**System implementation, testing and validation report**

**for**

**WIMEA-ICT AWS Setup Guide**

|  |  |
| --- | --- |
| Prepared by: | BSE19-03 |
| Date: | 30th-Apr-2019 |
| Version: | 1.0 |

# List of Tables

[*Table 3. 1 Design Details 8*](#_Toc10198093)

[*Table 4. 1 Inspection plan and performance 10*](#_Toc10198101)

[*Table 5. 1 Checklist of the Installation and system acceptance test 15*](#_Toc10198114)

[*Table 5. 2 Installation Procedure Check 16*](#_Toc10198115)

[*Table 6. 1 Performance and maintenance details 17*](#_Toc10198123)

[*Table 7. 1 Test Cases 20*](#_Toc10198130)

[*Table 7. 2 Allocation of Tasks to Team 21*](#_Toc10198131)

# Chapter 1: Introduction

## 1.1 Background and scope of the project

The WIMEA-ICT AWS Setup Guide is a desktop application intended to be used by the meteorological services in Uganda [1], Tanzania [2] and South Sudan [3] in simulating the setting up / assembling process of the various components of the WIMEA-ICT AWSs. The WIMEA-ICT AWS is made up of 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 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 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 be provided with an automated way of assembling, which does not require them to interact but only watch the process. This is especially be useful for first time users. Tutorials on technologies such as IEEE 802.15.4 [4] 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 [5] 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 [5].

## 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 can only be used by one user at a time.
* 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 needs.

**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 robustness 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. Design Details

| *Topics* | **Design output** | |
| --- | --- | --- |
| **Good programming practice** | Source code is... | Source code contains... |
| **Dynamic testing** | Comments: | |

## 3.2 Documentation

**Software Design Document:** The design phase produced the software Design document for WIMEA-ICT AWS [6]. 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. Inspection plan and performance

| *Topics* | **3.3.1 Inspection plan and performance** | *Date* |
| --- | --- | --- |
| **Design output** | **Comments**: The code is well commented and organized basing on functionality. | 20th Apr, 19  26th Apr, 19  16th May, 19 |
| **Documentation** | 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...* |  |  |
| **Result of inspection**  *Approval of inspection.* | 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 desktop application version 1.0 works according to the specifications provided in its System Requirement Specification document [7].

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 distribute 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 [8].

### 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

1. 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.
2. 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

1. 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.
2. 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;

1. *Copy the zipped folder of the system;* ***aws\_setup\_guide.zip*** *to and paste it to any preferred directory*
2. *Extract the aws\_setup\_guide.zip folder. This creates* ***aws\_setup\_guide*** *folder and a Firefox browser installer file.*
3. *You may or may not install the Firefox browser depending on your choice.*
4. *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. Checklist of the Installation and system acceptance test

| *Topics* | **Installation summary** |
| --- | --- |
| **Installation method**  *Manual Installation* | **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...* | **Comments**: The system can be transferred using any removable media for transferring files. It can also be downloaded from the WIMEA-ICT github repository[5]. |

Table 5. 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 |
| **Installation test**  *The following installations have been performed and approved...* | **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. 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 |
| **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. 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 | Descrption 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 metrological 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 provides 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 if the system, the design and functionalities of the system should be sent to the dedicated email [mnsabagwa@cit.mak.ac.ug](mailto:mnsabagwa@cit.mak.ac.ug).

