Pontifícia Universidade Católica
do Rio de Janeiro

Final Programming Project

MoLIC Editor: Create and Edit MoLIC Diagrams online

Caroline Loppi

Advisor: Prof. Simone Diniz Junqueira Barbosa

July 2, 2022

Contents

1 Objective **2**

2 Program Specification **2**
2.1 MoLIC Diagram 3
2.2 User Stories & Mockups 3

3 Definition of Done **8**

4 Architectural Project **9**
4.1 Class Diagram 9
4.2 Data Model 10
4.3 Main Technologies Used 12

5 Tests **13**

6 How to Download and Execute **13**

7 Future Work **13**

1 Objective

The goal of the Molic_Editor is to be a web application that allows its users to create and edit MoLIC diagrams using modern technologies (such as Angular and Node.js). This project aims to provide a quick and simple way to elaborate MoLIC Diagrams. Considering this is an open-source program, further collaboration of the community will be possible.

2 Program Specification

This system's requisites are divided into three specifications as follows: MoLIC Diagram, Mock-ups, and Use-Cases. This document contains the full representation of Molic Editor specifications, which exceeds the scope of this Final Project.

The Molic Editor is part of my Master's research and the current state is a MVP version that will be continuously enhanced during the research time (this and next semester). The Final Programming Project had its scope reduced to include the following user stories: (US1) Create a MoLIC diagram from scratch, (US3) Connect base MoLIC elements using arrows and (US6) Drag elements on canvas.

The project source data is available at: `https://github.com/carolineloppi/molic-editor`

2.1 MoLIC Diagram

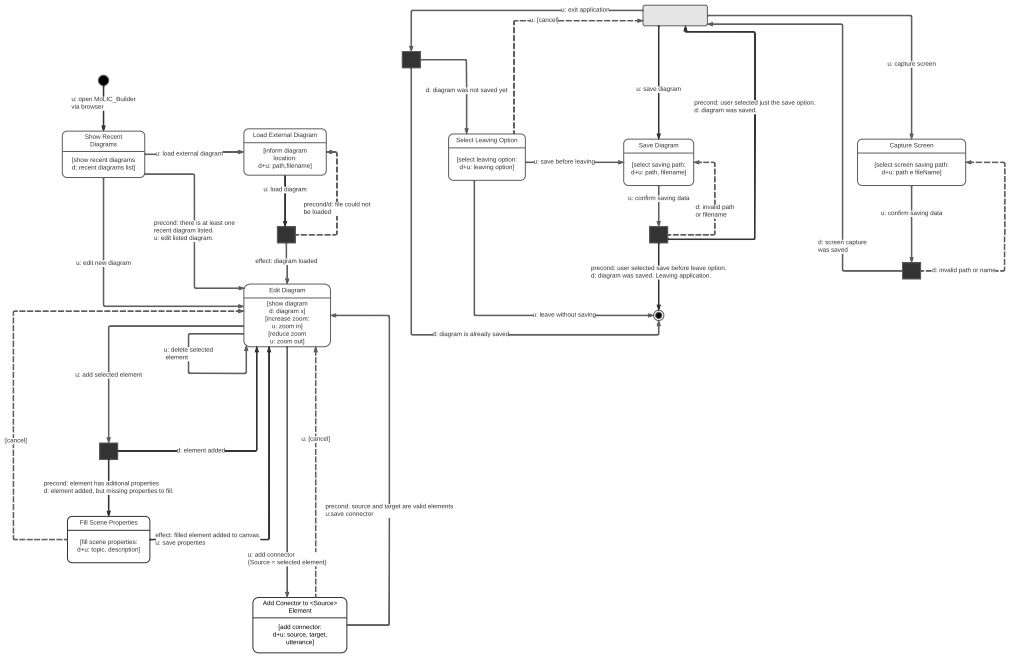

Figure 1: MoLIC Diagram representing the interactions between users and the system

2.2 User Stories & Mockups

In this section, we present the user stories associated with the mock-ups. The use case US1 represented in figure 2, shows the initial state of the application.

US1: As a user, I want to be able to create a MoLIC diagram from scratch, so that I can start a brand new work.

Figure 2: Initial Screen

Figure 3, shows the form where the user can populate a node (in this case, a scene).

