MAKERERE UNIVERSITY

WIMEA-ICT Automatic Weather Station Setup Guide Application

By
BSE 19-03
WEB APPLICATION
DEPARTMENT OF NETWORKS
SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

A Project Report Submitted to the School of Computing and Informatics Technology for the Study Leading to a Project in Partial Fulfillment of the Requirements for the Award of the Degree of Bachelor of Science in Software Engineering of Makerere University.

Supervisor
Ms. Mary Nsabagwa
Department of Networks
School of Computing and Informatics Technology, Makerere University
mnsabagwa@cit.mak.ac.ug., +256 701 124388,

June, 2019

Declaration

We, group BSE 19-03, hereby declare that the work presented is original and has never been submitted for an award to any university or institution of higher learning

#	Names	Registration Number	Student Number
1	Ninsiima Grace	15 / U / 1004	215001025
2	Mwesigye Robert	15 / U / 771	215001423
3	Mawanda Henry	15 / U / 7496 / PS	215004553
4	Ssemagoye Umar Munddu	15 / U / 12812 / PS	215012010

Approval

This project report titled WIMEA-ICT Automatic Weather Station Setup Guide has been submitted for examination with my approval as the supervisor of group BSE 19-03

Ms. Mary Nsabagwa

Department of Networks

School of Computing and Informatics Technology;

College of Computing and Information Sciences,

Makerere University

Signature: ..... Date: .....

Supervisor

Dedication

We devote this work to WIMEA-ICT, without whose sustenance it would not have been possible, and to our parents and guardians who have supported us in all financial and parental matters as regards to our education.

Acknowledgements

One of the great pleasures of writing any report is acknowledging the efforts of many people whose names might not appear on its cover, but whose work, cooperation, friendship and understanding were crucial to its success. Needless to say, we are sure we stressed out a bunch of people along the course of this project and without their constant effort this project would have tasted no success.

Most importantly, we thank the Almighty God for the life He has given us and the constant provision that enabled us to finish this final year project.

We express our deepest gratitude to all those who provided us chance to complete this report. Special thanks go to our final year project supervisor, Ms. Mary Nsabagwa, for her pearls of wisdom she gave to us during the course of this project. Her encouragement helped us to coordinate our project in all stages of software development life cycle.

Furthermore, we would also like to acknowledge with much appreciation the crucial role of WIMEA-ICT staff, which gave us permission to use all the required equipment, relevant documents, the AWS devices and the lab to carry out lab demonstrations to complete the project. Thanks also go to the team itself for that has worked in unity without faint to have the application developed.

Lastly, we thank the guidance given to us by other supervisors as well as the panels especially in our project presentation that has improved our presentation skills.

Abstract

This report follows the implementation of WIMEA-ICT Automatic Weather Station (AWS) Setup Guide system from the inception phase to its completion. The system provides a simulation approach to setting up WIMEA-ICT AWSs without practically doing so in the field. It also provides information about the technologies employed during development of AWSs. The developed system shall be used by WIMEA-ICT AWS deployment teams, AWS deployment personnel in meteorological authorities in Uganda, South Sudan and Tanzania and any individual who would love to study about WIMEA-ICT AWSs.

Blog Site

Project Blog: https://aws-setup-guide.bitrix24.site

Table of Contents

Software Design Document

BSE19-3

WIMEA-ICT AWS Setup Guide

Software Design Document

Team: **BSE19-3**

Name	Registration Number	Signature
Mwesigye Robert	**15 / U / 771**	
Ninsiima Grace	**15 / U / 1004**	
Ssemagoye Umar Munddu	**15 / U / 12812 / PS**	
Mawanda Henry	**15 / U / 7496 / PS**	

Date: 05 Feb, 2019

1 Introduction

This document contains the system design specifications for the WIMEA-ICT Automatic Weather Station (AWS) Setup Guide Application. The application shall be used for simulating the assembling process of the WIMEA-ICT AWSs, developed under the WIMEA-ICT project [1].

1.1 Purpose

This design document is intended to provide the architectural and system design of the WIMEA-ICT AWS Setup Guide to programmers who have interest in implementing the system. It is intended to satisfy requirements as set in the System Requirements Specification Document (SRS) [2].

1.2 Scope

The WIMEA-ICT AWS Setup Guide shall be a web application intended to be used by the meteorological services in Uganda [3], Tanzania [4] and South Sudan [5] in simulating the setting up / assembling process of the various components of the WIMEA-ICT AWSs. The WIMEA-ICT AWS is made up of the gateway; a device that transmits AWS data to a remote repository and four nodes for capturing data. i.e., two-meter node, ten-meter node, sink node and the ground node. The application shall therefore cover the simulation of setting up/ assembly of the WIMEA-ICT AWS. It will also enable users to assemble the AWS nodes and the Low Power Gateway (LPG) themselves.
Users shall be with tools that make up the WIMEA-ICT AWS and shall be required to identify the parts. Later on, users' knowledge of assembling the AWS shall be tested using drag and drop features. Users shall also be provided with an automated way of assembling, which does not require them to interact but only watch the process. This shall especially be useful for first time users. Tutorials on technologies used such as IEEE 802.15.4 [6] shall also be loaded in the emulator in order to provide more information on how the technologies apply to the operations of the AWS.
The application source code will be stored on the Github repository for WIMEA-ICT project [7] to enable users and other developers to access it and modify it where necessary.

1.3 Document Overview

This design document is written following the IEEE standard [8]. It is divided into eight sections in order to provide a complete perception about the system to the target readers. The **first section** is mostly about the system scope and the purpose of this document. **Section two** provides the system

overview and a general description of the system and its functionality. The **third section** provides the system architecture, which comprises the architectural design decomposition and the reasons for the choice of the design.

The organization of the data of the system is presented in **section four**, a data dictionary is also provided to give a detailed description of the system's major data. **Section five** describes the component design of the system. It provides a functional description of each component in detail and a summary of some of the algorithms for the functionalities specified in the system's SRS. The **sixth section** provides an overview of the user interface design and describes the functionality of the system from the user's perspective. In addition, some possible screen images and objects are provided and purpose of the screenshots explained.

In the remainder sections of this document, the cross-reference that traces components and data structures to the requirements specified in the SRS document is provided in **section seven**. The appendices are provided in **section eight**.

1.4 Reference Material

[1] WIMEA-ICT, “WIMEA-ICT project,” 2016. [Online]. Available: https://wimea-ict.net. [Accessed: 09-Oct-2018].

[2] BSE19-3 Group-3, “WIMEA-ICT AWS Setup Guide SRS,” 2018.

[3] UNMA, “Uganda National Meteorological Authority,” 2018. [Online]. Available: http://www.unma.go.ug. [Accessed: 05-Oct-2018].

[4] TMA, “Tanzania Meteorological Agency,” 2018. [Online]. Available: http://www.meteo.go.tz. [Accessed: 05-Oct-2018].

[5] UoJ, “University of Juba,” 2014. [Online]. Available: http://jubauni.net. [Accessed: 05-Dec-2018].

[6] I. Howitt and G. Jore A., *IEEE 802.15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, vol. 2011, no. September. 2011.

[7] WIMEA-ICT, “WIMEA-ICT github repository,” 2015. [Online]. Available: https://github.com/wimea-ict. [Accessed: 08-Oct-2018].

[8] IEEE, “IEEE Recommended Practice for Software Engineering Terminology,” vol. 1998, 1998.

[9] V. J. Hodge, S. O. Keefe, M. Weeks, and A. Moulds, “Wireless Sensor Networks for Condition Monitoring in the Railway Industry: A Survey,” *IEEE Trans. Intell. Transp. Syst.*, 2015.

[10] J. Reuder and J. Sansa-otim, “WIMEA-ICT: Improving Weather Information Management in East Africa for effective service provision through the application of suitable ICTs,” no. November, pp. 1–6, 2013.

[11] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. 2013.

[12] W3Schools, “JavaScript Versions,” 2018. [Online]. Available: https://www.w3schools.com/js/js_versions.asp.

[13] BSE19-03; WIMEA-ICT;, “AWS Setup Guide github repository,” 2019. [Online]. Available: https://github.com/wimea-ict/AWS-Setup-Guide.git. [Accessed: 29-May-2019].

[14] M. H. Mwsigye Robert, Ninsiima Grace, Ssemagoye Umar, “AWS Setup Guide SDD approved,” p. 35, 2019.

[15] M. W. Mozilla, “Cross-Origin Resource Sharing (CORS).” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS%0A. [Accessed: 09-May-2019].

1.5 Definitions and Acronyms

Term	Details
AWS	**Automatic Weather Station**. A weather station which uses suitable ICTs to capture weather data.
DOM	Document Object Model
GUI	Graphical User Interface
ICT	Information Communication Technology
LPG	**Low Power Gateway.** The gateway device that relays the weather data over the Internet designed for generation 3 (Gen 3) AWSs developed by WIMEA-ICT.
Node or AWS node	The electrical sensing and transmitting device used on an AWS. WIMEA-ICT uses four nodes per AWS i.e., 2-meter node, 10-meter node, ground-node and sink node.
OOD	Object-Oriented Design.
OS	Operating System
PC	Personal Computer
SRS	**System Requirements Specification**. A document that contain the requirements for the development of a software related system.
UI	User Interface
Weather Station	A facility either on land or sea with instruments for measuring and recording atmospheric conditions to provide information for weather forecast and studying climate.
WIMEA-ICT	**Weather Information Management in East Africa** through application of suitable **ICTs**
WSN	Wireless Sensor Network. Self-configured wireless network that monitors physical or environmental conditions [9]
XML	eXtensible Markup Language. Similar to HTML but designed to store and transport data.
PNG	Portable Network Graphic

2 System Overview

2.1 Background Information

WIMEA-ICT [10] is a combined research and capacity building project coined by the Norwegian Agency for Development Cooperation (NORAD) under the Norwegian Program for Capacity Development in Higher Education and Research for Development (NORHED) scheme. The project partners and implementors include Makerere University (MUK) in Uganda, University of Juba (UoJ) in South Sudan, Dar es Salaam Institute of Technology (DIT) in Tanzania and the Geophysical Institute of the University of Bergen in Norway.

WIMEA-ICT has been designing, installed, and is maintaining Wireless Sensor Network (WSN) based AWSs since 2015. The first batch of AWSs was deployed in September 2018 in Uganda and deployment still continues in the three African countries. During the deployment exercise, a number of challenges were identified. These include limited technical skills of the observers, complex terminologies used, and damage of equipment used to train the AWS deployment among others.

The purpose of this project is to develop a graphical tool that will simulate the AWS setup process, which shall enable users understand the WIMEA-ICT assembly process. Components to be included in the simulation include; the sensor nodes and the LPG. The simulator is expected to help meteorology to set up the nodes of an AWS before physically doing so in the field. This will reduce on the many components used during the training programs, reduce on costs of training personnel and increase technical knowhow of setting up and assembling WIMEA-ICT AWSs.

2.2 System Description

The WIMEA-ICT AWS Setup Guide shall be designed to enable meteorologists learn how to set up an AWS without/before doing it practically in the field. It shall be composed of three major components i.e., Auto-assembly, Manual-assembly and Explore. Auto-assembly – where the system will show the user how the various components of an AWS are connected. Manual-assembly – which shall require the user to assemble/set up the AWS node components by themselves through drag and drop. The Explore (About WIMEA-ICT AWSs), shall contain tutorials and reference material explaining in detail, the technologies employed by WIMEA-ICT project.

On loading the system, users will be able to choose the mode in which he/she would wish to run it. Following are the modes**;**

- **Auto-assembly mode**: the system will ask the user to choose the kind of AWS node he/she wishes to simulate automatically. On selecting the desired node, the system shall load all the AWS components required for the simulation of the chosen node. Under this mode, the user may choose to reduce/increase the volume of the audio description of the simulation, adjust the speed of the simulation and pause and resume the simulation by clicking on the available button on the interface.
- **Manual Assembly mode**: the system will prompt the user to select the kind of AWS node he/she would wish to assemble. The system will then load all the necessary AWS components required to setup/assemble the desired node. The user can then drag and drop the components to the drawing stage (canvas) and make connection between them (see Figure 6.4). The system shall award points for correctly connected components and also give rejection messages for wrong connections.
- **Explore mode:** In this mode the system shall display the nodes and when the user clicks on a given node, the system will display the components that makes it up and a short description of each. The user can click on a particular AWS component to view its details and its various interfaces. The user can also click on the AWS component's link to view more information about it on the internet via the browser.

The user can also choose to change the mode of operation of the system at any time.

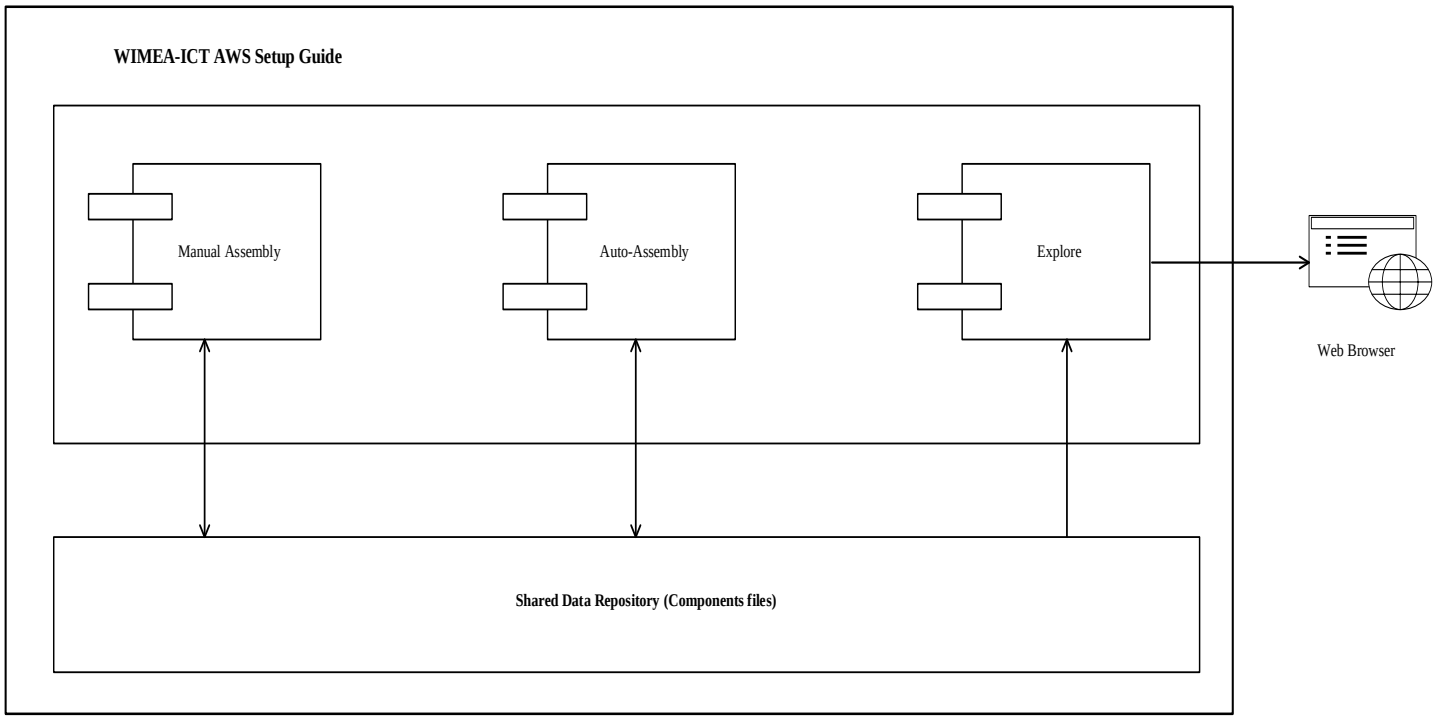

Figure 2. 1 Overview of WIMEA-ICT AWS Setup Guide

3 System Architecture

This section discusses the architectural structure for WIMEA-ICT AWS Setup Guide. It provides the architectural design, the decomposition description for each system-component/module and the reasons for the choice of the architecture.

3.1 Architectural Design

The architecture of WIMEA-ICT AWS Setup Guide shall be based on Shared-data Repository architectural pattern [11, p.27]. It consists of components that allow the WIMEA-ICT AWS Setup Guide application to interact with the host environment. Figure 3.1 shows the architectural diagram for the WIMEA-ICT AWS Setup Guide.