# 3 Getting Started

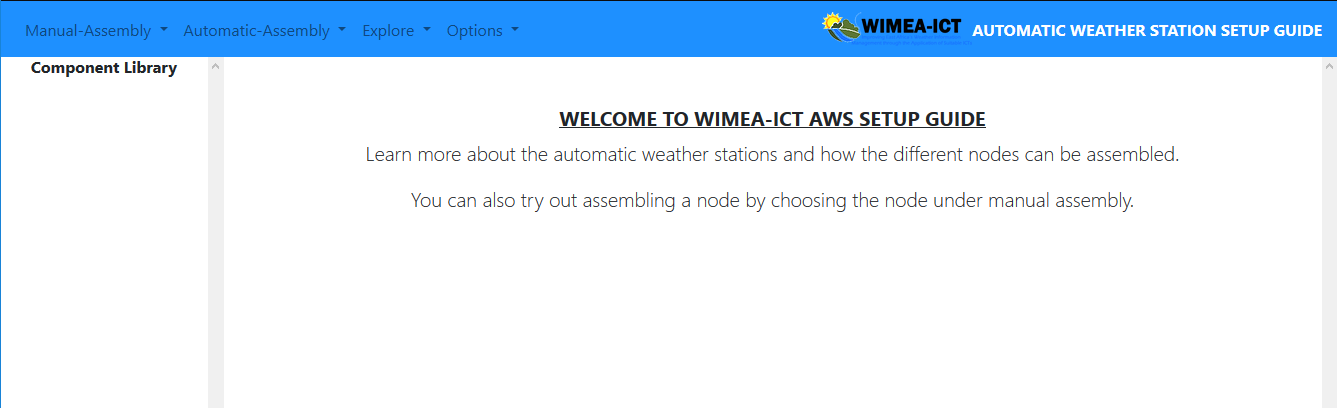
## 3.1 How to access the system

The user can obtain a fresh copy of the system by downloading it from the WIMEA-ICT github repository[1].

Simply the user has to;

1. *Copy the zipped folder of the system;* ***aws\_setup\_guide.zip*** *to and paste it to any preferred directory*
2. *Extract the aws\_setup\_guide.zip folder. This creates* ***aws\_setup\_guide*** *folder and a Firefox browser installer file.*
3. *You may or may not install the Firefox browser depending on your choice.*
4. *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.*

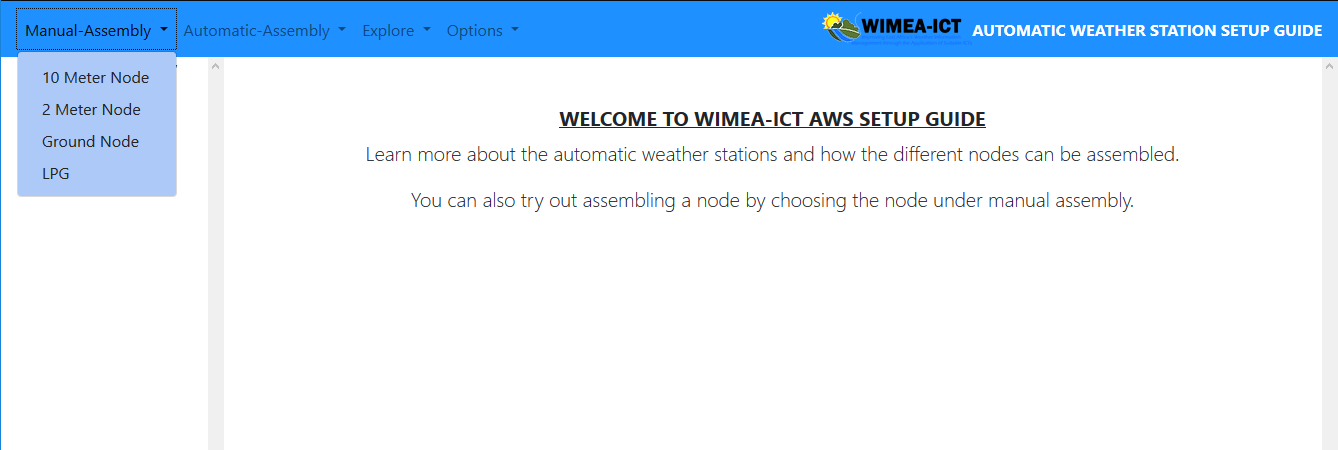


*Figure 3. 1 welcome page of the simulator.*

## 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 side menu of the simulator. You are also provided with a layout which shows the order in which the components can be assembled.