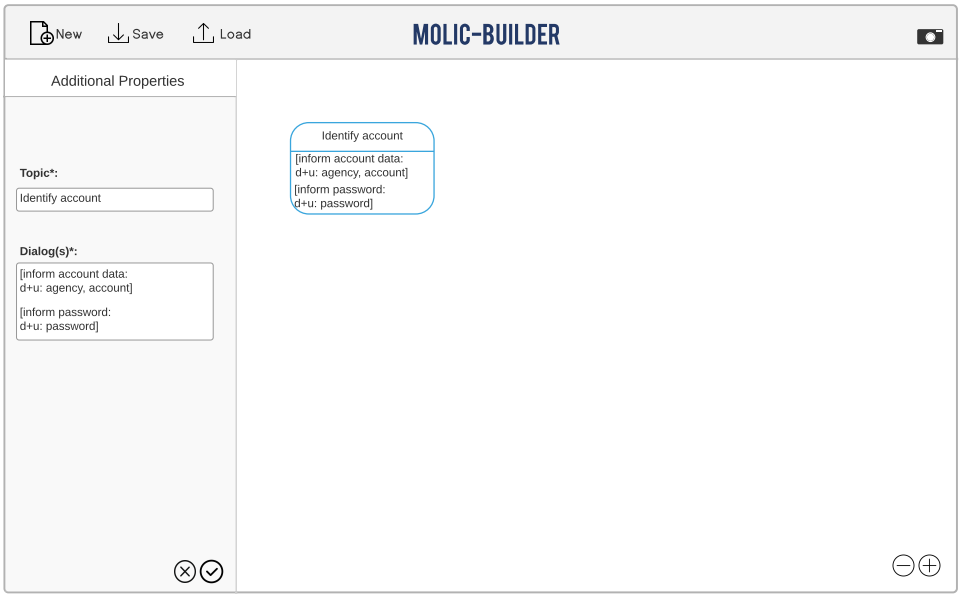

Figure 3: Create Scene

Use case US2 shows an alternative path a user can follow, which is

loading an existing diagram rather than creating one from scratch. Figure 4 illustrates the dialog where the user will select the file path of the diagram.

US2: As a user, I want to be able to load and edit an existing MoLIC diagram, so that I can continue my work whenever I want.

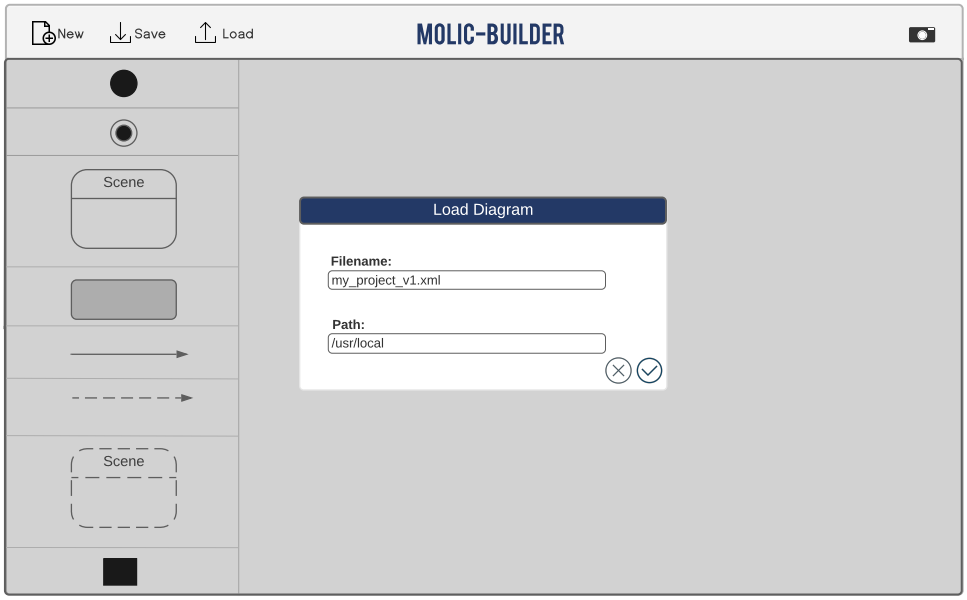

Figure 4: Load Existing Diagram

Users can interact with the application by connecting elements. As shown in use case US3, the user has to double click a type of connection in the left panel (as shown in figure 5), select the source and target elements, to then, enter the associated utterance.