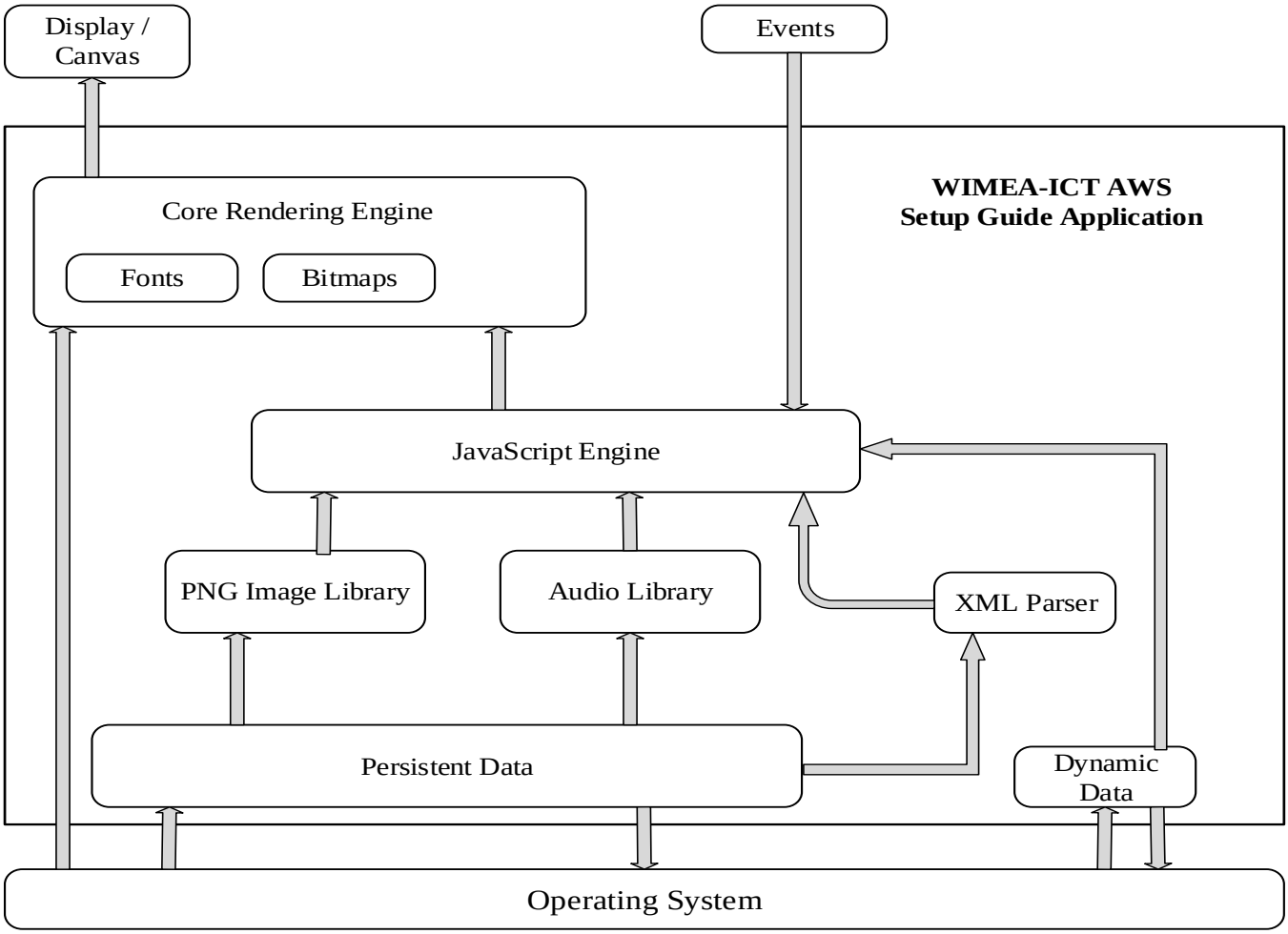

Figure 3. 1 Architecture of WIMEA-ICT AWS Setup Guide

The rendering Engine; Is responsible for the composite visual elements for displaying on the screen and the host computers' devices like speakers and microphones (for audio outputs).

JavaScript Engine; the JavaScript engine interacts with the rendering engine to process events (e.g. key presses, mouse clicks) to enable dynamic interactivity in the application. It also structures content in a way that is most appropriate for the rendering engine as define by the designers.

PNG Image Library; this is responsible for the .png images (manipulating them, converting them to bitmaps etc.) that shall be displayed and used by the application.

Audio Library; Is responsible for manipulating and playing the embedded audio files that are used by the application. This shall be used in the WIMEA-ICT AWS Setup Guide application during Auto-assembly process.

XML Parser Engine; this deals with processing and loading the xml. This component is vital to our application as it will be used to parse the component descriptions and load images of the components to the rendering engine.

Persistent Data; this holds data used by the application, in this case, it shall contain XML files holding pointers to the locations of the AWS components images, the AWS component descriptions and properties which are used by the application. It shall also contain the component images and their corresponding bitmaps.

Dynamic Data; is responsible for temporarily holding data that changes during run-time of the application. In our case, data like the component connection vectors and awarded points during Manual assembly shall be held by this component.

Events; these are external to the application and shall be generated instantaneously. For example, button presses, mouse clicks, hovering, double clicks events etc. which are used during system navigation and use.

Device Operating system; this is also external to the system; it mediates interaction between the application and the host computer's hardware.

3.2 Decomposition Description

WIMEA-ICT AWS Setup Guide takes its design from the Object-Oriented approach. This sub section therefore provides a high-level view of the system in terms of a component diagram showing the functions achieved by each system-component. It then provides a breakdown showing the major tasks of the system using the activity diagram.

3.2.1 Activity diagram

This subsection provides activity diagrams for the Use Case Narratives of the system specified in the SRS [2]. It shows the different activities and sub-activities and how they are carried out to achieve the functional requirements of the system.

3.2.1.1 System Activity Diagram

This shows the high-level view of the system, figures 3.3 up to 3.5 show the breakdown of the sub-activities.