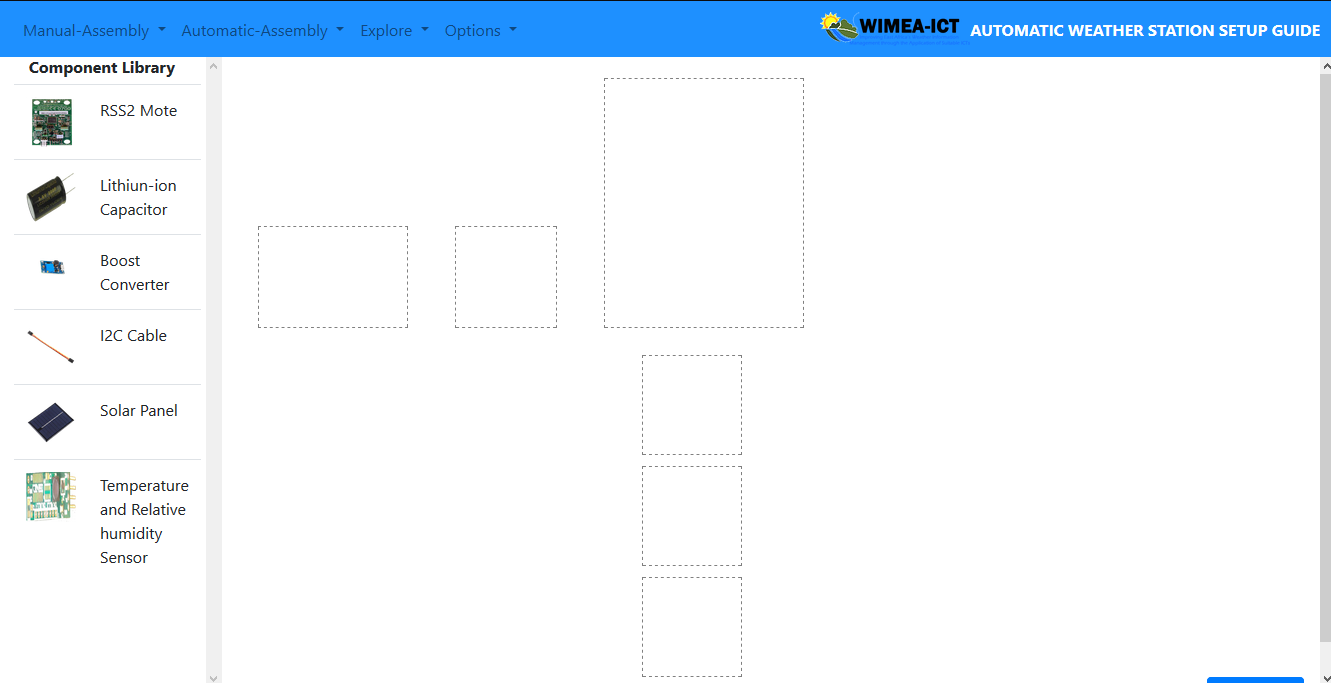
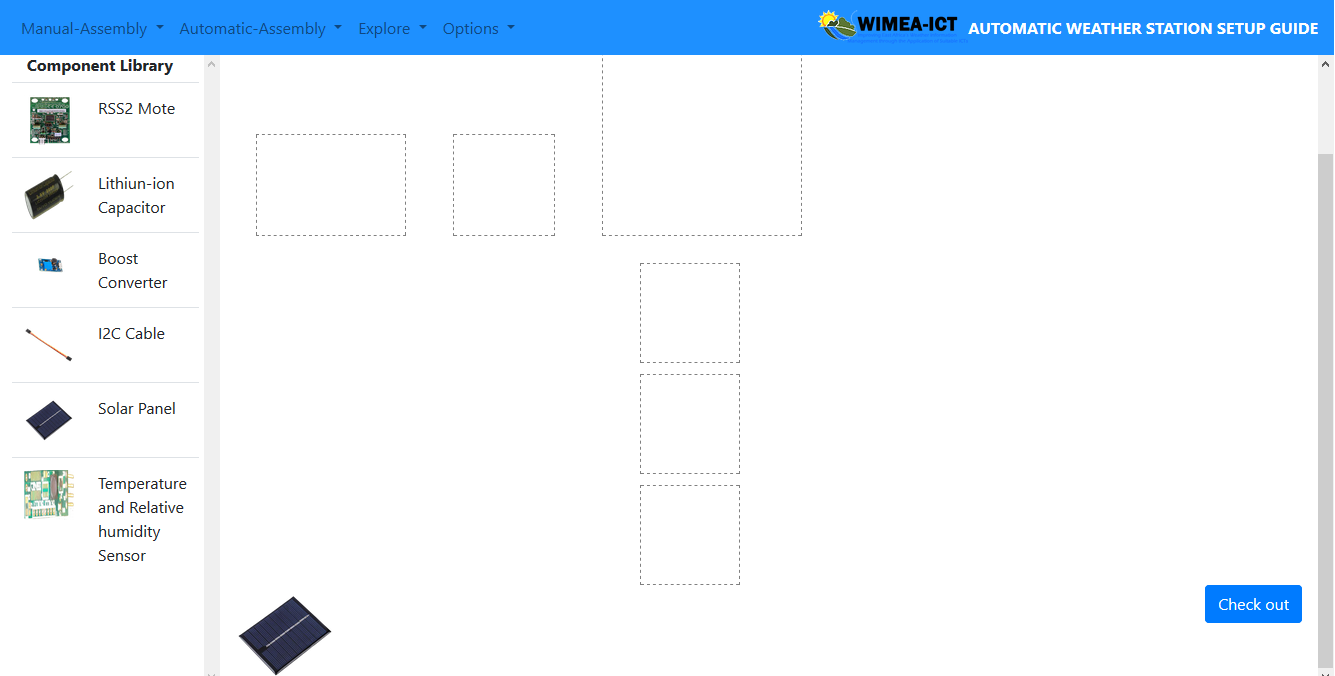


