information chapter of this guide.

*Note*: a prototype version of the application with reduced functionality is deployed already and can currently be accessed without further installation or registration requirements at [prozesse.yayu.ch/](http://prozesse.yayu.ch/).

**1.2 Key features**

After accessing Y Process Suite, users can choose whether they want to use the editor or the viewer module.

Note: on the local build, the editor is opened by default. Changing to the viewer module is currently done by manually changing the URL from in diagram-editor to diagram-viewer.



**1.2.1 Y-Editor**

The [Y-Editor](https://prozesse.yayu.ch/editor.html) (local build: diagram-editor) can be used for modelling BPMN diagrams. After opening the Y-Editor, a standard diagram is pre-loaded inside the drawing container. On the left side of the container is the main toolbar, which can be used to add elements to the diagram via drag-and-drop.

*Element Toolbar*

When clicking on an element, an element-specific toolbar appears. This toolbar offers an efficient way of adding further elements to the process diagram.

*Change element color*

The standard element color is black/white. Individual colors can be set by clicking on the brush icon (set color) in the element toolbar. Currently, the colors are only indicated by text. A representative visualization of the colors in the toolbar is planned to be implemented in a future version of Y Process Suite.

*Save diagram*

To save the diagram, click on the *Save* button at the top of the page.

*Add subprocess*

To add a subprocess to an element, select the element and click on *Add Subprocess* at the top of the page.

*View/edit subprocess*

If a subprocess is already linked to a selected element, the subprocess can be accessed by clicking on *View/Edit Subprocess* at the top of the page (only visible when a subprocess exists).

*Attributes*

The attributes panel is located at the right side of the editor. Its functionality is currently under review and will change in upcoming versions of Y Process Suite.

**1.2.2** **Y-Viewer**

The Y-Viewer can be used to access and view BPMN diagrams that were modelled in Y-Editor beforehand. After opening Y-Viewer, the predefined standard diagram is shown on the screen.

*Zoom-In / Zoom-Out*

The diagram is always rendered in a size that fits the available space in the viewer container on screen. If the screen / the browser window is very small and/or the diagram is very large, that might lead to usability problems. To circumvent this, adjusting the diagram size is possible. Zooming in and out is possible with CTRL + Mouse Wheel and/or Pinch-to-Zoom with touchscreens and trackpads.

*Open Diagram*

Currently, this button does not serve any function. This is planned to be implemented in an upcoming version of Y Process Suite.

*Export Diagram*

Currently, this button does not serve any function. This is planned to be implemented in an upcoming version of Y Process Suite.

*Show Attributes*

Currently, this button does not serve any function. This is planned to be implemented in an upcoming version of Y Process Suite.

*Attribute Panel*

To maximize usability and the available viewing space, the process attributes will be shown below the diagram. Currently, this does not work. It is planned to be implemented in an upcoming version of Y Process Suite.

**1.3 Technical Information**

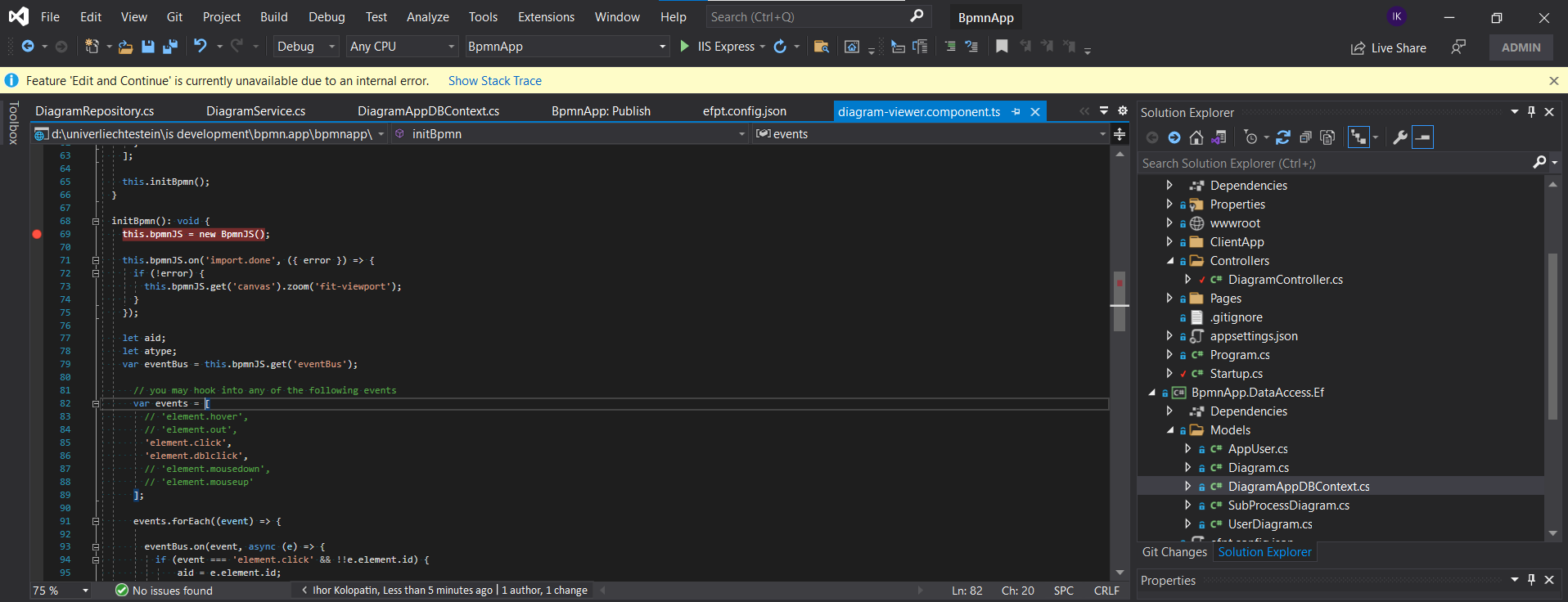
This section documents the libraries and toolkits that are used in the development of Y Process Suite. Also, it explains how to set up the application locally.