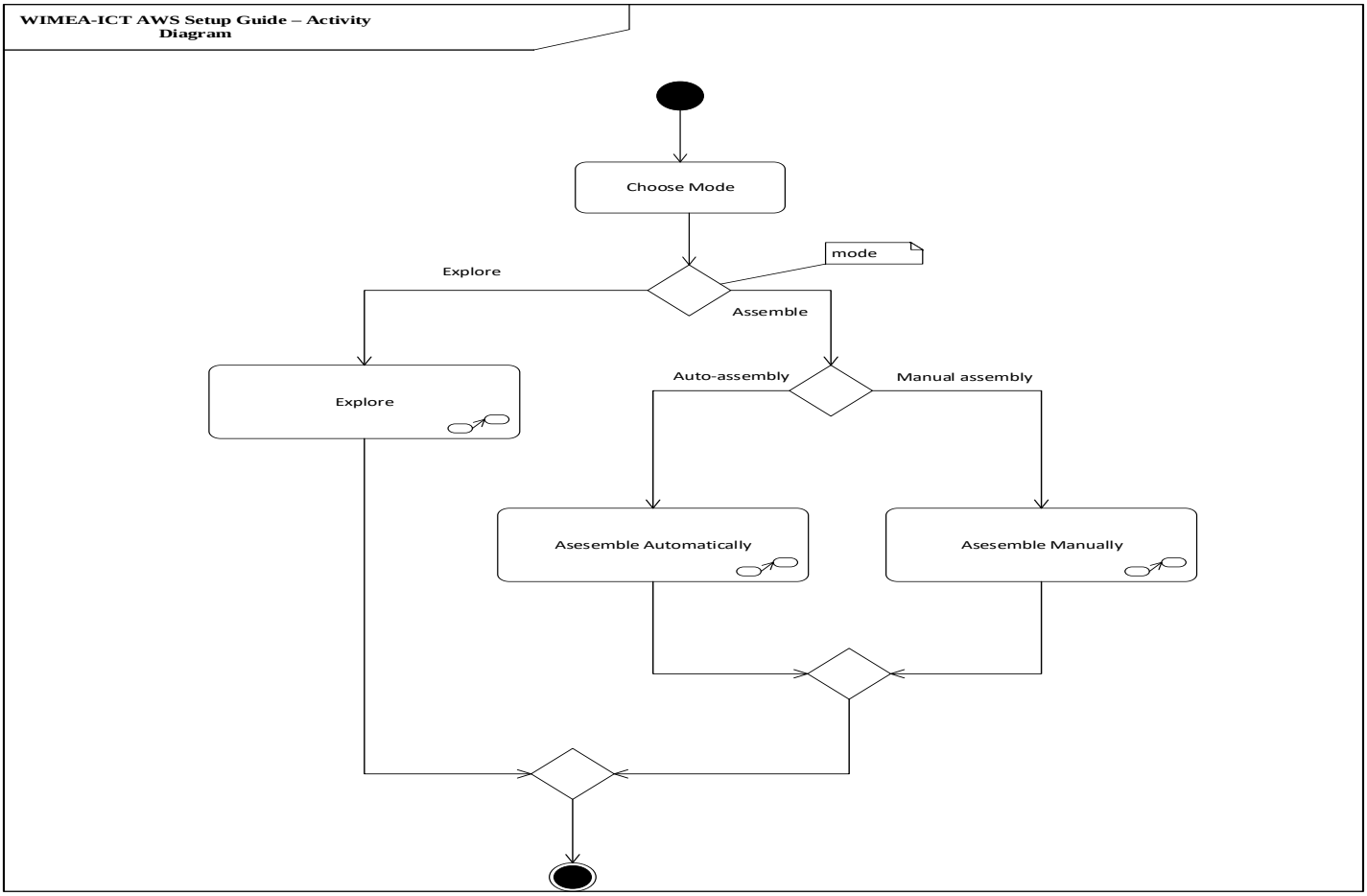

Figure 3. 2 WIMEA-ICT AWS Setup Guide Activity Diagram

3.2.1.2 Manual Assembly Activity Diagram

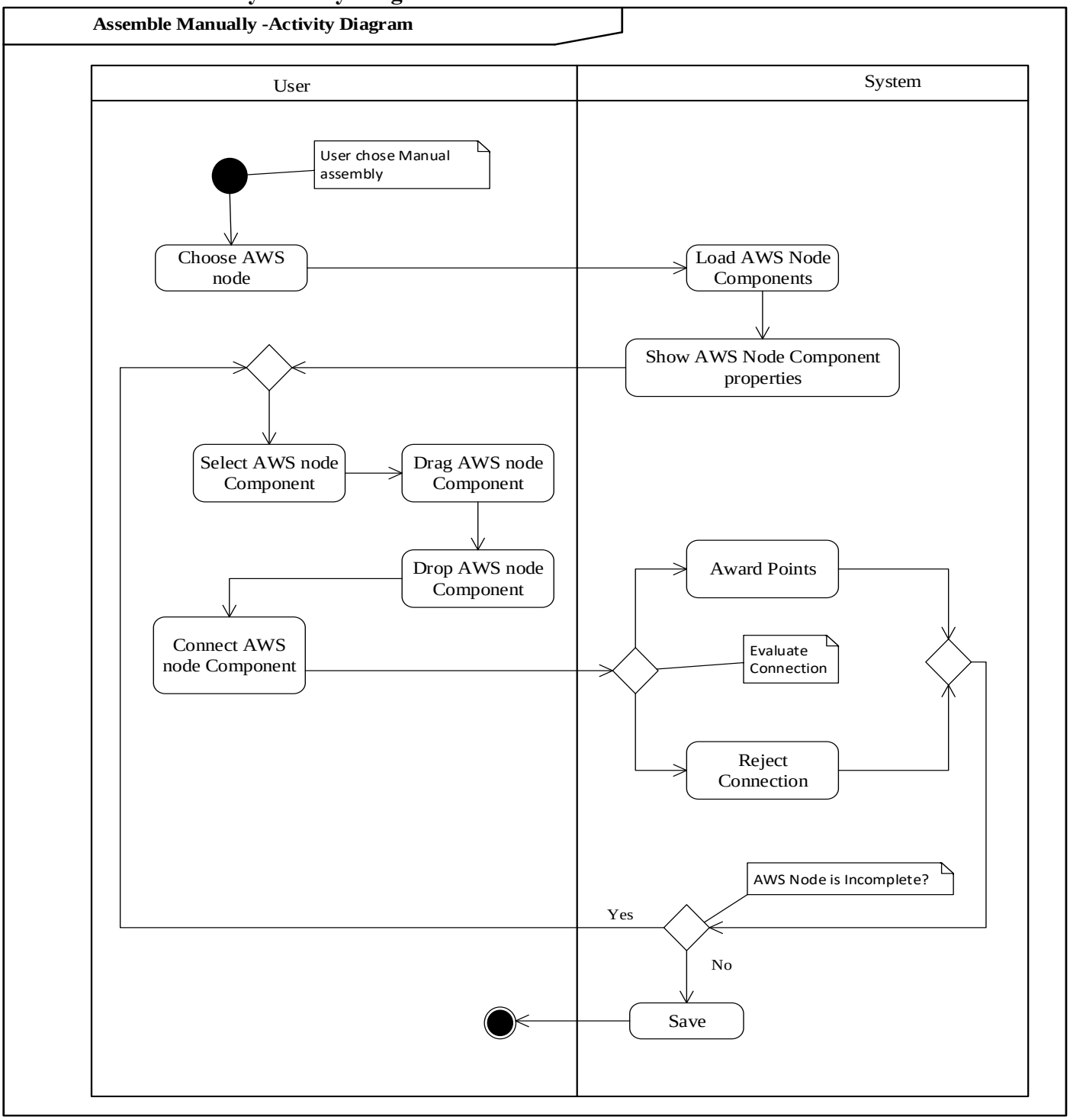

Figure 3. 3 Manual Assembly Activity Diagram

3.2.1.3 Auto-assembly Activity Diagram

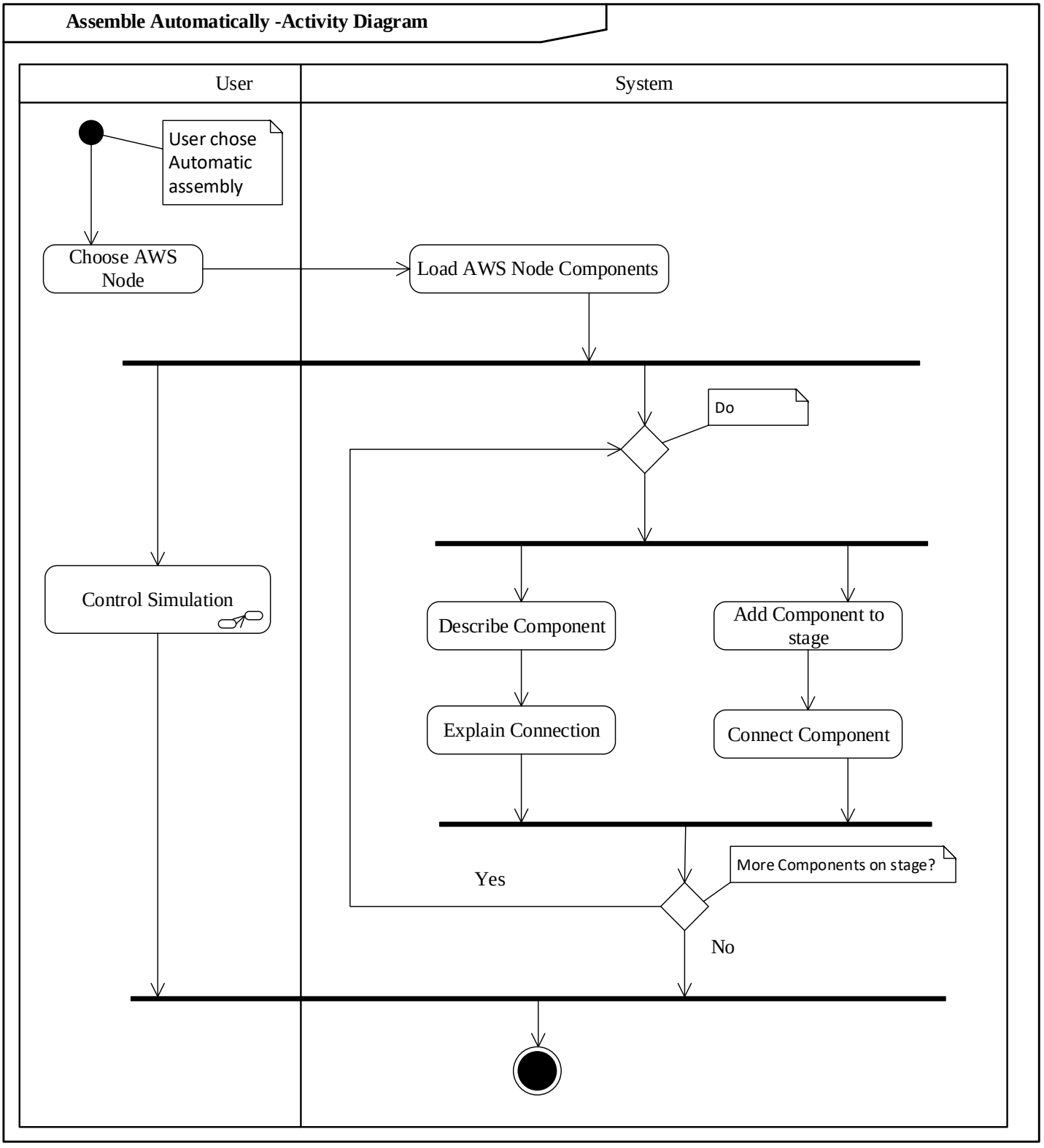

Figure 3. 4 Auto-assembly Activity Diagram

3.2.1.4 Explore Activity Diagram

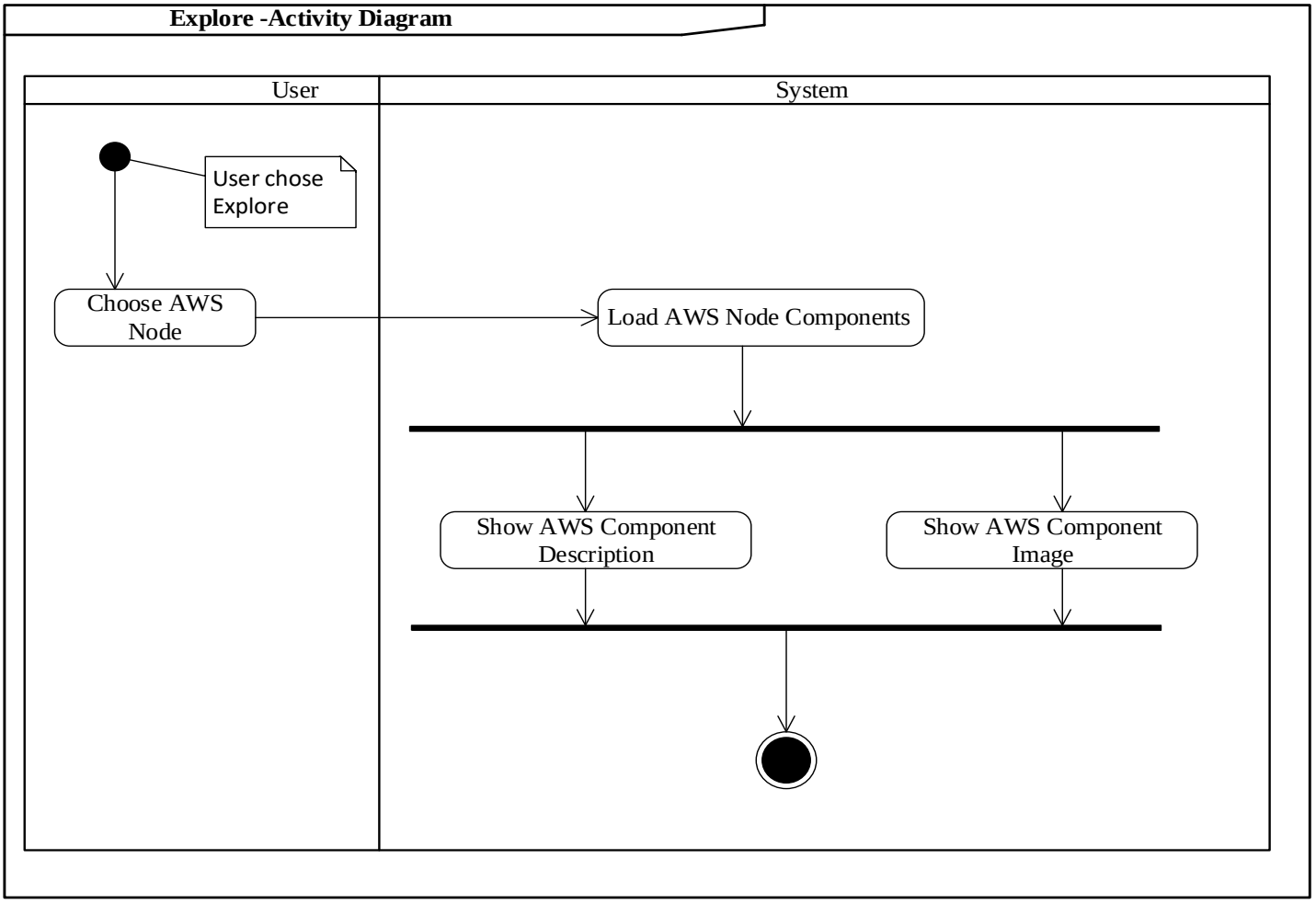

Figure 3. 5 Explore Activity Diagram

3.3 Design Considerations

3.3.1 Assumptions

i. The users of the WIMEA-ICT AWS Setup Guide are aware of basic operations of the computer and web-pages.
ii. Users have some background information about WIMEA-ICT AWS technologies.
iii. The user should have installed a browser on his/her computer. Examples of supported browsers include later versions of; Safari 4, Firefox 40.x.x, Internet explorer 5, Microsoft edge, Google Chrome 48.x.x.x.
iv. The system shall not require internet to run Auto-assembly and Manual assembly activities.

v. User has internet access in order to load and view the AWS component from the internet using a browser. This shall be required in case the user wants to get more details about the AWS components.

3.3.2 Constraints

i. The system will be implemented using JavaScript, HTML5 and CSS3 technologies.

ii. The system will run in a web browser.

3.4 Design Rationale

The major reason for choosing a shared data repository architectural pattern is attributed to the fact that the system components make use of the data independently. Since the system can only be loaded in one mode at a time, this independence of components calls for a data centric architecture to ensure high cohesion.

During the implementation stage, we shall use JavaScript 6 [12] which is an object-oriented language, this compelled us to use object-oriented design approach. Furthermore, an object-oriented system tends to model the real world in a more complete fashion than do traditional methods such as functional design approach. Object oriented design approach improves system reliability and flexibility since new behaviors can be built from the existing objects.

The choice for JavaScript is attributed to the fact that it is client sided, therefore, the system shall not require any server. The application being web based, it runs in a browser thereby saving the user from any computer system special requirements to run the application.

The user interfaces shall be developed using HTML5 and CSS3 which makes the user interfaces light and easy to load, easier to add drag and drop functionality, support for audio and support for legacy and cross browsers.

4 Data Design

4.1 Data Description

Each AWS node has a predefined set of components, for this reason all the AWS components shall be implemented as an object. Each AWS node component shall be saved as an image and shall be loaded via XML processing classes defined by the implementation language (JavaScript in this case). JavaScript being Object-Oriented defines everything in it as an object therefore an object will be used as the lowest data representation of AWS component.

Each AWS Component data shall be stored in an XML file (saved in the name of that particular component e.g. RSS2_mote.xml for the RSS2 Mote component). The figure 4.1 below shows the structure of the XML file for each AWS component.

```xml
<?xml version="1.0" encoding="utf-8"?>
<component>
    <!-- COMPONENT NAME -->
    <name> Name of the Component</name>

    <!-- COMPONENT IMAGES -->
    <image title="general"> path to the location of the main image</image>
    <!-- By interface we mean the specific parts on the component where connections are made -->
    <image title="interface1">path to the image that shows the specific interface 1</image>
    <image title="interface2">path to the image that shows the specific interface 2</image>
    <image title="interfacen">path to the image that shows the specific interface n</image>


    <!-- COMPONENT PURPOSE -->
    <purpose>purpose of the  component</purpose>

    <!-- COMPONENT PROPERTIES -->
    <properties>
        <property1 title="propert1"> property 1</property1>
        <property2 title="propert2"> property 2</property2>
        <propertyn title="propertn"> property n</propertyn>
    </properties>

    <!-- COMPONENT DESCRIPTION -->
    <description>
        The description of the component goes here
    </description>

    <!-- LINK TO COMPONENT'S DETAILS ONLINE -->
    <link> valid link pointing to the detailed description of the component</link>

</component>
```

Figure 4. 1 Structure of XML file for an AWS Component's data

Data files for each component including; interface images, main image, audio description, component xml file and any addition files related to a particular AWS component shall be saved in a folder in that AWS component's name. The figure 4.2 below shows the folder having data for RSS2 Mote.

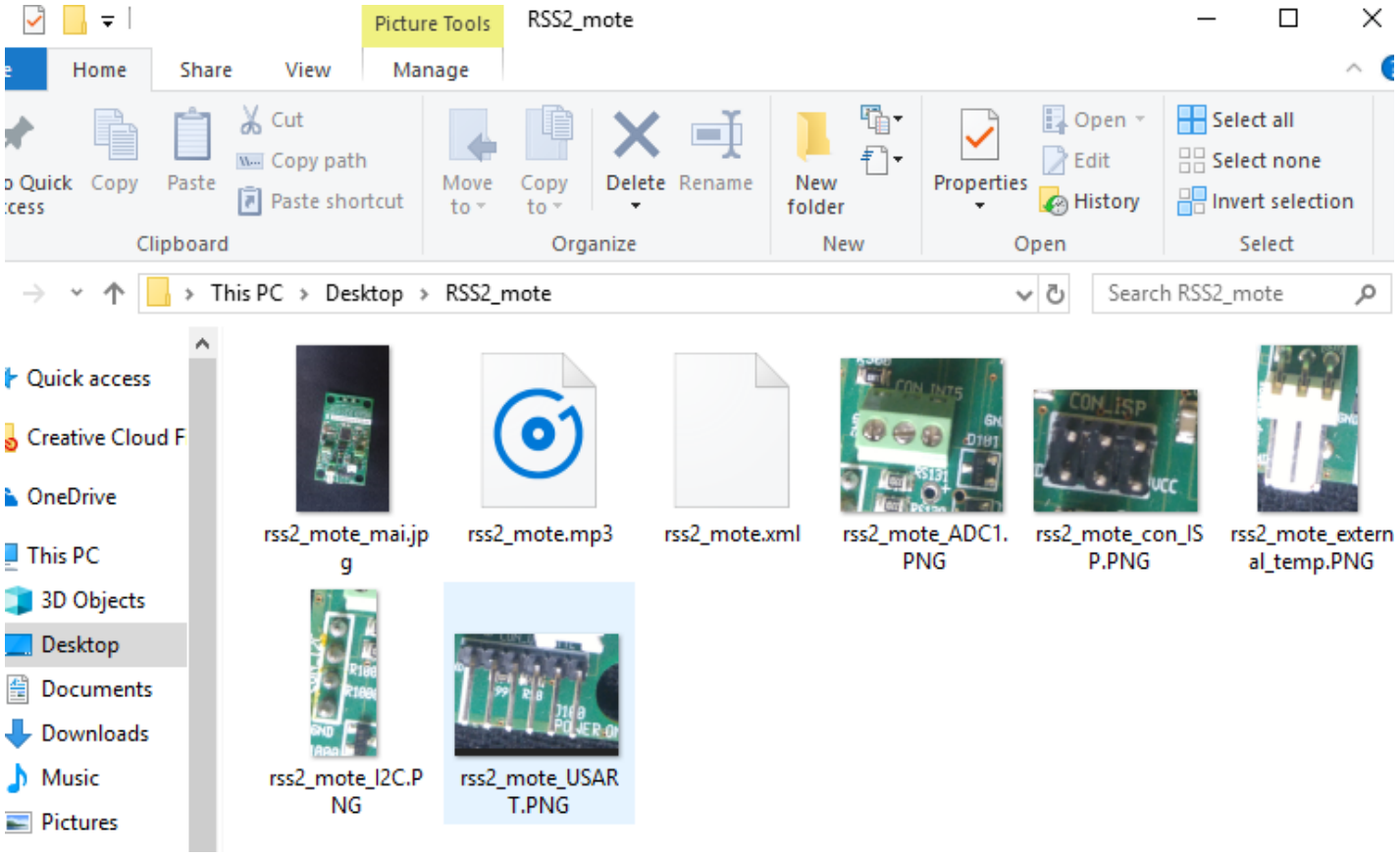

Figure 4. 2 Directory having data files for RSS2 Mote

4.1.1 Assembly Collaboration diagram

The collaboration diagram in figure 4.3 shows the assembly mode of how the objects in the system get created to achieve the Assembly functionality. When the user chooses either Auto or Manual assembly, the mode object gets created, it then loads the components that make up the required node.

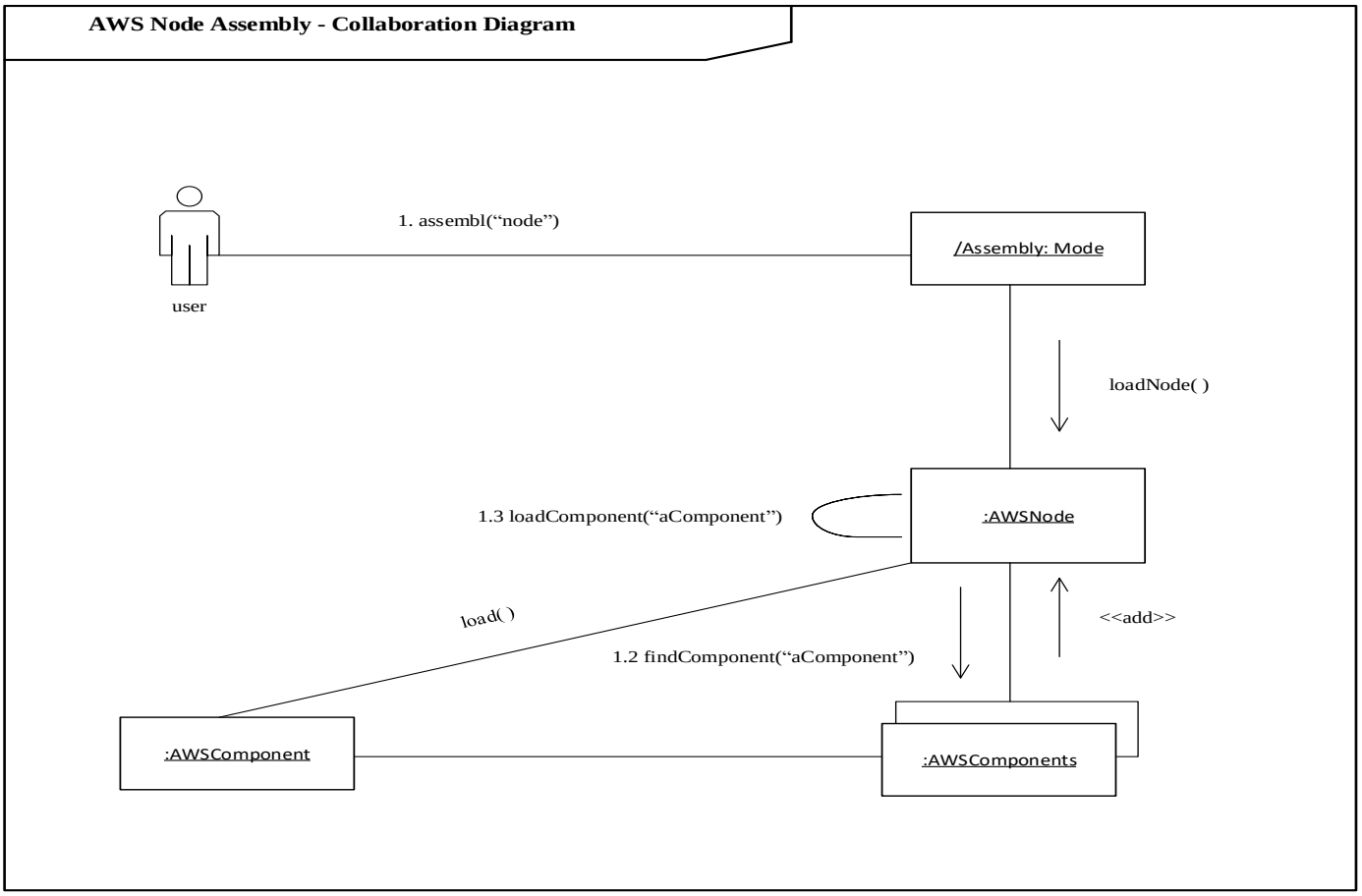

Figure 4. 3 WIMEA-ICT AWS Setup Guide Assembly Collaboration Diagram

4.2 Object Diagrams

JavaScript allows for the creation and use of objects therefore, the object diagrams provided in this section represent a series of object interactions that take place during the execution of the system.

4.2.1 XML Object Creation

The XML object creation object diagram shows how a raw XML file for an AWS node or AWS component is converted into an object representation i.e. *XMLObject*.This object holds data for a specific AWS node or AWS component.

The *XMLFilePath* object holds a pointer to the relative path of the XML file on the file system. The *XMLHttpRequest* object holds a reference to the *XMLFilePath* object in order to create the *XMLObject*.