Figure 3. 3 Loaded components and layout for the 2-meter node.

## 3.4 Drag and drop of the components.

To assemble a node, you are required to drag and drop the components to the appropriate position on the layout of the node.

Figure 3. 4 the selected component appears on the simulation environment.

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

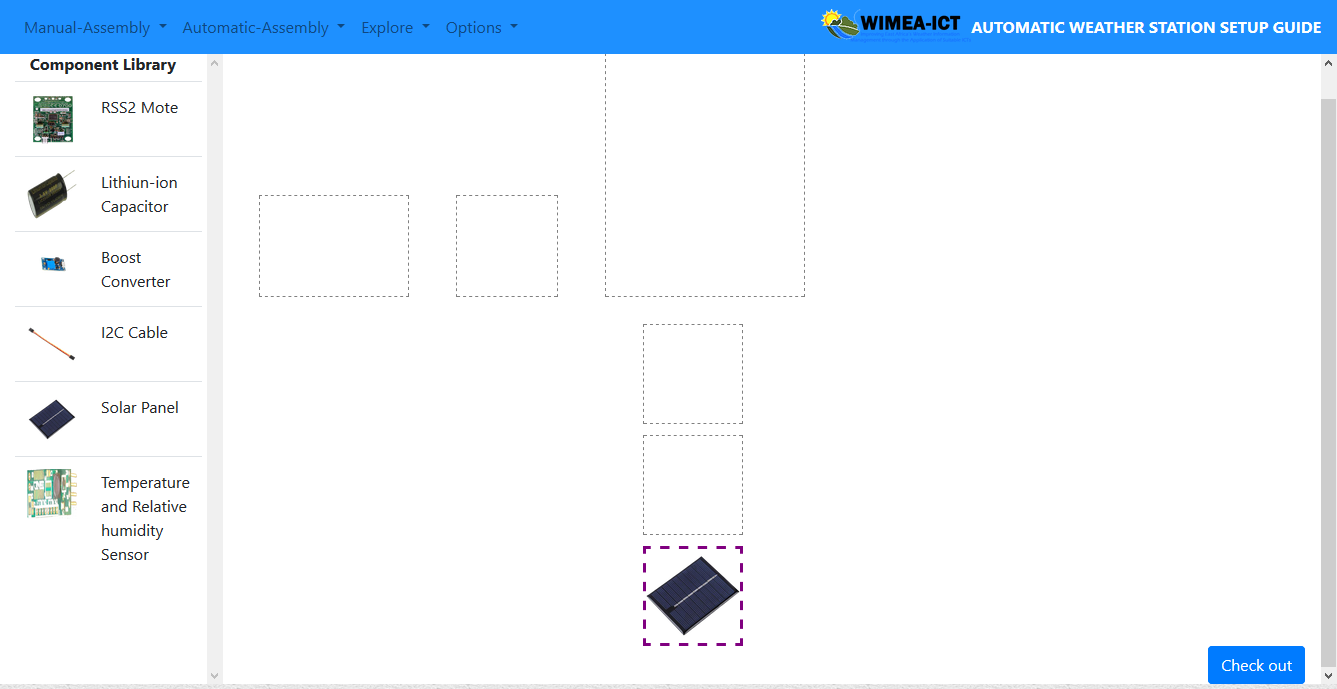


Figure 3. 5 The component is placed or dropped in its appropriate location.