US3: As a user, I want to connect the base MoLIC elements using arrows so that I can mount my MoLIC diagram in a simple and interactive way.

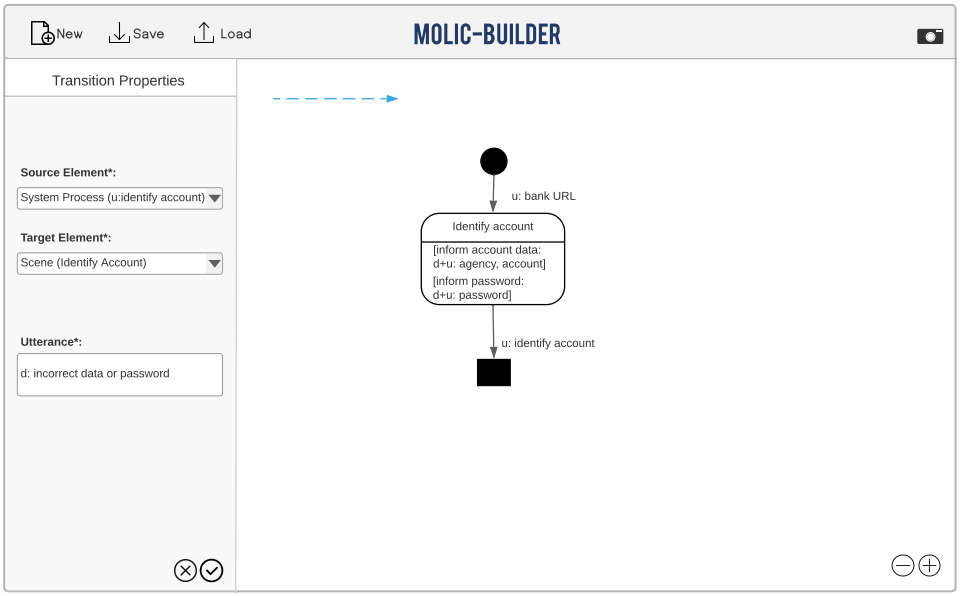

Figure 5: Connect Nodes

Figure 6 shows that the application also allows to screen capture the current state of the diagram, as described in the use case US4.

US4: As a user, I want to be able to screen capture the MoLIC diagram under edition so that I can share it with my colleagues.

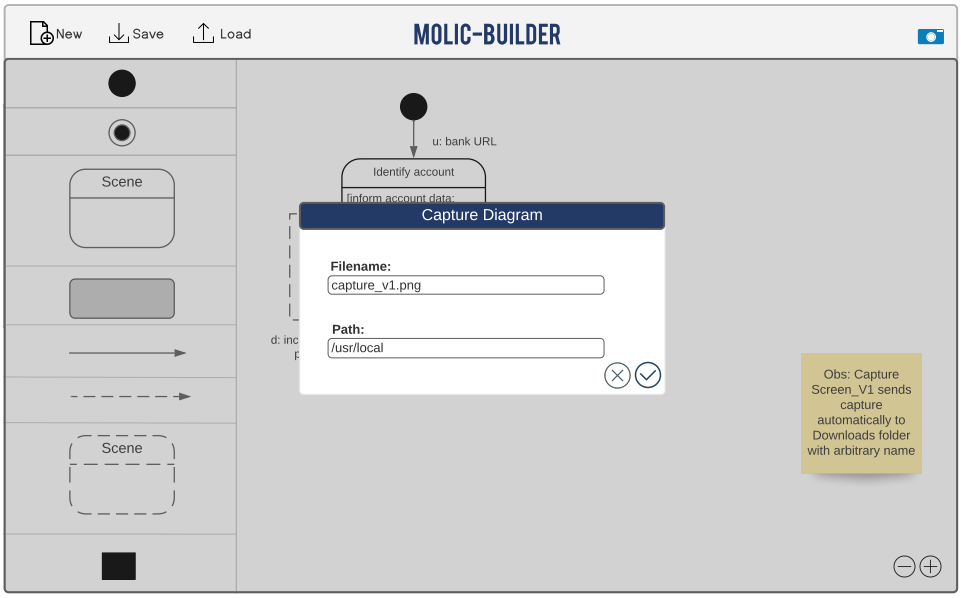

Figure 6: Capture Screen

Through the dialog presented in figure 7, users can export the current diagram they are working on, so that they can share or go back later. This action is described in the use case US5.