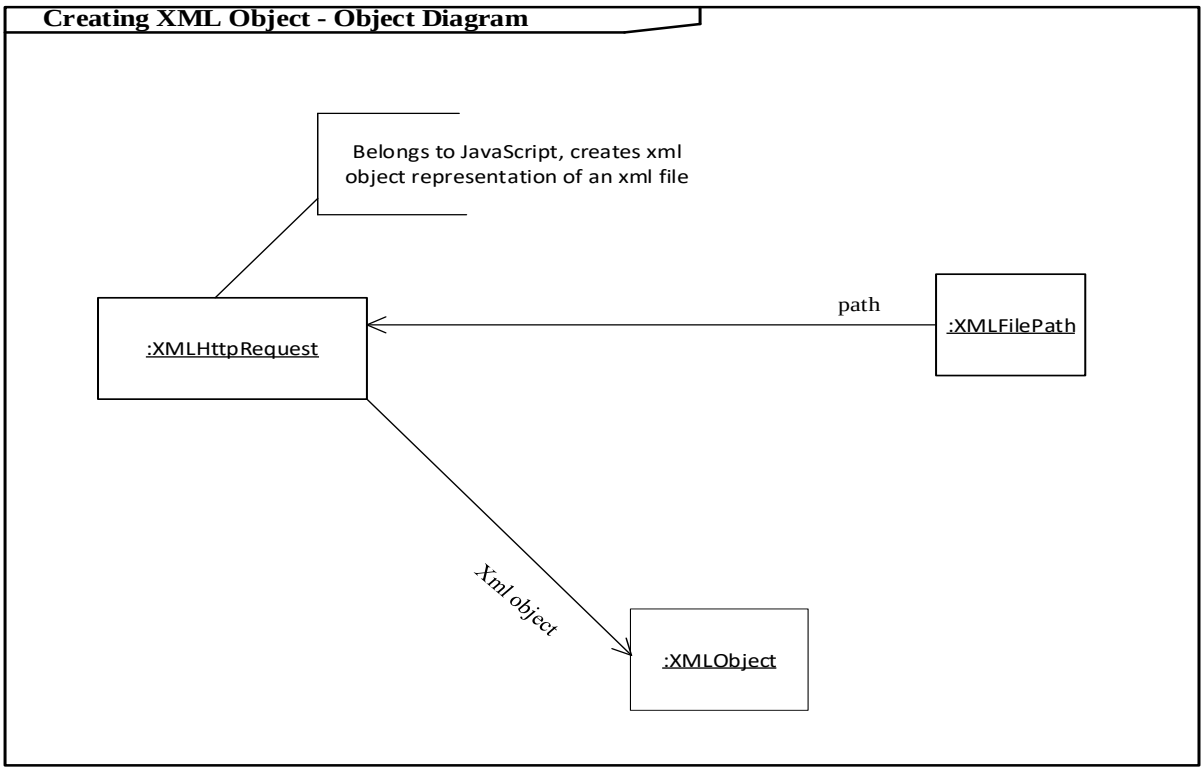

Figure 4. 4 Create XMLObject object diagram

4.2.1 Assembly Mode Creation

The assembly mode object diagram shows that a *Mode* object has a reference to the *Assembly* reference which in this case can be either Automatic or Manual. It also has a reference to *AWSNode* object representing the AWS node to be assembled. The *AWSNode* object has a reference to the *XMLObject* representing the node data from the XML file. An *AWSNode* object can have many references to different *AWSComponent* objects. Each *AWSComponent* object has a reference to the *XMLObject* representing its data from the XML file.

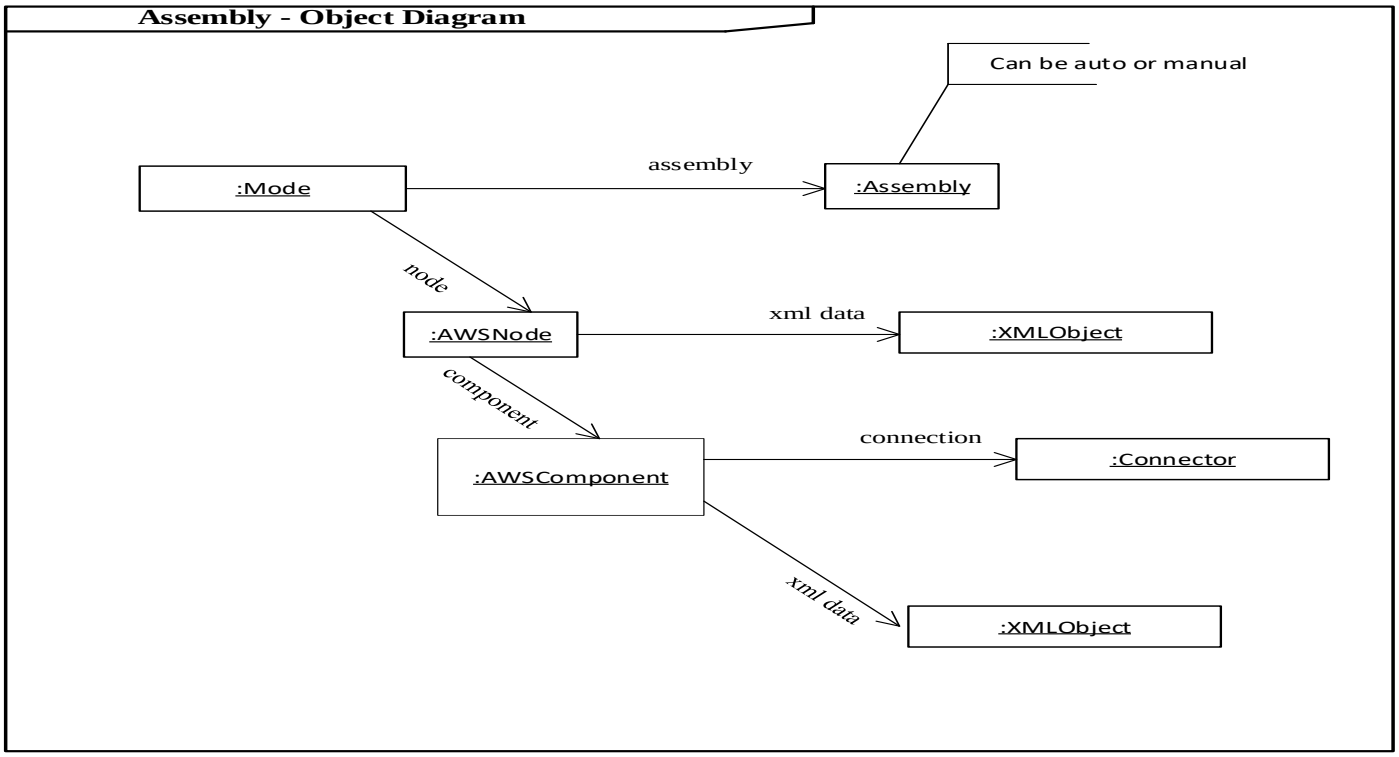

Figure 4. 5 Assembly Mode object diagram

4.2.1 Explore Mode Creation

Under explore mode, the *Mode* object holds a reference to the *Explore* instance. For the same case the Explore object holds a reference to the *AWSNode* object to be explored (i.e. explained or studied about).

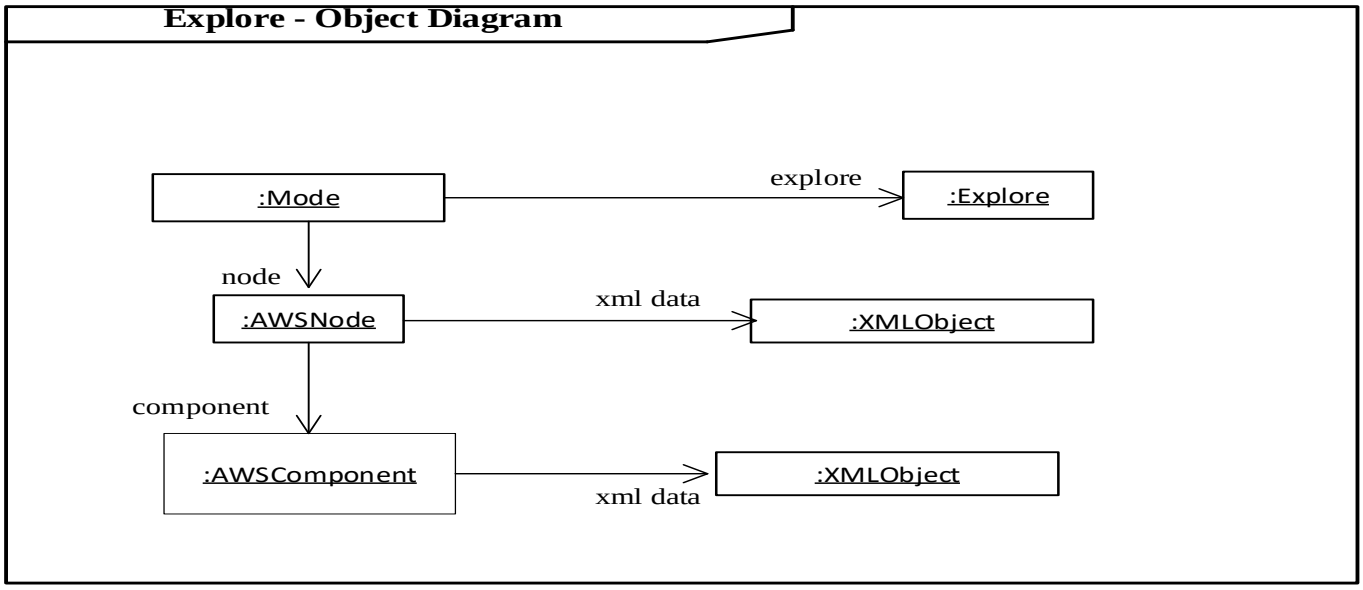

Figure 4. 6 Explore Mode object Diagram

4.3 Data Dictionary

This subsection describes the format, structure and contents of the data used by the WIME-ICT AWS Setup Guide application and their relationships. Data used by the system include audio files, images, textual data, symbols, bitmaps and vector graphics. The table 1 below presents the data dictionary for the system's data.

Table 1 Data Dictionary

Data	Type / format	Properties	Description
Images	Bitmap	Height = variable (depending on the AWS component itself) Pixels = 100	The images of components used during manual and automatic assembly
	PNG	size = 100 X 80cm Pixels = 100	Images to be converted to bitmaps and vector graphics. Also used in Explore mode
Audio	mp3	Item type = mp3 bitrate = 128kbps	The audio files that are played during the connection of components in Auto-assembly
Component Description	Text	encoding = utf-8	This shall be contained in the XML file (see structure in Figure 4.1). Information concerning a certain AWS component.
Component properties	Text	encoding = utf-8	Properties of an AWS component displayed to enable choosing of the appropriate component during assembly. Properties shall include voltage, size, color, resistance among others

5 Component Design

This section discusses the major system designs for the components in figure 4.3 following the best practices of design patterns and Object-Oriented design. It also provides sets of algorithms that can be used to implement the major system functionalities.

5.1 Algorithms

5.1.1 Main program

Start

I. initialize system's interface
II. ***If*** *Operation Mode = manual Assembly*
a) create manual assembly mode instance
III ***Else if*** *Operation Mode = auto Assembly*
b) create auto Assembly mode instance
IV. ***Else if*** *Operation Mode = explore*
c) create explore mode instance

Stop

5.1.2 Manual assembly

Start

I. load chosen node
II. For each component in the node
a) display component's name
b) display component's thumbnail on the components pane
End for each
III. Prepare canvas for connection of components
IV. **Event:** User selects Components, Drags component on canvas
move selected component to stage at the x and y position of the cursor
V. **Event**: User makes connection between Components
evaluate connection type.

Stop

5.1.3 Get component

Start

I. create an XML loader for the component xml file
II. Open the XML file in step I
III. Travers the open XML file in step II, search for all the elements
IV. Set the Component's properties from the xml file
V. set the Component's image location from the xml file
VI. Set the Component's name from the xml file
VII. Set the Component's description from xml file
VIII return the Component

Stop

5.1.4 Evaluate connection

Start

I. get connected interfaces
II. Load allowed connection types
III. Identify used connection type
IV. If: connected interface matches stored connection type
    a). Evaluate points to award
    b) set the points
V. Else:
    a) reject the connection type
    b) send rejection message

Stop

Events

Below are the algorithms for the events that happen from user interaction with the systems. Control automatic simulation involves clicking play, pause, stop and resume buttons, sliding speed and volume control sliders.

5.1.5 Speed Control Algorithm

Start

I. get the click event from the Forward/Rewind Button on the user interface
II. Set the frame rate (Frames per Second) value to the value got from step I
III. Proceed the simulation with new frame rate.

Stop

5.1.6 Volume Control Algorithm

Start

I. get the value from the volumeSlider on the user interface
II. get the audio channel stream whose amplitude is to be changed
III. Set the audio channel stream amplitude to the value from the volumeSlider in step I
IV. Proceed the simulation with new set volume

Stop

6 Human Interface Design

6.1 Overview of User Interface

The human interface design for the WIMEA-ICT AWS Setup Guide shall be based on the Graphical User Interface (GUI). Users will interact with the system to perform the Manual assembly, Auto-assembly and Exploration activities;

Start

At the launching of the system the user will have to first choose the mode in which to run the system, either Auto-assembly, Manual assembly or Explore. For Manual assembly and Auto-assembly, the user will have to choose the AWS nodes he/she would like to assemble or view being assembled respectively.

Auto-assembly

To view assembly process of any WIMEA-ICT AWS node, the user will launch the system in Auto-assembly mode. The system will load all the necessary components for the whichever node the user has chosen to be assembled automatically.

Manual assembly

For the user to be able to assemble the AWS components manually, he/she will load the system in Manual assembly mode. From this interface, the user shall select an AWS node component he/she would like to assemble, drag and drop the component(s) on the canvas and connects them. The system will evaluate each connection made and awards points for each correctly connected component interfaces.

Explore (About WIMEA-ICT AWSs)

On this interface, the user will click on a specific AWS node and the system will display the detailed textual description of that particular component. It also provides a link onto which the user can click to view the component's details in the browser from the internet.

6.2 Screen Images

6.2.1 Start (choose operation mode)

This interface will be loaded every time the system is started. From this interface, the user will be able to choose the mode in which the application should run, either *Auto-assembly* or *Manual-assembly* mode by clicking the preferred choice. It is on this interface where the user shall also be able to click the link which provides information about WIMEA-ICT AWSs i.e., Explore (*About WIMEA-ICT AWSs*).

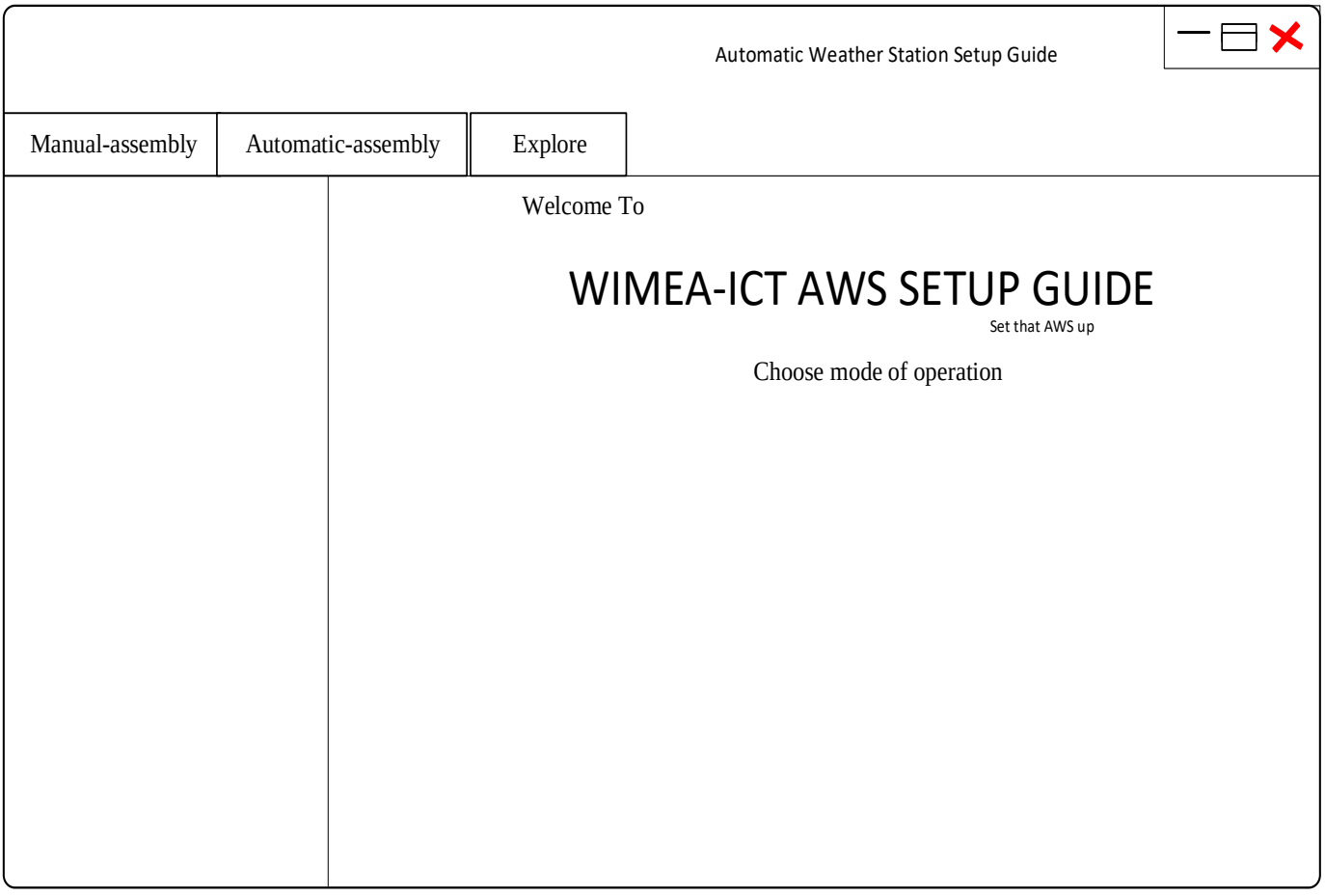

Figure 6. 1 Start Interface

6.2.3 Select node

On this interface, the user will select the kind of AWS node to assemble manually or simulate automatically depending on the chosen mode of operation.