## 3.5 Award points for each connection made.

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

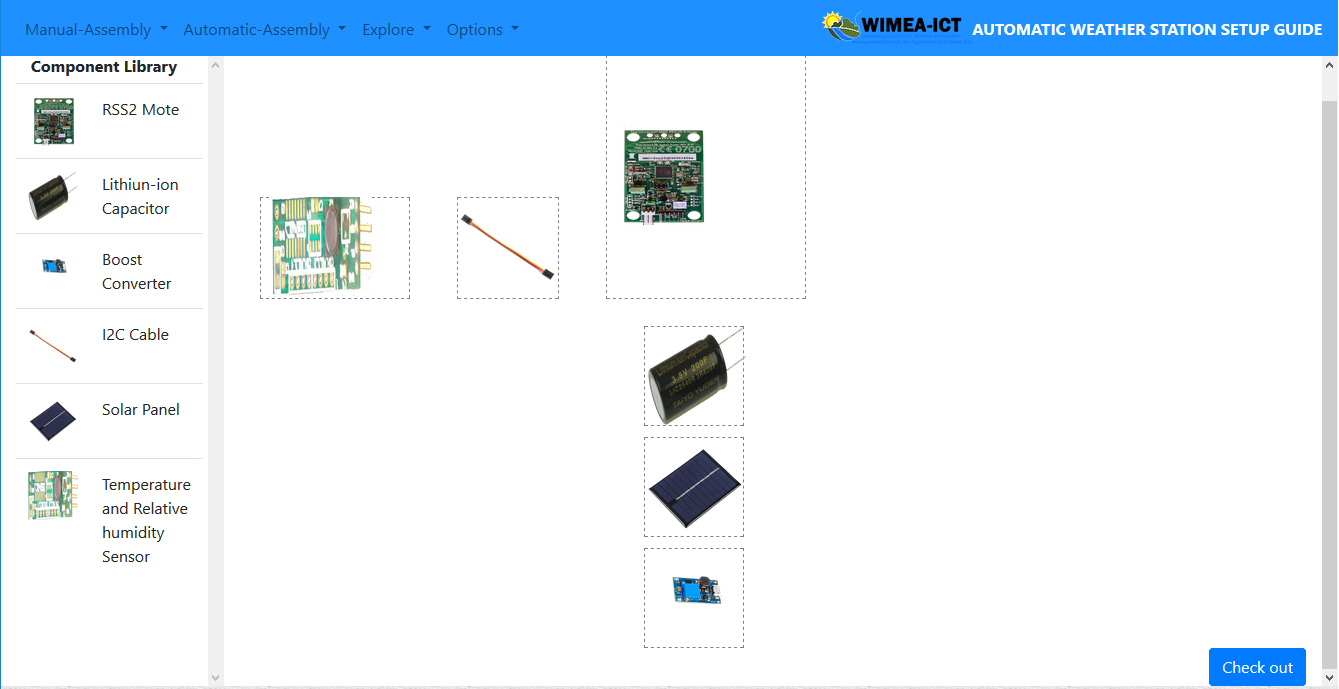


Figure 3. 6 All components assembled in the layout.

Check out button in the right bottom corner of the simulating environment.