US5: As a user, I want to be able to save the MoLIC diagram I am currently working on so that I can keep my work on my computer.

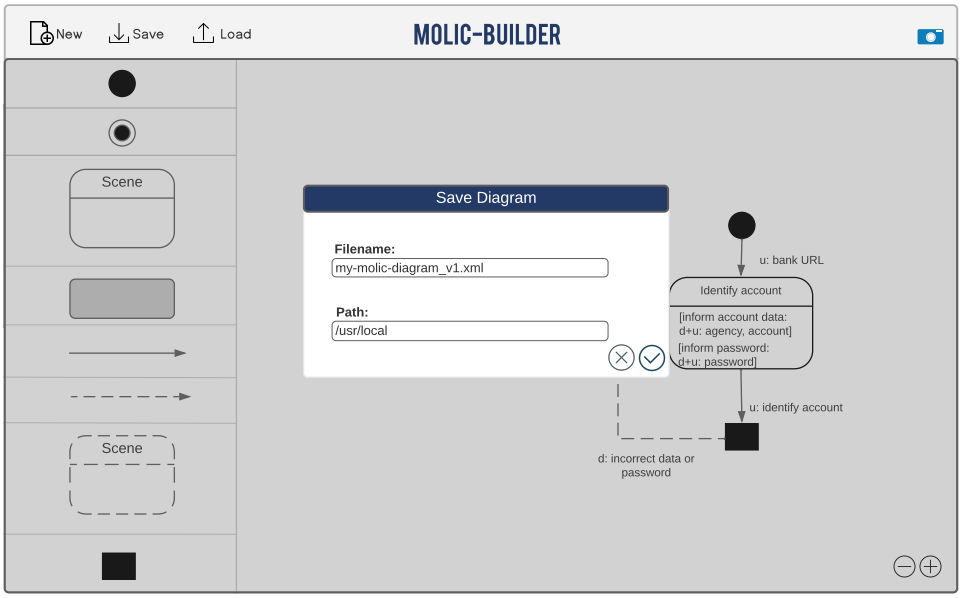

Figure 7: Save Current Diagram

When a user decides to leave the application, he receives a notification that the work can be lost, as described in use case US6 and shown in figure 8. This dialog asks whether he wants to save it or not.

US6: As a user, I want to be notified about the possibility of saving my diagram when I decide to leave the application.