**1.3.1** **Project setup:**

To set up the project, these tools and technologies are needed:

1. Visual Studio 2019
2. VS Code to go through the frontend comfortably (optional)
3. Node.Js v16.13.0 (with v18.12.1, problems were observed during development)

After installation of this tools, take the project from [GitHub](https://github.com/Y-creation/00-Project-Y) and open the solution file (BpmnApp.sln) in Visual Studio and run the project by clicking this button:



**1.3.2 How it’s done:**

Y Process Suite uses Angular for frontend and ASP.Net Core for backend.

**FrontEnd – Angular:**

Start-module: AppModule – defines all child modules that are inside the application.

Routing-module: AppRoutingModule – defines the possible routing for the application and components that are responsible for them.

DiagramMainModule – this module is responsible for the webpage, where diagram-editor or diagram-viewer is opened by the user. Inside this module is DiagramMainComponent that opens Viewer and Editor and contains Save button, which is responsible for saving content.

DiagramEditorModule – module, that is responsible for loading and editing diagram and Adding/Editing subprocesses

DigramVieweModule – module, that is responsible for viewing diagrams

The diagram is loading by Id, which is got from DiagramMainComponent, as for now it’s hardcoded there.

Service part:

DiagramService – responsible for communicating with BackEnd to save or get values.

**BackEnd – Asp.Net Core:**

As for now it has some look-like microservices structure, which has five projects inside solution:

**BpmnApp.ApiContract:**

As for now stores only requests and responses models classes.

**BpmnApp:**

Stores ClientPart(Angular Application) and Controller Part( takes requests and gives responses)

**BpmnApp.DataAccessEF:**

Implements DataBase logic inside objects( with the help of Entity Framework)

As for now DataBase is deployed, so connection link goes directly to the uploaded to the server database

**BpmnApp.Domain:**

Contains application logic and interdependencies between controller and repository part. The main classes are Services: here it is DiagramService, which should impement the logic and call right repositories’ methods. As for now mostly calls onlly Repository methods

**BpmnApp.Repository:**

Direct work with DB objects( get and set data inside database)

**How it should be done:**

**FrontEnd – Angular:**

Should be updated the logic of viewing SubProcesses instead of editing

Should be added possibility to delete subProcess if neccessary

Should be added another modules, that would get list of diagrams that are available for the user and provide him with the possibility to **update** or just **view** these diagrams. Also should be impemented adding new diagrams and sharing with another users, giving them different rights(View or Edit)

For this functionality should be also implemented Register, Login and Log Out options.

**BackEnd – Asp.Net Core:**

Should be added new controllers, services and repositories to divide some current functionality by its logical point(meaning) and to add a new functionality, which was discribed in FrontEnd part

**Database:**

Should be updated current tables to exlude duplication of data and to cover all neccessary functionality.

**Some Dev Deffects, that are known after development:**

* As for now if user deletes task from Diagram and this task already has SubProcess, the SubProcess still exists in DataBase.
* As for now if subproccess is just added, user should reload page to see View/Edit Subproccess button instead of Add Subprocess -> should be implemented the logic where user can not add subprocess for the element, if it already contains it and Grid should be reloaded just after user added new subprocess

**How To:**

**How to access edit mode:**

Just run your application)

**How to access View Mode:**

The link should contain diagram-viewer instead of diagram-editor

Like this:

**How to access View Mode for SubProcess:**

You can find that in next release)

**Which tecnologies were used:**

**BackEnd**

AspNetCore,

EntityFrameWork,

MSSQL Database, for publishing Database we used smarterasp.net hosting

**FrontEnd**

Angular

RXJS,

BPMN.JS

[00 Project Y](https://campusfl-my.sharepoint.com/:f:/r/personal/denis_dadybaev_uni_li/Documents/00%20ISD/00%20Project%20Y?csf=1&web=1&e=XFFmYk)