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

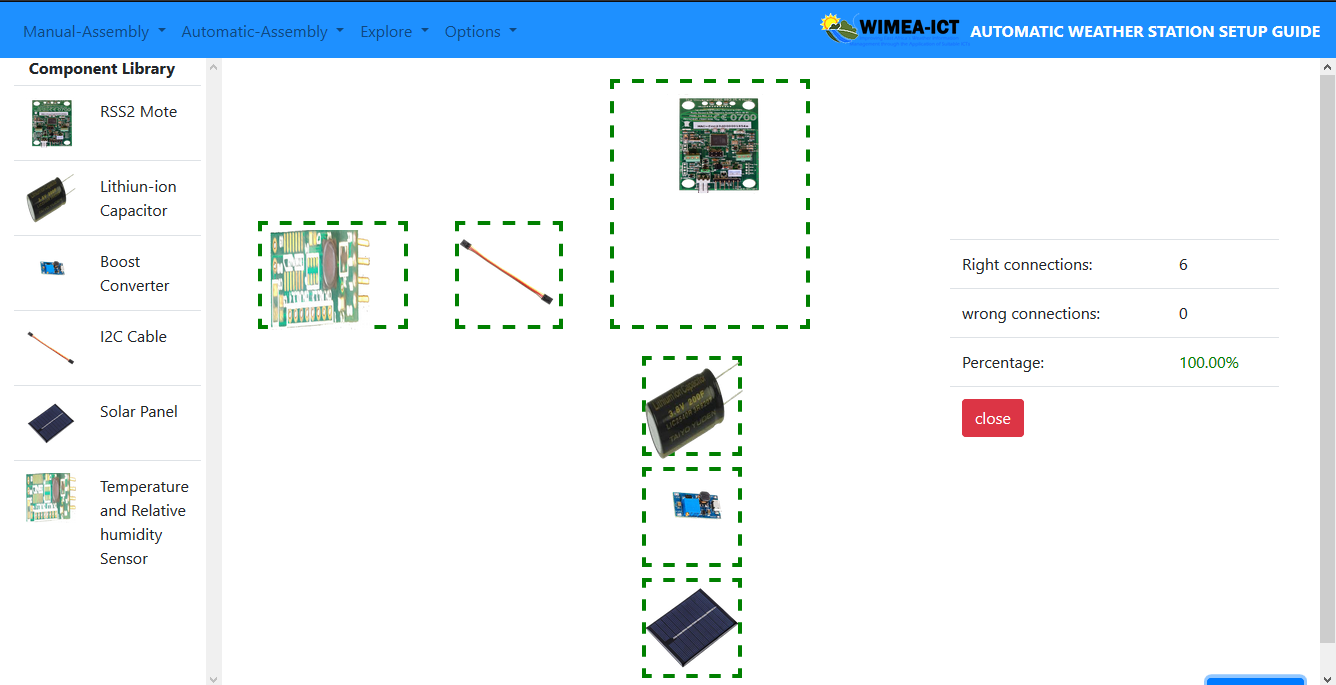


Figure 3. 7 layout borders highlighted green for the wright connection.

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

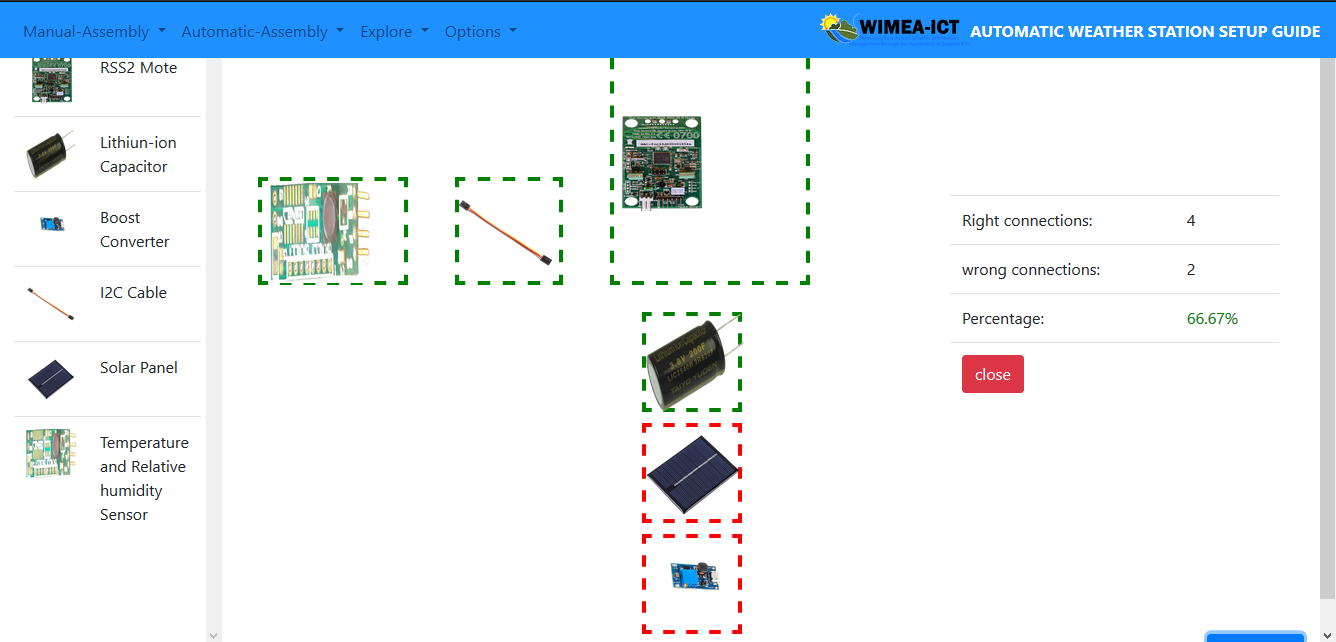


Figure 3. 8 Wrong connections highlighted red.