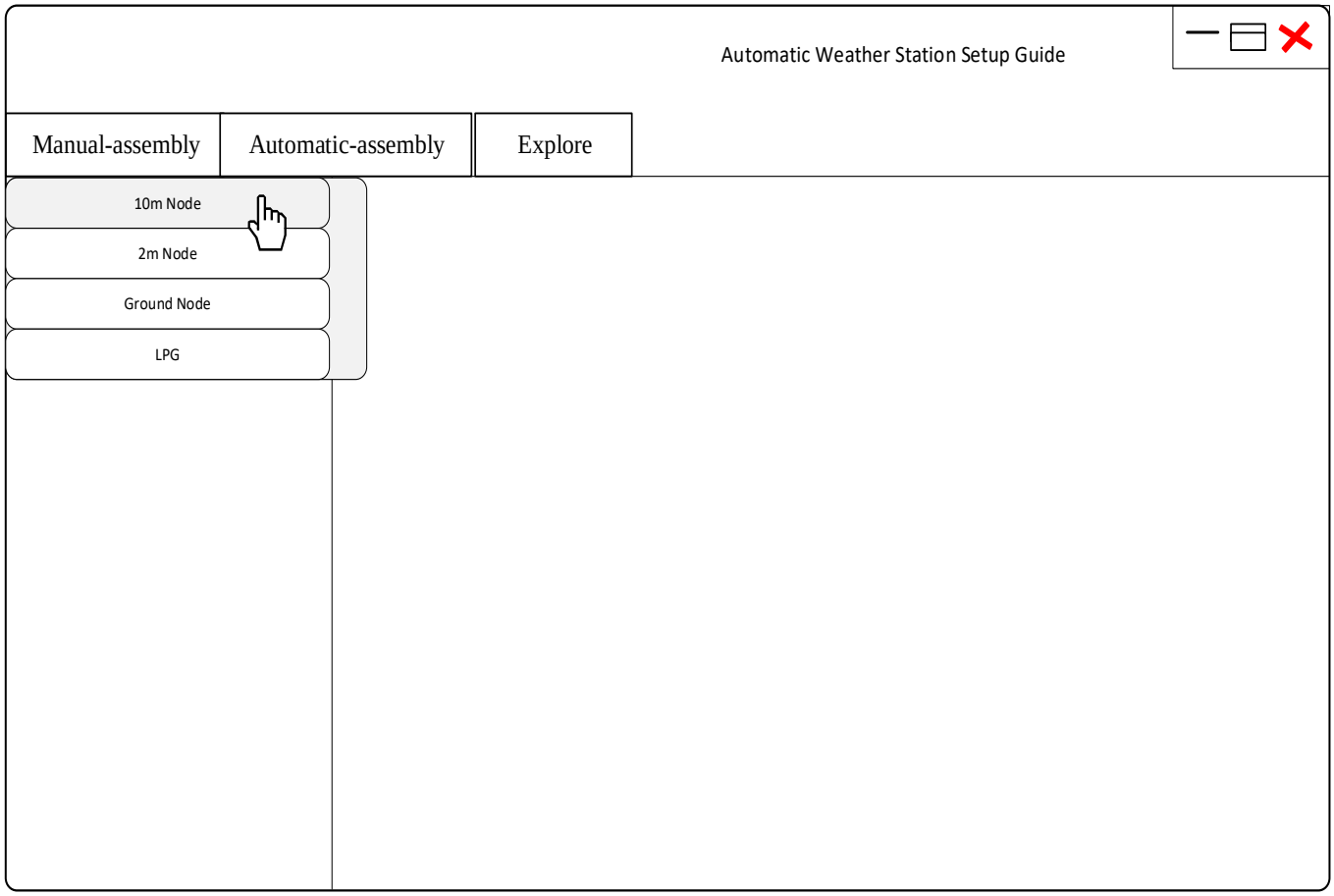

Figure 6. 2 Select Node Type

6.2.4 Auto-assembly

From this interface, users will view an automatic simulation of the AWS node or LPG assembly process. The user shall be able to start/stop, play/resume/pause, increase/decrease speed of the automatic simulation, reduce/increase the volume of the audio output of the simulator.

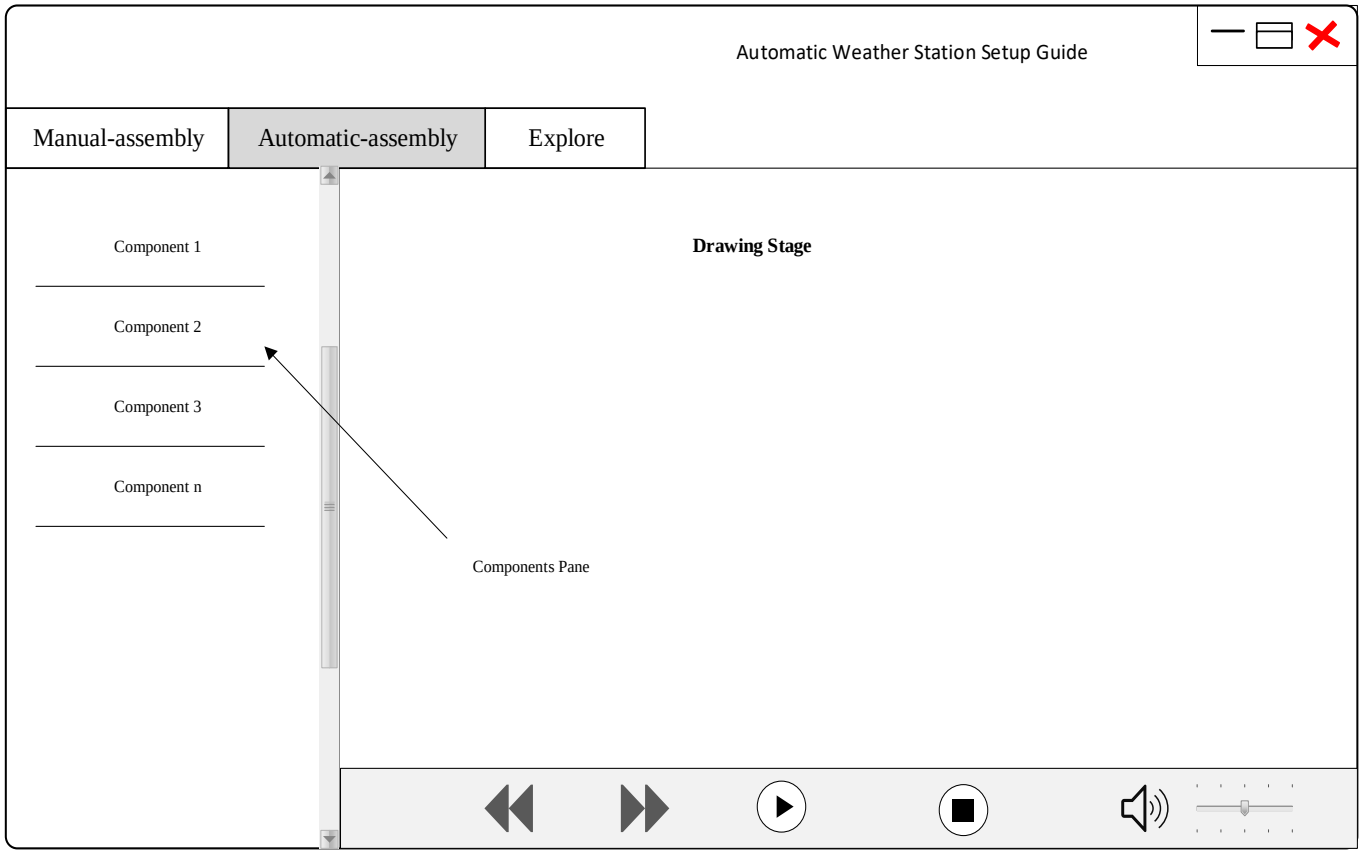

Figure 6. 3 Auto-assembly Interface

6.2.5 Manual assembly

This interface will enable the user to assemble the AWS node or LPG by themselves by dragging the components from the components pane and dropping them on the working area (canvas) and connecting them.

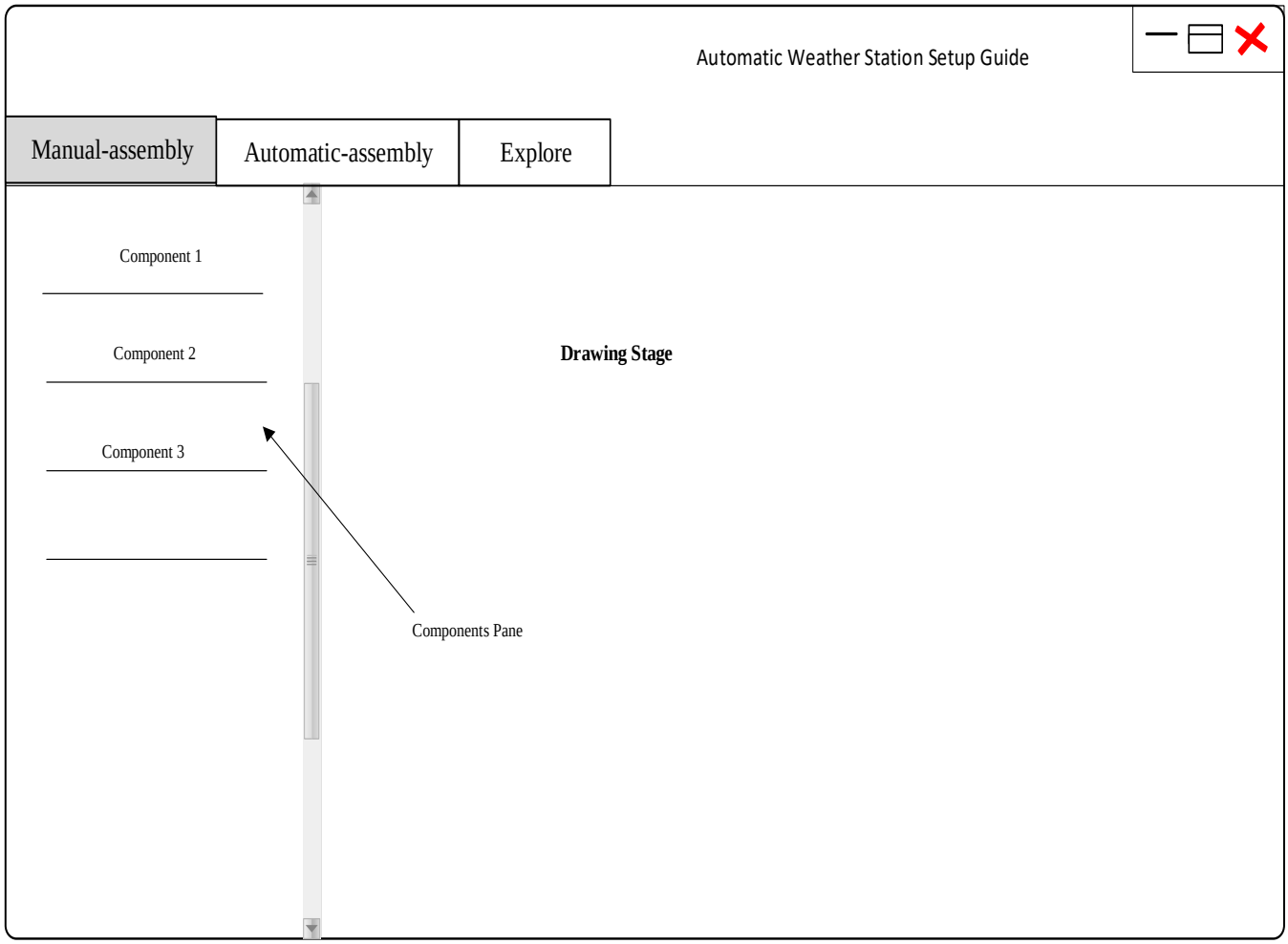

Figure 6. 4 Manual Assembly Interface

6.2.6 Explore (About WIMEA-ICT AWS)

This interface will provide the user with information about the WIMEA-ICT AWSs. On this interface, the user will click on a node and its information will be displayed in details i.e., the weather parameters the node captures, the components the node has.

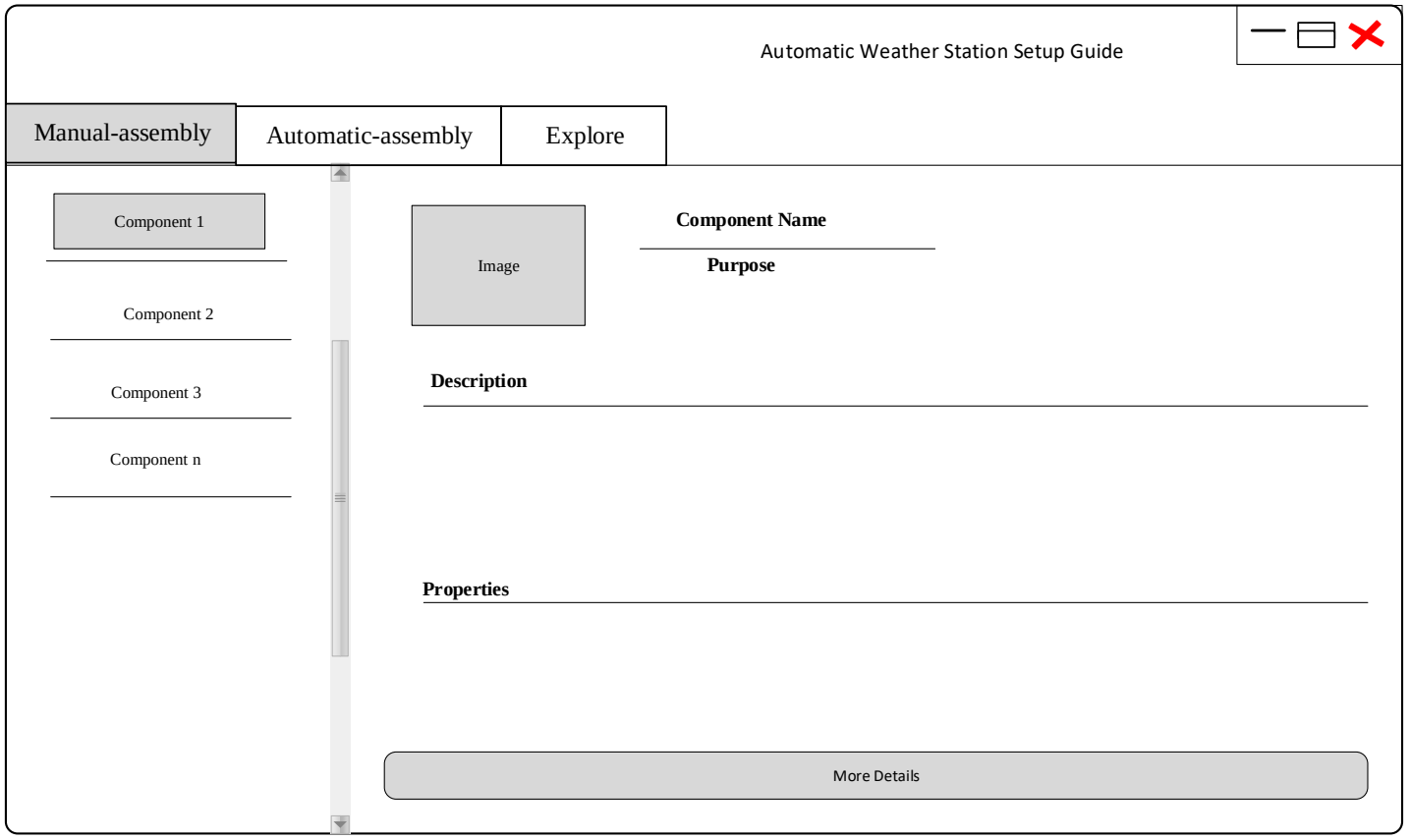

Figure 6. 5 Explore Interface

6.3 Screen Objects and Actions

This section describes the screen objects for the WIMEA-ICT AWS Setup Guide interface.

Table 2 Screen Objects and Actions

Screen Object	Action
Button	Click
Menu Item	Click
Select Boxes	Choose
Sliders	Mouse Down & Drag

Project Report

System implementation, testing and validation report

for

WIMEA-ICT AWS Setup Guide

Prepared by: BSE19-03

Date: 30th-Apr-2019

Version: 1.0

Table of Contents

List of Tables

Chapter 1: Introduction

1.1 Background and scope of the project

The WIMEA-ICT AWS Setup Guide is a web application intended to be used by the meteorological services in Uganda [3], Tanzania [4] and South Sudan [5] in simulating the setting up / assembling process of the various components of the WIMEA-ICT AWSs. The WIMEA-ICT AWS is made up of a gateway (device that sends the weather data to a remote repository) and four nodes (weather data capturing devices) i.e., two-meter node, ten-meter node, sink node and the ground node. The product therefore covers the simulation of setting up/ assembly of the WIMEA-ICT AWS. It also enables users to assemble the AWS nodes and the Low Power Gateway (LPG) themselves.

Users are presented with tools that make up the WIMEA-ICT AWS and are required to identify the parts. Later on, users' knowledge of assembling the AWS is tested using drag and drop features. Users are also provided with an automated way of assembling, which does not require them to interact but only watch the process. This is especially useful for first time users. Tutorials on technologies such as IEEE 802.15.4 [6] are loaded in the emulator in order to provide more information on how the technologies apply to the operations of the AWS. Text displayed on the user interface is in English language.