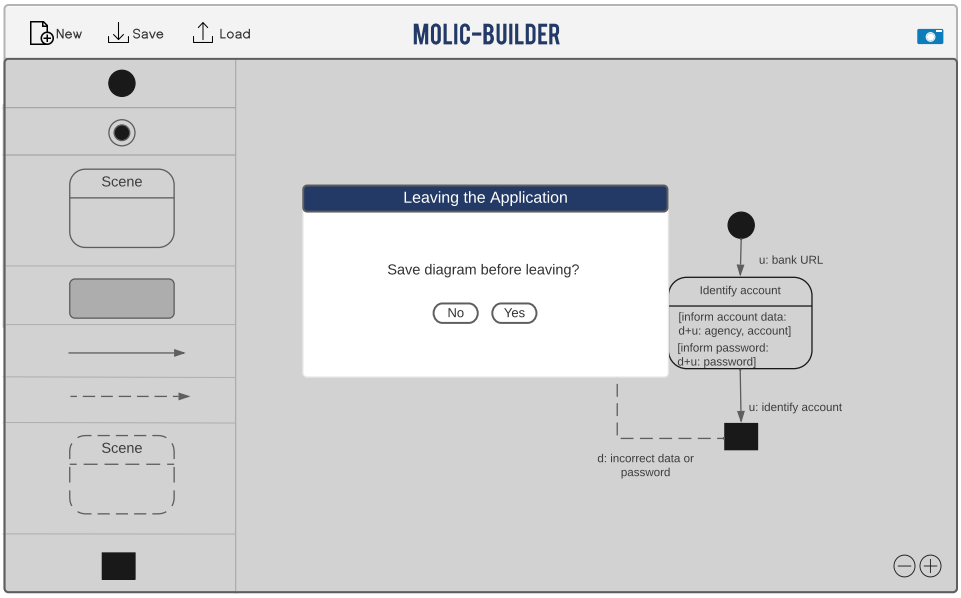

Figure 8: Select Leaving Option

In figure 8, if the user decides to save the work, the use case US5 is executed right after, allowing him to decide where this diagram file will be saved.

Another use case, is US6, which allows the user to drag the elements of the diagram on the canvas, in order to dispose the elements as desired by user.

3 Definition of Done

The user should be able to create, screen capture, export, and import the following MoLIC diagram:

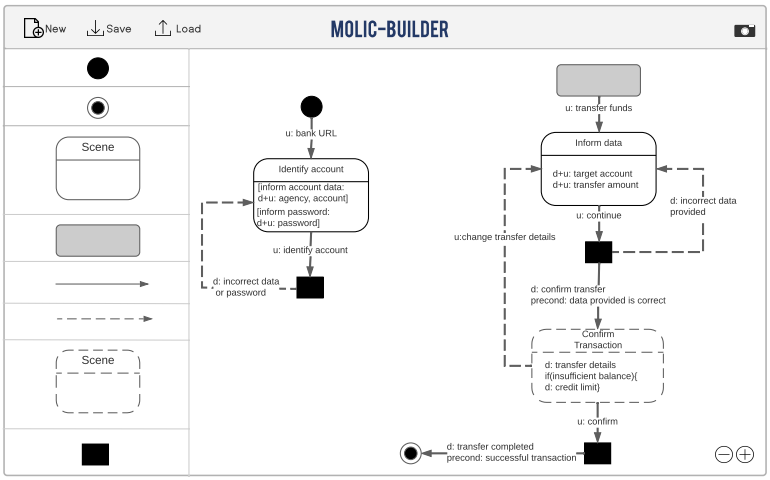

Figure 9: The diagram that should be able to create when the application is finished

Important Note: as stated in section 2, the scope of this Final Programming Project includes only User Stories 1, 3 and 6. Which consists of creating a diagram and connecting elements on it using turn giving elements.

4 Architectural Project

4.1 Class Diagram

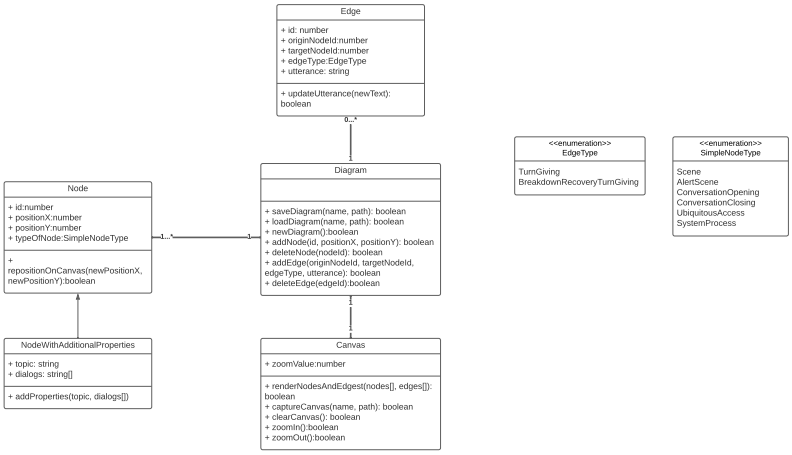

Figure 10: Class Diagram

- **Edge**: represents the concept of relationship between two nodes.

- **Node**: represents an abstraction of the types of elements (scene, start, ubiquitous access).
- **NodeWithAdditionalProperties**: represents a specialization of Node to include additional properties (topic and dialogs).
- **Diagram**: represents an abstraction of the diagram where all the elements are connected.
- **Canvas**: represents an abstraction of the canvas element where the diagram is rendered.
- **EdgeType and SimpleNodeType**: enumeration classes to represent each group of elements in the application.