## 3.6 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 3.9 below the boost converter was selected

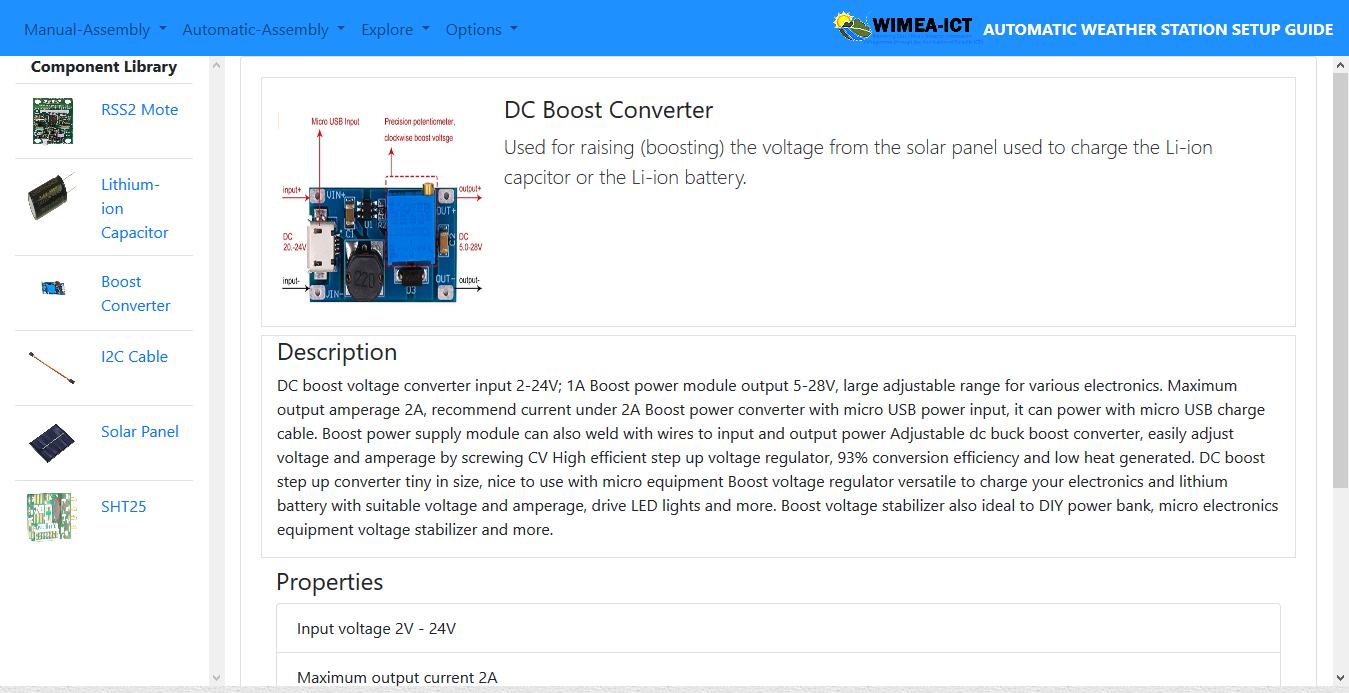


Figure 3. 9 Boost convertor Explore mode

## 3.7 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.

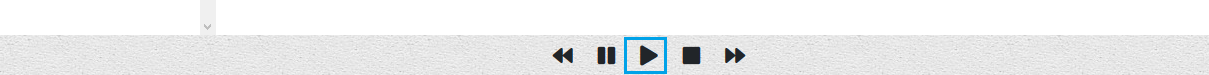


Figure 3. 10 Play simulation

# Appendix C: Allocation of Tasks

Allocation of the system development tasks amongst the team members.

Table 7. 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: | |