The application source code has been stored on the Github repository for WIMEA-ICT project [7] to enable users and other developers to access it and modify it where necessary.

1.2 Overview of the document

This document describes the implementation, testing and validation findings for the WIMEA-ICT AWS setup guide system. The provided tables are filled in with information about the tasks performed, methods used, criteria for acceptance, inputs and outputs for each task, required documentation, the persons responsible for validation among others. It is divided into the following sections:

Section 1: This section gives an overview of the system

Section 2: This section describes and specifies the system completely and is the basis for the validation process.

Section 3: This section describes the design output of the system.

Section 4: This section describes the inspection and testing the WIMEA AWS setup guide system.

Section 5: This section describes the installation and system acceptance test for the system.

Section 6: This section describes the performance, servicing, maintenance for the system.

Section 7: This section describes the conclusions and recommendations about the WIMEA-ICT AWS setup guide project report.

2. System Specifications

2.1 Version of requirements and Version Control

The requirements specified in the SRS for the WIMEA-ICT AWS Setup Guide system version 1.0 were sufficient and approved. These were followed to design and implement the system. For version control of the code, git has been used and commits are available on WIMEA-ICT github repositories [13].

2.2 Input

XML files – These are of type system default, they contain all the relevant information about any weather component (e.g. sensors, devices, etc.) to be assembled. This information includes the location of that component on the file system, its properties and other sub-components related to it.

Component Images – These are also of type system default, they are of **png** format and they are dragged to assemble the required component.

Audio – These are embedded in the system; they provide description of the steps taken to assemble a given AWS node.

2.3 Output

Textual descriptions – These provide information concerning the components of the AWS node.

Electronic representation of an AWS node that has been assembled either manually by the user or automatically by the system.

Motion graphics – animated outputs for the connection of the AWS node components combined with audio descriptions showing the steps taken to connect the components that make up an AWS node.

Points awarded to correct connections made during manual assembly process.

2.4 Functionality

The application provides the user with;

- Information about the technologies used by WIMEA-ICT to design and develop the automatic weather stations.
- A simulation for the assembly process of a given AWS node as developed by WIMEA-ICT project.

- Drag and drop feature which enable the user to manually setup the AWS node themselves.
- Simulation controls; these enable the user to pause/resume, increase/reduce the speed of the simulation, and increase/reduce the volume of the audio description.

2.5 Limitations and safety

- The system does not allow addition of new or more components which may arise due to change in technologies used by WIMEA-ICT to develop AWSs.
- The system is limited to English language only therefore, it cannot be used by individuals who don't know English.
- The user is required to be having internet connectivity on his/her computer in order to read about a given component's details from the web.

Safety

- The system holds no physical implications on its user

2.6 Special Requirements

Since the application is client sided, it can therefore be run from any directory on the user's computer. This however comes with an issue of accidental deletion by the user if it is run from frequently used directories like the desktop. For this reason, the application needs to be saved on not frequently accessed directories for example the C: drive in windows or ~ (home directory on Linux).

2.7 Errors and alarms

For wrongly connected components during manual assembly process, an error message is flagged to the user.

Chapter 3: Design output

3.1 Implementation (coding and compilation)

The system has been developed using tools that were evaluated to meet its requirements.

JavaScript 6.0 – this has been used to implement the back-end functionality of auto assembly module, manual assembly module and some parts of the explore module. Visual Studio 1.34.0 Code has been used as the environment to support the writing and interpretation of JavaScript code.

The choice of JavaScript is attributed to its high reliability and availability in every browser. This makes the system available every time it is needed. JavaScript is also robust in processing XML files which we have used for data storage.

HTM5 and CSS3 – these have been used to design the user interface of the system. HTML5 has support for canvas functionalities which support drag and drop functions and also drawing tools.

Adobe® Audition – this has been used to edit (trim, synchronize and join) **mp3** audio descriptions which are embedded in the Auto-assembly mode of the system.

Adobe® Photoshop – this has been used to format (resize and converting) the **png** images which are used in the system. The choice for this is attributed to its ability to adjust the image dimensions without affecting their pixel quality.

XML – this has been used as the primary data storage mechanism for the AWS node components. The flexibility with which properties and descriptions for each AWS component could be added required an extensible approach which is provided by XML nonetheless to say, data about a given AWS node does not change often. XML has also given us the ability to add our custom tags since different AWS node components have different properties.

A template of the XML file for an AWS node component is provided in the appendices section.

Git – this has been used for version control. It made it possible to track the changes made in the systems code during its implementation phase.

3.1.1 Modules and Integration

The explore module

This module reads the XML files for a given selected AWS node component and displays information (textual description and images) about it on the screen for the user to read. It also provides links as pointers to the detailed description of a given AWS node component on the web.

The Manual-assembly module

This has a drawing canvas where the AWS node components can be connected from by the user. It provides drag and drop features and images for a selected node which can be used during connections.

The Auto-assembly module

This also reads the components XML file and shows the steps for assembling a given AWS node using a list of its components. It also provides the user with audio description that is well synchronized with the motion pictures during the assembly process.

This module also provides controls for volume and speed of the simulation which can be used to increase/reduce volume and to increase/reduce frame rate of the simulation respectively.

Table 3. 1 Design Details

Topics	**Design output**	
Good programming practice	Source code is... ☑ Modulized ☑ Encapsulated ☑ Functionally divided ☑ Strictly compiled ☐ Fail-safe (handling errors)	Source code contains... ☐ Revision notes ☑ Comments ☑ Meaningfull names ☑ Readable source code ☑ Printable source code

Topics	**Design output**
Dynamic testing	☑ All statements have been executed at least once ☑ All functions have been executed at least once ☑ All case segments have been executed at least once ☑ All loops have been executed to their boundaries ☐ Some parts were not subject to dynamic test Comments:

3.2 Documentation

Software Design Document: The design phase produced the software Design document for WIMEA-ICT AWS [14]. It provides the architecture and the design of the system to the programmers who have implemented the system.

User Manual: The implementation phase produced the System User Manual which shall be used by the users to learn how to use the system. See Appendix B

Chapter 4: Inspection and testing

4.1 Introduction

Table 4. 1 Inspection plan and performance

Topics	**3.3.1 Inspection plan and performance**	*Date*
Design output	☑ Program coding structure and source code ☑ Evidence of good programming practice ☑ Design verification and documented reviews ☐ Change-control reviews and reports **Comments**: The code is well commented and organized basing on functionality.	20th Apr, 19 26th Apr, 19 16th May, 19
Documentation	☑ System documentation, flow charts, etc. ☑ Test results ☑ User manuals, On-line help, Notes, etc. ☑ Contents of user manuals approved Comments: The documents were inspected and reviewed by the Supervisor.	26th Apr, 19 16th May, 19 29th May, 19
Software development environment *Environment elements inspected...*	☐ Data integrity ☑ File storage ☐ Access rights	
Result of inspection *Approval of inspection.*	☑ Inspection approved Comments: Project supervisor approved the	30th May, 19

4.2 Test plan and performance

4.2.1 Test objectives

The major objective of the test is to verify that the functionality of the WIMEA-ICT AWS Setup Guide application version 1.0 works according to the specifications provided in its System Requirement Specification document [2].

The tests were executed to verify that the test cases, identify, fix and retest the defects in the severity classes. However, the highest priority was given to the severity class which degrade the basic system capability, like dragging and dropping of weather components in the Manual simulation mode. Other objectives included;

- To define tools to be used through the testing process.
- To define how the tests were to be executed.
- To communicate to the responsible parties, the items to be tested, the schedule, the test budget and defined environment needs for testing.

The test has yielded a ready-to-deploy software and a set of possible test cases that can be reused in future testing plans.

4.2.2 Scope and Relevancy of tests

The tests were designed to cater for verification and validation testing techniques. Verification testing covered the system's requirements during the requirements phase while Validation testing was employed during the implementation phase.

Testing has been essential because of the following reasons;

- It has guided us in ensuring that the right quality is engineered in the system during its design and implementation.
- It has enabled ensure that the satisfaction of the end user in the system in achieved.
- It has helped us achieve effective performance of the system.
- It has also enabled us to identify and fix the errors which were made during system implementation.

4.2.3 Levels of tests

Module Tests: These were carried out on module basis. Each module was tested independent of each other to ensure correct functionality.

Integration Tests: The three modules of the system were integrated and the entire system tested to ensure no breakage of the system functionality after integration.

Smoke tests: These were carried out whenever a piece of code was added to the system during its implementation to ensure that the added in code does what it was intended to do.

System Acceptance Tests: These tests were performed to ensure that the application is in compliance with the system requirement specifications. It was carried out by the BSE19-03 group members under the guidance of the supervisor after running the system on several computers and different browsers of students on the WIMEA-ICT project.

4.2.4 Types of tests

a) Verification Testing

This was done primarily at an earlier stage of system development to uncover defects at an earlier stage and directly from their source. This was done during the requirements phase and the programming phase.

Purpose: To uncover defects from their origin (requirements). This test focused on validating that the specified requirements were attainable and also make sure that critical defects are removed before later testing phases commenced.

Scope: System requirements for all the three subsystems and the design specifications.

Testers: Testing Team – BSE19-03

Methods: Verification testing was carried out through requirements review, code comparisons and walk throughs.

Timing: Verification testing was carried out at the end of the requirements collection phase.

b) Functional Testing

This has been carried out to check the functions of the system by feeding the inputs into the system and validating the output it provides.

Scope: functions of the system.

Methods: functional scripts.

Timing: Functional testing commenced when the first working prototype of the application was produced and after verification testing was completed.

4.2.5 Sequence of tests

Test cases have been written based on the different modules of the system and have been distributed to some interns of WIMEA-ICT project for validation. Each test case has been given a unique identifier of the format ***TC_M_n*** (M for the module it belongs to either, EX, MA, and AA for Explore module, Manual Assembly module and Auto-Assembly module respectively) and n is a number. See appendix A for a list of test cases and their fail/pass criteria.

4.2.6 Configuration and calculation tests

The system has been tested on later versions of; Firefox 40.x.x, Internet explorer 5, Microsoft edge, Google Chrome 48.x.x.x browsers. All the features of the system have been verified to function appropriately without breakage. However, Opera browser requires to first configure to accept http requests made using file:/// in order to load XML files i.e. enabling Cross-Origin Resource Sharing [15].

4.2.7 Calculation Tests:

For the case of awarding points incrementally for each correct connection, test cases have been written to ensure that the points are awarded correctly.

4.3 Precautions

4.3.1 Anomalous conditions

i. The system might fail to load AWS Component images if it is run on a browser which has lower versions of JavaScript earlier than version 6.

ii. The system might fail to load the AWS Component if it is run from a browser that does not allow HTTP requests made from file:///. For browsers which have Cross-Origin Resource Sharing (CORS) restricted, the system might fail to respond.

4.3.2 Precautionary steps taken

i. In case of CORS restriction, the user can enable CORS feature in the browser settings. The problem of CORS restriction can also be solved by running the system from a server.

ii. The user can update the browser so as to enable JavaScript 6 engine.

Chapter 5: Installation and system acceptance test

5.1 Input files

Zipped folder (***aws_setup_guide.zip***)

This contains the ***aws_setup_guide*** folder which has the system files

It may or may not contain an installer for Firefox browser since for it by default allows Cross-Origin Resource Sharing so as to allow the application run from any directory on the user's computer without need for a server.

5.2 Supplementary files

The system user manual (***user_manual.pdf***) has been provided to guide the users on how to use the system.

A readme file; ***read_me.txt***, has also been included in the system folder (*aws_setup_guide*) to guide the users on how to install and run the system

5.3 Installation qualification

Simply the user has to;

I. *Copy the zipped folder of the system;* ***aws_setup_guide.zip*** *to and paste it to any preferred directory*

II. *Extract the aws_setup_guide.zip folder. This creates* ***aws_setup_guide*** *folder and a Firefox browser installer file.*

III. *You may or may not install the Firefox browser depending on your choice.*

IV. *Open the* ***aws_setup_guide.hml*** *file in the browser (ensure the chosen browser is set to enable CORS, otherwise open using Firefox browser).*

That is all you have to do to have the system running.

Table 5. 1 Checklist of the Installation and system acceptance test

Topics	**Installation summary**
Installation method *Manual Installation*	☐ Automatic - installation kit located on the installation media ☑ Manual - Copy & Paste from the installation media **Comments**: The system is distributed through a zipped folder which can be copied and pasted in any user's directory on the computer.
Installation media *Media containing the installation files...*	☑ Diskette(s) ☑ CD-ROM ☑ Source disk folder (PC or network) ☐ Download from the Internet **Comments**: The system can be transferred using any removable media for transferring files. It can also be downloaded from the WIMEA-ICT github repository [13].

Table 5. 2 Installation Procedure Check

Topics	**Installation procedure**	*Date*
Authorization *Approval of installation in actual environment.*	**Person responsible:** Project Supervisor and BSE19-03 group.	29th May, 19

Topics	**Installation procedure**	*Date*
Installation test *The following installations have been performed and approved...*	☑ Tested and approved in a test environment ☑ Tested and approved in actual environment ☑ Completely tested according to test plan ☑ Partly tested (known extent of update) **Comments**: The system has however not been tested on Safari browser to check out how it is rendered.	29th May, 19

Chapter 6: Performance, servicing, maintenance, and phase out

6.1 Service and maintenance

Addition of new AWS component images and XML files in case the technologies used by WIMEA-ICT project in setting up any given AWS change. In case new AWS components have to be added to the system, their images have to be edited and XML files written following the component XML file template (***component.xml***) and added to the system folders so as to be used in the system.

6.2 Performance and Maintenance

Table 6. 1 Performance and maintenance details

Topics	**Performance and maintenance**		*Date*
Problem / solution	**Problem**	**Solution**	
	Accidental Deletion of system files	One approach can be to save the system files in Program files or C: drive on Windows or ~ in Linux and then paste a shortcut to the aws_setup_guide.html file on the user's desktop.	Possible even currently
	Accidental Deletion of system files and Browser incompatibility.	Hosting the system on WIMEA-ICT server so that the system can be accessed online.	Future
	New AWS components.	Addition of configuration options to allow for addition of new AWS components in case new ones arise in WIMEA-ICT AWSs.	Future

Topics	**Performance and maintenance**	*Date*
Functional expansion and performance improvement	*Suggestions and requests, which can improve the performance of the system. e.g.* • Inclusion of a tour feature such that the user is asked if he/she may want to tour around the system to get acquainted with how to use the system. • Addition of the help section so as to load the user manual pdf in a new browser tab. • Compiling html files for the AWS components such that their details are locally available on the system so as to avoid internet requirements during access to the components' details on the web. • Hosting the system on WIMEA-ICT server so as to have an online version of the system. • Having better connection options for components on the stage as new technologies may arise.	Future Future Future Future

Chapter 7: Conclusion and Recommendations

The main objective of the project has been to develop a graphical emulation tool for setting up AWS nodes developed under the WIMEA-ICT project. Through this project we have been able to work as a team and improve our software development skills.

We therefore recommend that WIMEA-ICT adds an extra effort on improving this system so as to drastically reduce on the knowledge gap which exists between them (the developers of the AWSs) and the actual people they develop them for (the AWS deployment team in meteorological agencies in Uganda, South Sudan and Tanzania). Nevertheless, the application is also one of a kind.

Appendix A: Test Cases

Table 7. 1 Test Cases

Module	Test Case ID	Testcase / Test Procedure	Expected	Passed?
Manual	TC_MA_01	User drags and drops AWS components in correct location	correct evaluation, correct placement on the stage	Yes
	TC_MA_02	User drags and drops AWS component in wrong location on stage	incorrect placement of component on the stage	Yes
	TC_MA_03	User makes connection between connection points on the component correctly	correct connection verified	Yes
	TC_MA_04	User makes connection between connection points on the component wrongly	incorrect connection identified	Yes
	TC_MA_05	Drag working	Components can be dragged on the stage	Yes
	TC_MA_06	Drop component working	Components can be dropped at their current drag position.	Yes
	TC_MA_07	Drag speed on the screen.	correct drag speed relative to position of the cursor on the screen.	Yes
	TC_MA_08	AWS component selected	AWS component is highlighted	Yes
	TC_MA_09	Deletion of AWS component from stage	Selected component for delete is removed from the stage	Yes
	TC_MA_10	Checkout button pressed	points awarded for correct point, wrong connections highlighted	Yes
Automatic	TC_AA_01	Volume slider adjusted to the left	volume reduced relative to new slider position	Yes
	TC_AA_02	Volume slider adjusted to the right	volume increased relative to new slider position	Yes
	TC_AA_03	stop button pressed	Automatic simulation is terminated	Yes
	TC_AA_04	pause button pressed	Automatic simulation is halted	Yes
	TC_AA_05	play button pressed	Automatic simulation is started, Automatic simulation is resumed.	Yes
	TC_AA_06	Rewind button pressed	Frame rate reduced (speed of simulation reduced)	Yes

	TC_AA_07	Forward button pressed	Frame rate increased (speed of simulation increased)	Yes
	TC_AA_08	Audio Output	Description sound is in sync with the motion images	
Explore	TC_EX_01	AWS component selected for explore	AWS component name, properties, description and image are shown on the screen.	Yes

Appendix B: User Manual

USER MANUAL

FOR

WIMEA-ICT AWS SETUP GUIDE

1 Introduction

This is a user manual for the WIMEA-ICT AWS Set up guide system used to learn about the different components of the automatic weather stations and how they can be connected.