4.2 Data Model

The MoLIC diagram will be represented in the system using a XML structure, allowing the system to convert the visual object into a textual representation.

Figure 12 is a translation of the visual model presented in figure 11 to an XML representation.

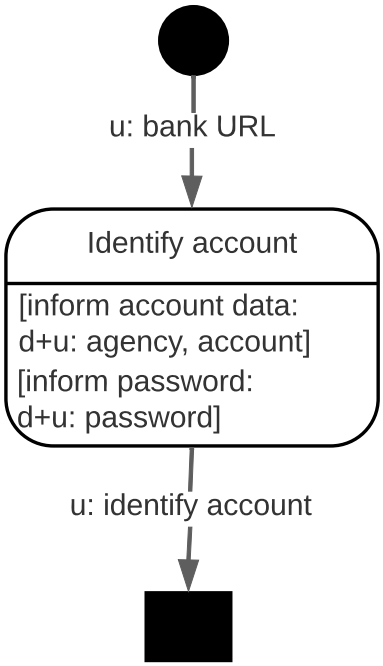

Figure 11: MoLIC Diagram Snippet

```
<?xml version="1.0" encoding="UTF-8"?>
<diagram>
	<name>Lucy's First Diagram</name>
	<id>1</id>
	<creationDate>2021-06-01 10:27</creationDate>
	<lastSaved>2021-06-01 10:27</lastSaved>
	<nodes>
		<node>
			<id>1</id>
			<positionX>210px</positionX>
			<positionY>100px</positionY>
			<typeOfNode>conversation_opening</typeOfNode>
		</node>
		<node>
			<id>2</id>
			<positionX>210px</positionX>
			<positionY>300px</positionY>
			<typeOfNode>scene</typeOfNode>
			<additionalProperties>
			<topic>Identify account</topic>
			<dialog>
				[inform account data: d+u: agency, account]
				[inform password: d+u: password]
			</dialog>
			</additionalProperties>
		</node>
		<node>
			<id>3</id>
			<positionX>210px</positionX>
			<positionY>500px</positionY>
			<typeOfNode>system_process</typeOfNode>
		</node>
	</nodes>
	<edges>
		<edge>
			<edgeId>1</edgeId>
			<originNodeId>1</originNodeId>
			<targetNodeId>2</targetNodeId>
			<edgeType>turn_giving</edgeType>
			<utterance>u: bank URL</utterance>
		</edge>
		<edge>
			<edgeId>2</edgeId>
			<originNodeId>2</originNodeId>
			<targetNodeId>3</targetNodeId>
			<edgeType>turn_giving</edgeType>
			<utterance>u: identify account</utterance>
		</edge>
	</edges>
</diagram>
```

Figure 12: MoLIC Diagram Snippet XML representation

4.3 Main Technologies Used

- **FrontEnd**: Angular (Framework for TypeScript)

- **Backend**: Node JS
- **Unit Tests**: Jasmine

5 Tests

This project will continuously be tested by unit tests created along with the application.

6 How to Download and Execute

In order to run this project in a computer, it has to have installed the following software: **Angular** (version: 10.1.1), **Node** (version: v10.18.0) and **Git** (any version).

Once the computer has the latter prerequisites installed, clone this project's repository using the following command in terminal:

```
git clone https://github.com/carolineloppi/molic-editor.git
```

When the cloning step is completed, type in terminal the following:

```
cd molic-editor
```

Now in the project's folder, install all the dependencies needed by the project using the command (also in terminal):

```
npm i
```

Finally, with the dependencies installed, the project can be run using the following command in terminal:

```
ng serve -o
```

A browser should automatically be opened with the application running!

Important Note: the latest version of MoLIC Editor is available in this link: `https://molic-editor.herokuapp.com/`

7 Future Work

As future work, the following user stories could be included:

- As a user, I want to be able to delete elements from canvas, so that I can remove objects from the diagram.

- As a user, I want to be able to zoom in and out the MoLIC diagram under edition, so that I can focus on a specific area of the diagram.

Another improvement to this project is the inclusion of utterance text validation.