2 General Information

The AWS setup guide is a simulator designed to enable the different teams of WIMEA-ICT and meteorological organizations that carry out deployment of the automatic weather stations learn more about the connections of the different components. The main service provided by the simulator is automatic assembly of the different nodes of the automatic weather stations.

2.1 System overview

The WIMEA-ICT AWS Set up guide provides the following functionalities:

- Automatic simulation of an AWS node.
- Simulating environment to assist in emulating the WIMEA-ICT AWS.
- Enabling users to choose the AWS node or LPG to be assembled.
- Loading of the different components of the selected node.
- Drag and drop of components to assemble a node.
- Award points to the user after manual assembly of the nodes.
- Indicate the wrong connections made in case of any.
- Controls for automatic simulation which may include play, pause, resume.
- Audio to provide more information about the connections and how they are made.
- Viewing of different properties of the different components.

2.2 Contact

Request for access and inquiries on the use of the system, the design and functionalities of the system should be sent to the dedicated email mnsabagwa@cit.mak.ac.ug.

3 Getting Started

3.1 How to access the system

The user can obtain a fresh copy of the system by downloading it from the WIMEA-ICT github repository[13].

Simply the user has to;

V. *Copy the zipped folder of the system;* ***aws_setup_guide.zip*** *to and paste it to any preferred directory*

VI. *Extract the aws_setup_guide.zip folder. This creates* ***aws_setup_guide*** *folder and a Firefox browser installer file.*

VII. *You may or may not install the Firefox browser depending on your choice.*

VIII. *Open the* ***aws_setup_guide.hml*** *file in the browser (ensure the chosen browser is set to enable CORS, otherwise open using Firefox browser).*

That is all you have to do to have the system running.

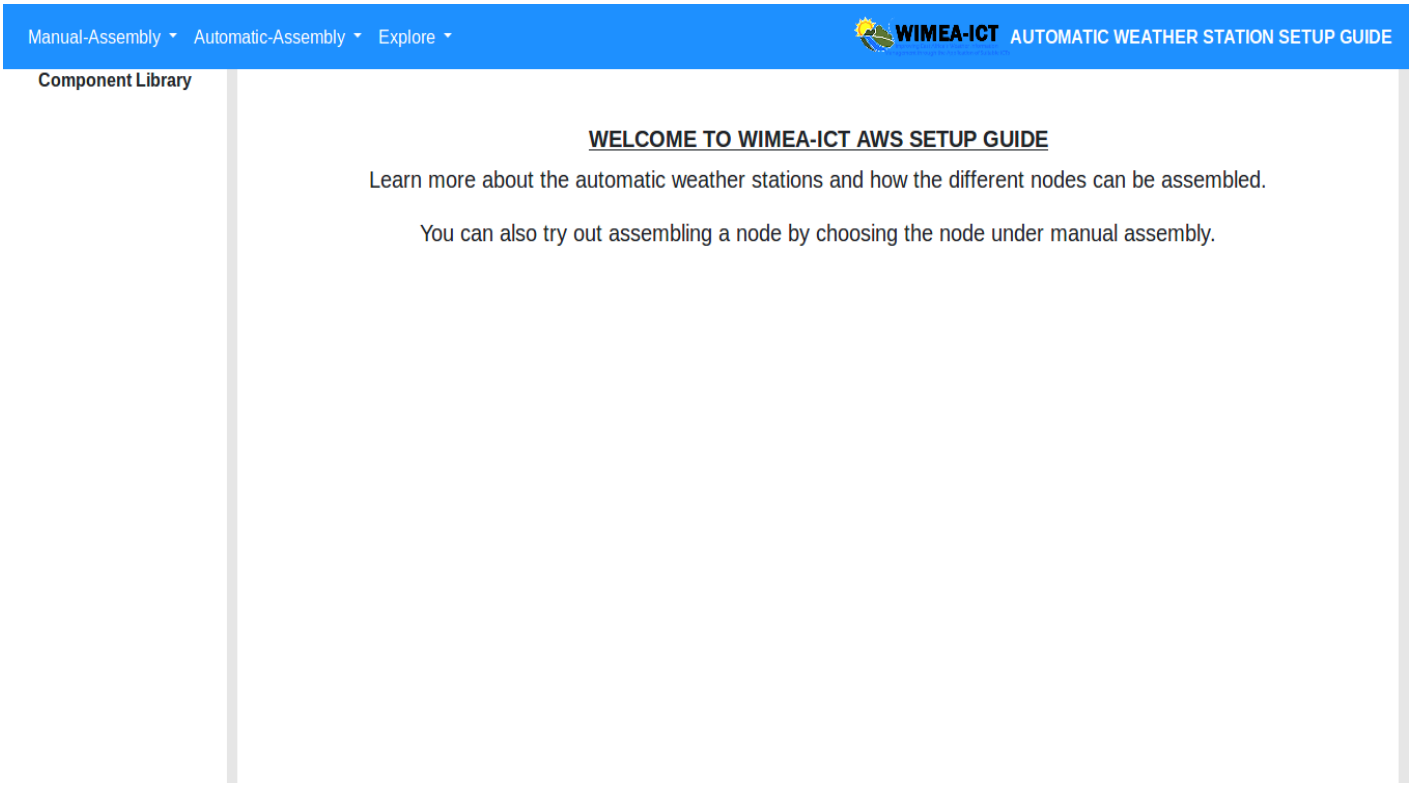

Figure 3. 1 welcome page of the simulator.

3.2 Choose a node to be assembled.

Once the system loads, you can now choose the node you want to assemble either automatically or manually, or even the AWS node you would like to explore.

To choose a node, click on the mode, for example, manual assembly and select a node from the drop-down menu.

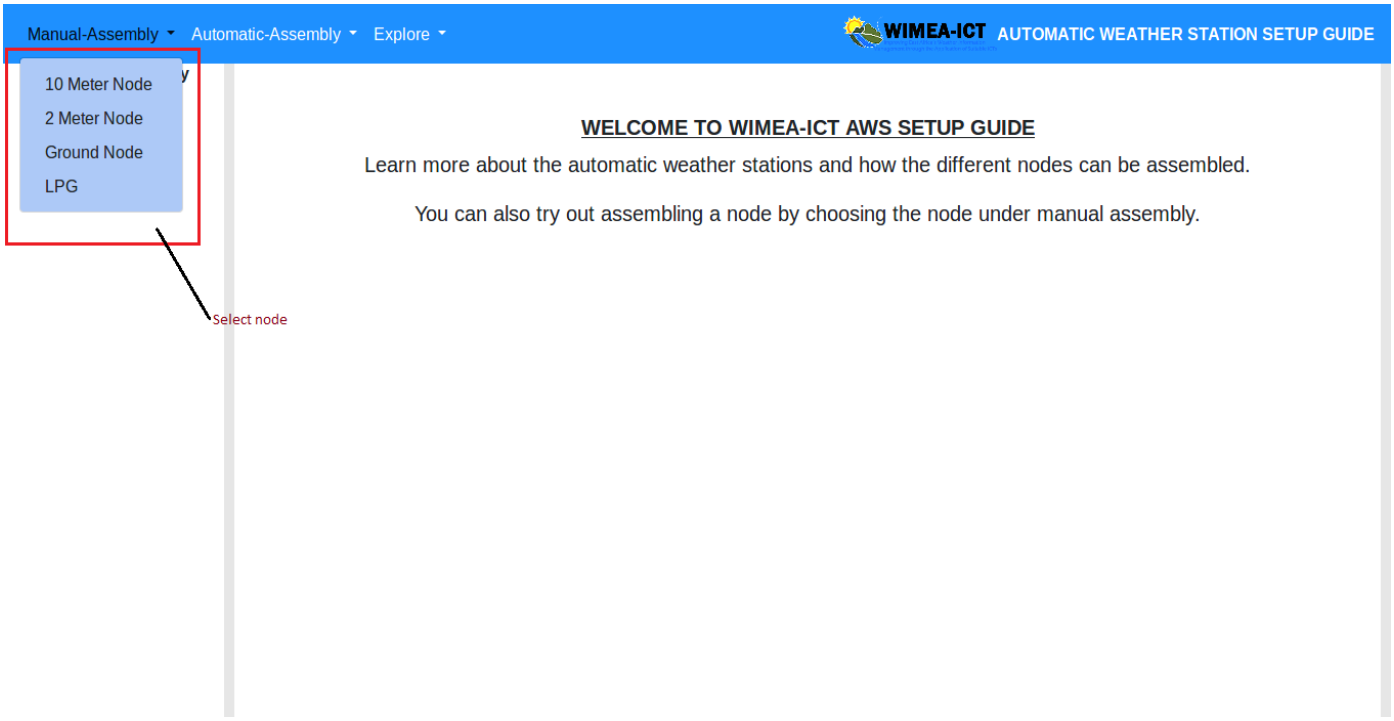

Figure 3. 2 choose a node to be assembled or explored from the drop-down menu.

3.3 View Loaded components for a node

Once the node to be assembled or explored has been selected, you are able to view the components required for the node on the left (Components pane) of the simulator.

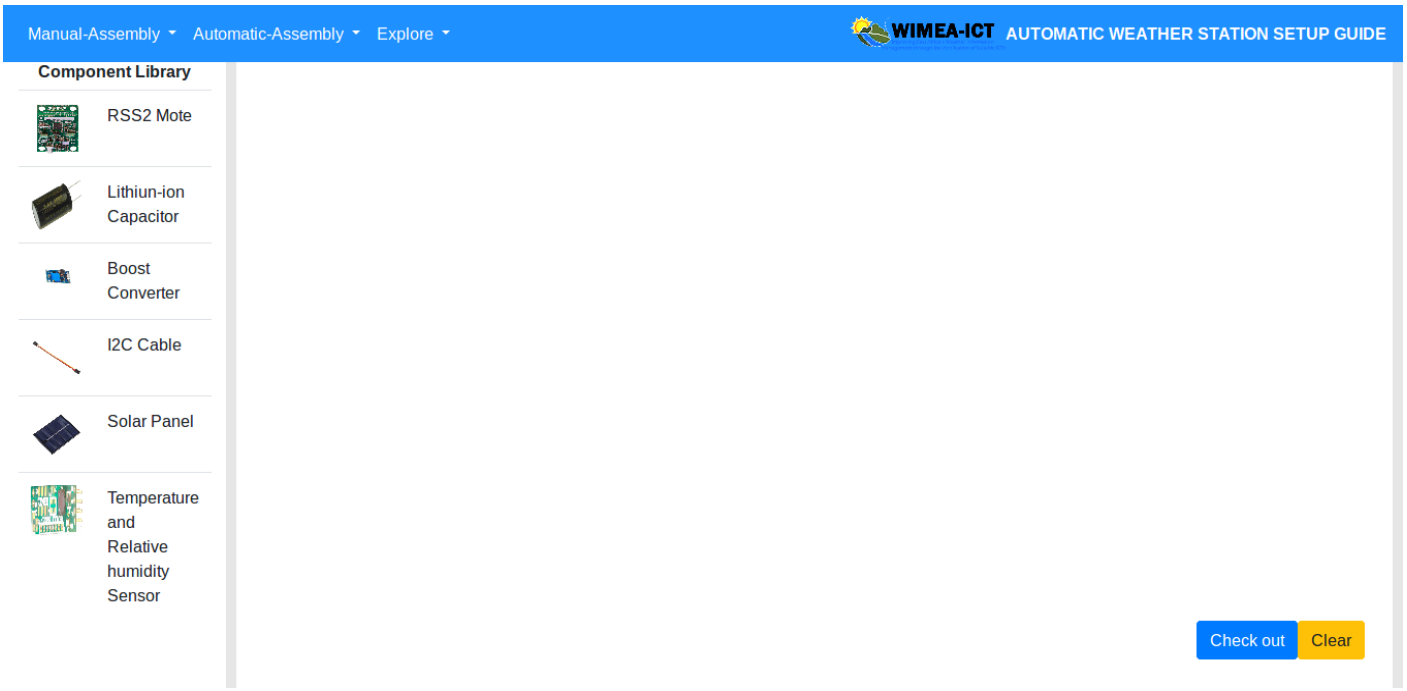

Figure 3. 3 Loaded components for the 2-meter node.

3.4 Drag and drop of the components.

While pressing down your mouse, move the selected component on the stage to highlighted location on the layout of the node being assembled.

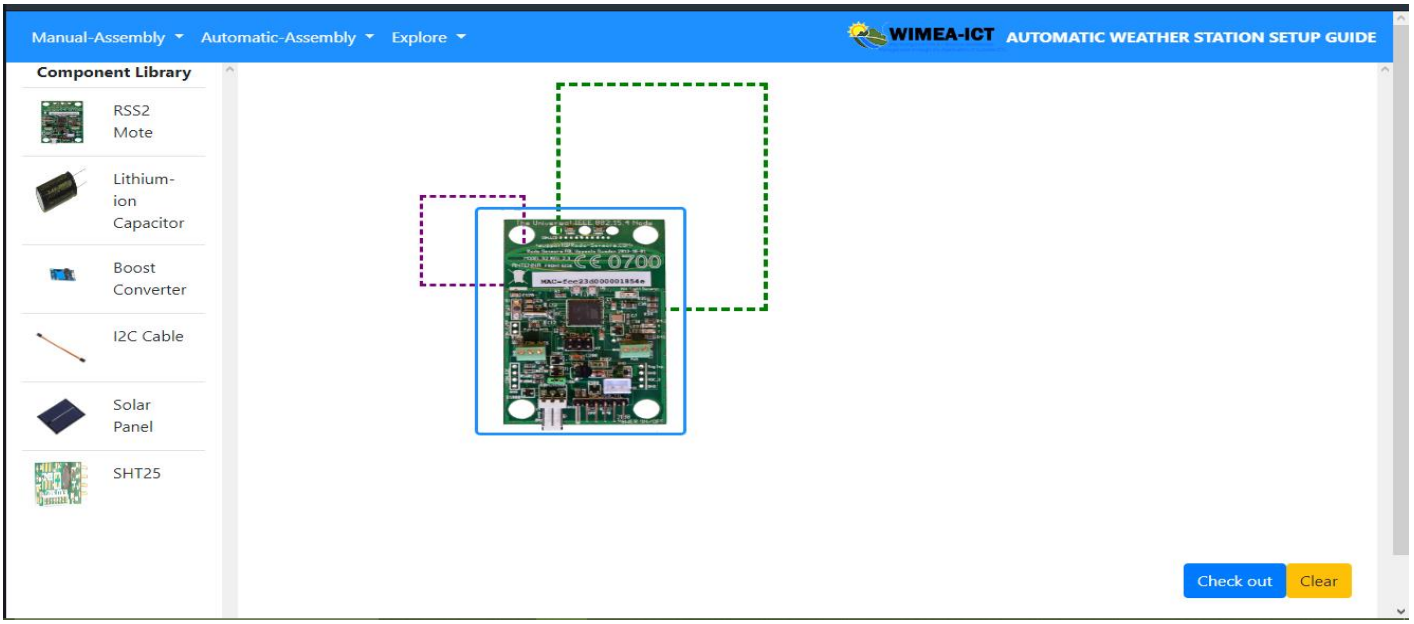

Figure 3. 4 The component being placed (dragged) on the stage

To assemble the chosen AWS node, drag and drop AWS components to the stage and connect them using their connection points.

Figure 3. 5 One of components dragged on the stage.

3.5 Award points for each connection made.

Once you have finished placing or assembling the different components for a node, click the check-out button at the bottom of the simulating environment.

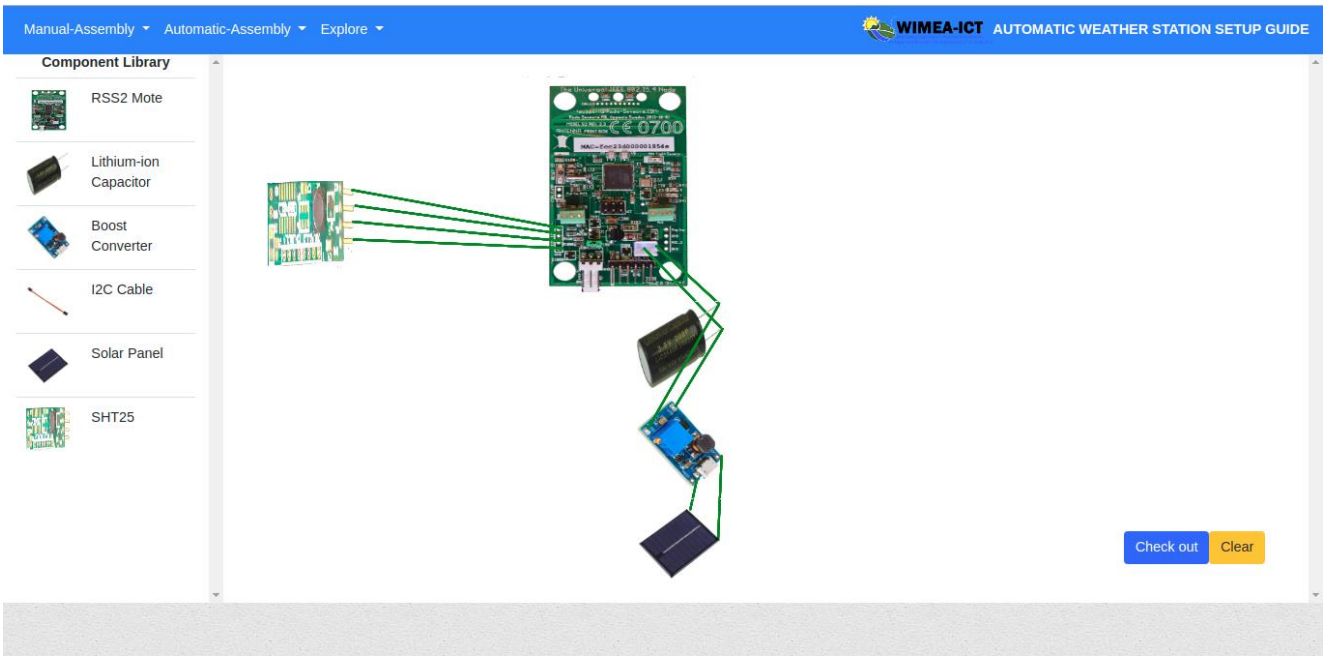

Figure 3. 6 All components assembled in the layout.

Once the button has been clicked, you are able to see your scores with the number for right connects and number of wrong connections. If all the connections are right, all the borders of the layouts for the components with be highlighted green to show wright connections.

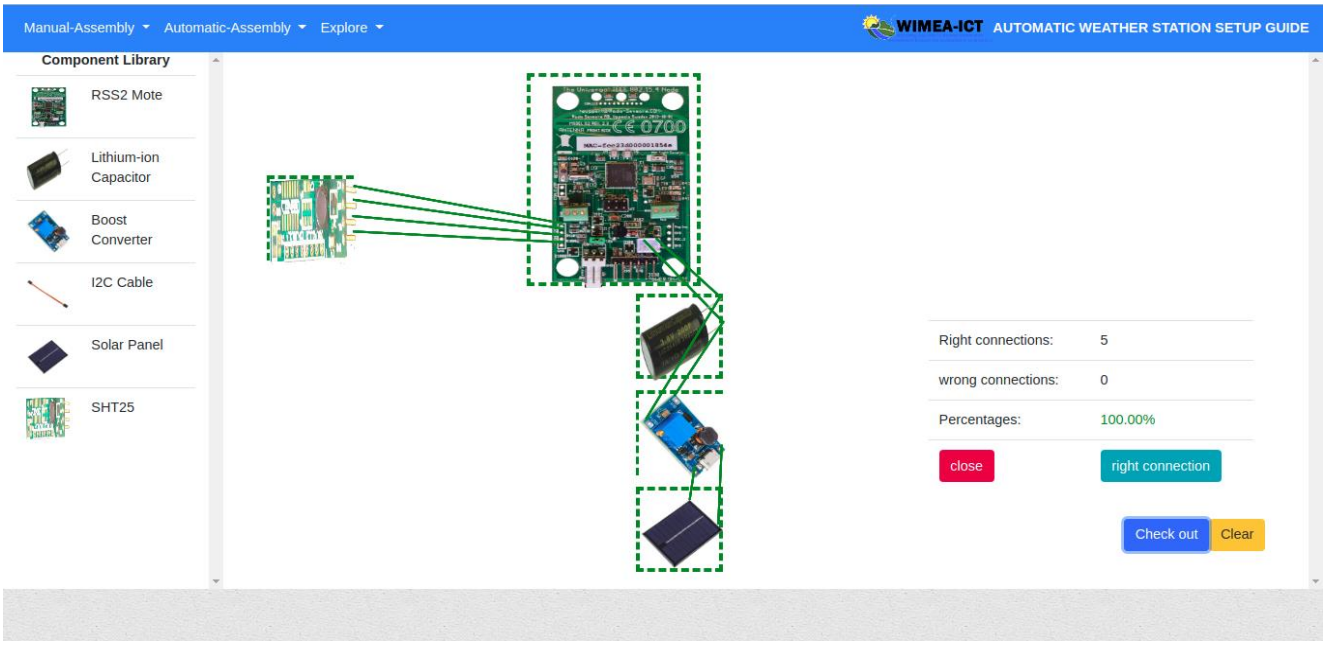

Figure 3. 7 layout borders highlighted green for the right placement.

In case of any wrong connection, the border of the layouts with wrong connections, will be highlighted red.

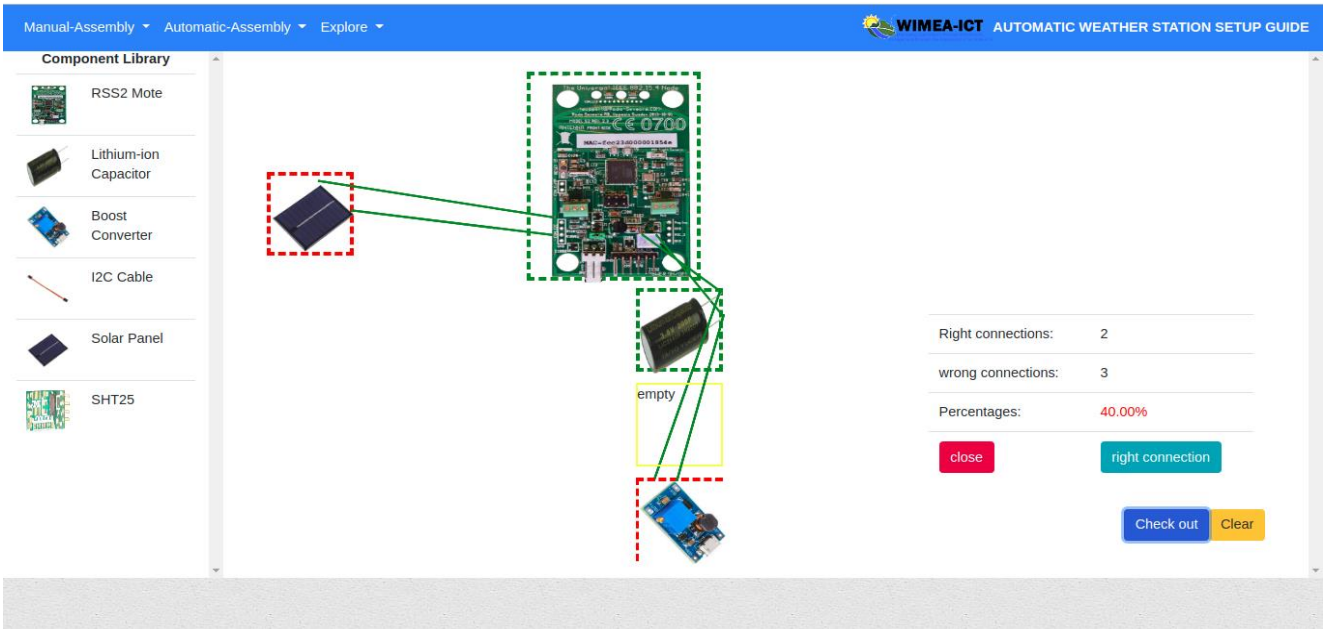

Figure 3. 8 Wrong connections highlighted red.

3.6 Clear and Delete.

To clear the stage, just click the clear button

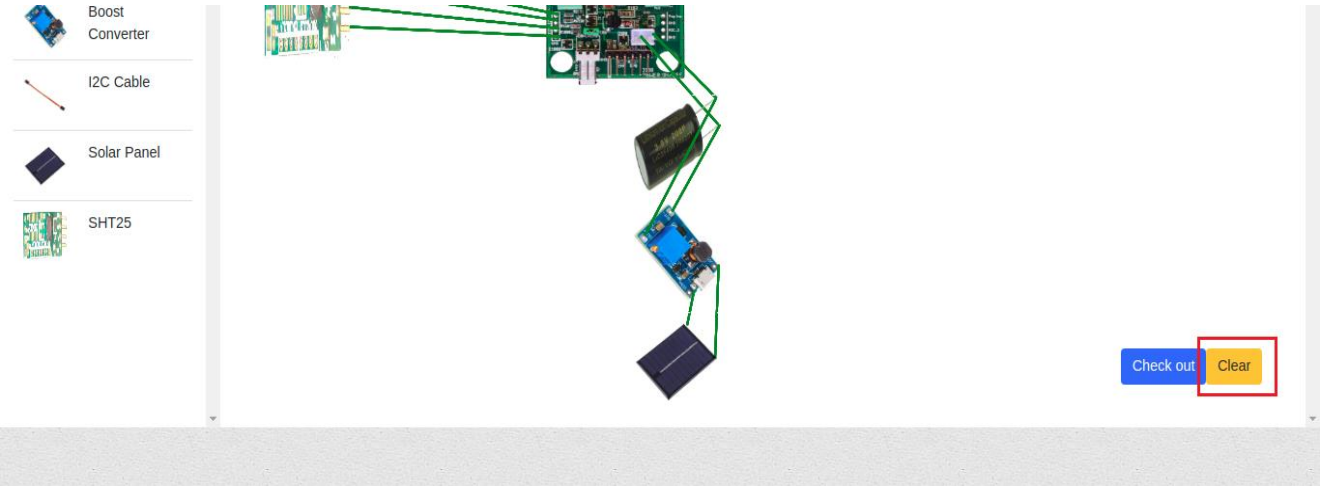

Figure 3. 9 Clear Button.

To delete a component from stage, double click on it, a small delete window is displayed on the screen, click delete and the AWS component shall be deleted for the screen.

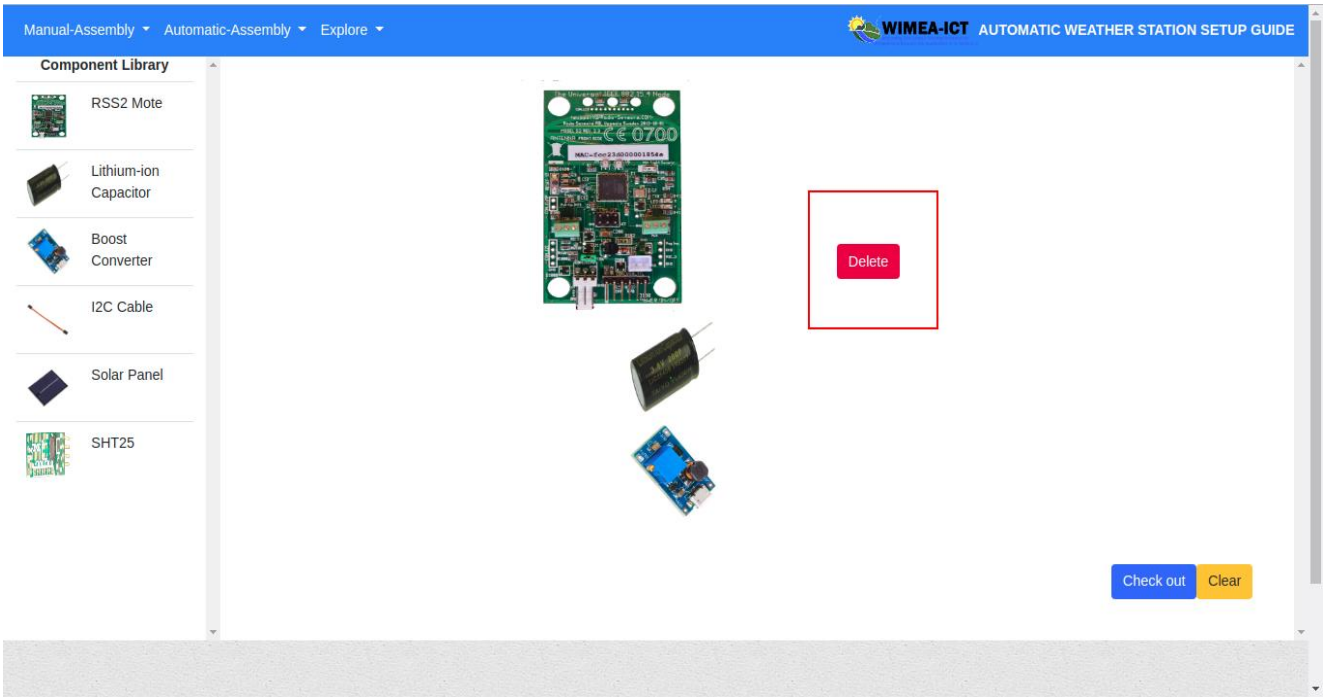

Figure 3. 10 Delete Component.

3.7 View component information in Explore.

Once the node to be explored has been selected under the explore drop-down menu, you can select a component and information will be displayed for that particular component. For example, in figure 3.11 below the boost converter was selected

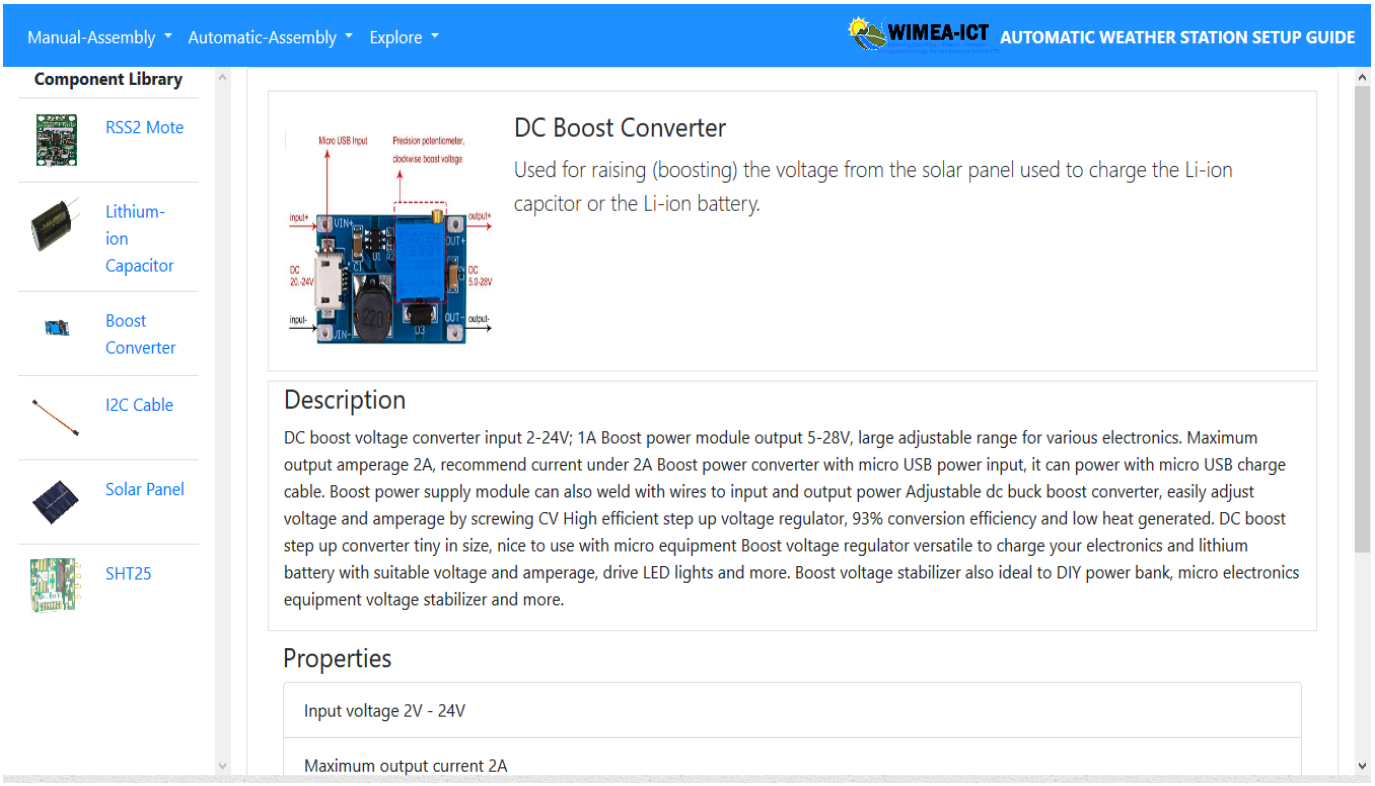

Figure 3. 11 Boost Convertor Explore mode

3.8 Control the automatic simulation.

To start the automatic simulation, you are required to click the play button and the simulation starts. You can be able to pause, resume, stop, forward or rewind the simulation.

Figure 3. 12 Play simulation

Appendix C: Allocation of Tasks

Allocation of the system development tasks amongst the team members.

Table 7. 2 Allocation of Tasks to Team

Member	Role / Tasks
Ninsiima Grace	• Manual Assembly Module • System Documentation
Mawanda Henry	• Manual Assembly Module • Automatic Assembly Module
Mwesigye Robert	• Explore Module • System Documentation
Ssemagoye Umar Munddu	• Data Design (Writing XML files and Images) • Automatic Assembly Module

Final approval for use	
Identification:	
Responsible for validation:	
Remarks:	
Date:	Signature: