WEATHER INFORMATION MANAGEMENT OF EAST AFRICA IMPLEMENTATION, TESTING AND VALIDATION **REPORT** FOR: WEATHER DATA API AND ITS APPLICATIONS

Prepared by:

STUDENT NAME	REGISTRATION NUMBER	STUDENT NUMBER
MUTESASIRA JOVAN	14/U/729	214000361
KASOZI MULONDO MOSES	14/U/7239/PS	214018794
KYEBAMBE SARAH	14/U/8490/EVE	214014236
OKWII STANLEY	14/U/14042/PS	214012931

Name	**Role**	**Date**	**Signature**
Ms. Mary Nsabagwa	Supervisor		

Date: 07/06/2018

Version: 1

Contents

1 Introduction

1.1 Overview of System

Weather Information Management in East Africa ICT (WIMEA-ICT) project aims at improving the accuracy and access to weather information by the communities in the East African region through suitable ICTs.

Weather Data API provides the means of accessing weather data from Weather Data Repository (WDR) which is a component of WIMEA-ICT. WDR receives data from both the Automatic Weather Station(AWS) and manual weather stations across Uganda. This data is required for weather assimilation, modelling, forecasting and also as an input to the message switching system at the National meteorological center.

The Weather Data API provides an abstraction of the WDR data without direct access to its underlying database. We developed two applications that provide input data to Weather Research and Forecasting model (WRF) which is a weather modelling tool and the Message switching system at National Meteorological Center, this data can be accessed through the weather data API.

This section consists of the general description of the project's functionality, context and design that include;

The API is web based thus running on the latest browsers such as google Chrome, Mozilla Firefox and many others. It is accompanied by an interface that has a Home, Documentation and About page. The Home page, users have to sign-in to acquire an account so as to access the API key which is unique to every user. This key is appended to the API url enabling the user to access the API services. The documentation page contains the guidelines of how to use the API to access the weather data. The About page explains who can access the API data and what kind of data ranges can be acquired, for example the current weather data, Weather data recorded for 7 days and Weather data recorded for 30 days.

The Weather Data messaging application is a web based application running on a Linux server, the application transmits (sends) the data through an email and SMS to the message switching system in the Table Driven Code Format (TDCF) where the data is analyzed and later sent to Nairobi.

The Little_R generator Application is also running on Linux platform. The application sends the data in its required data format (Little-R) to the weather research and forecasting modelling software where this data is used for modelling weather predictions.

1.2 Overview of document

This document describes the implementation, testing and validation findings for the Weather Data Application programming Interface. It is divided into the following sections:

Section 1.0.0: This section gives an overview of the system explaining how the system operates and an overview of the document.

Section 2.0.0: Requirements acceptance test specification

This section gives the requirements and system acceptance test showing what the system does and the objective criteria on how the computer system should be tested to ensure that the requirements are fulfilled and that the computer system performs as required in the environment in which it will be used. This is represented in a table format showing the version of the requirements, inputs, outputs, limitation, safety, default setting, version control, dedicated platform, installation, service and maintenance and the errors and alarms..

Section 3: Design Output

This section gives the full description of the System design and implementation process which is relevant when developing new systems and handling changes subjected to existing systems. The output from this life cycle phase is a program approved and accepted for the subsequent inspection and testing phase.

Subsection 3.1: Development Plan, this describes the tools, manpower and methods that were used to develop the system.

Subsection 3.2: Design input and output, which is in a tabular format has the following subsections;

Subsection 3.3: Design changes, this section describes the different changes that occurred during implementation of the system.

Section 4: Inspection and Testing.

This section gives the System inspection and testing plan and documentation of the test plan. It also contains the requirements compliance with the testing, the acceptance test specification, the approach, complexity, risks, and the intended and expected use of the computer system. Subsection 4.1.1: Test plan and performance.

This subsection identifies all the elements that are to be tested, the expected results of testing, how the testing is carried out, importance of tests to the system, the scope and level of the tests carried out and the different types of tests along with their expected results.

Section 5: Installation and system acceptance tests.

It details the validation of the installation process to ensure all system elements are properly installed in the host system and that the user obtains a safe and complete installation. It also includes the precautions of using the third party software environment such as Microsoft Windows.

Section 6: Performance, Service maintenance and phase out.

This section contains descriptions of Performance, servicing, maintenance, and phase out stages. In this phase the computer system is in use and subject to the requirements for service, maintenance, performance, and support. This phase is where all activities during performance reside and where decisions about changes, upgrades, revalidation, and phase out are made.

Section 7: Conclusion and recommendations.

This section summarizes the whole project and makes remarks and highlights several issues about the project.

Section 8: User manual.

This section contains the user manual for the system which gives details on how the system can be used and where to go for help on using the system.

2. Requirements and system acceptance test specification.

The requirements describe and specify the system completely and are the basis for the development and validation process.

Table 1 Requirements specifications.

Topics	**Requirements specification**
Version of requirements.	Version 1.0 first document made by BSE 18-8 on 13/11/2017. Version 1.1 removing conflicting requirements by the supervisor of BSE 18-8 on 4/12/2017. Version 1.2 made diagram and text formatting clear by the BSE 18-8 supervisor on 13/12/2017.
Input.	**Inputs required for other users to use the API include;** ● Weather data from Weather data repository. ● The Login credentials for the API user. **Input are required by the LITTLE_R Generator application**;

	● Weather data from the weather data API. **Inputs for the Weather data messaging application include**: ● Administrator's username and password. ● Recipient's email, name and phone number. ● Weather data from the weather data API.
Output	Weather data API. ● User API key. ● Current Weather data in JSON format In the Little_R generator application, ● LITTLE_R file, this file is an ASCII-based observation file format. It contains weather data converted into a format which supports its use in the WRFDA model. LITTLE_R is designed to be an intermediate format so that WRFDA might be able to assimilate as many observation types as possible in a universal manner. In the Weather Data messenger ● Weather data in Table driven code format. ● Email and SMS. ● Recipient's information.
Limitations	● Weather data errors due to wrong data formats that are at times input into the weather data repository. ● Unreliable internet access, because the weather data messaging application uses the internet services to send emails and sms every after 30 minutes. ● R studio has to be installed and run appropriately since the Little_R Generator requires WRF library from R studio which contains write_obs function that writes observations into LITTLE_R format. ● The Little_R Generator relies solely on the weather data from the API as the major data input. Therefore, inaccurate data from the API implies that Little_R file data will have wrong data.

Safety	In the Little_R generator application, safety has been in-built into application in the following ways: - ● The API JSON object containing the weather data is inspected to verify that it has values. ● The generated LITTLE_R file is checked to verify that it conforms to the World Meteorological Organization standards before it is released for use. In the weather data messenger application, Phone number and email validation at the time of registering a recipient.
Dedicated platform	The application runs on Ubuntu server and is also accessible online by using web browsers such as google chrome, Mozilla Firefox and opera. The LITTLE_R Generator is executed at regular intervals by a Linux bash script running on an Ubuntu 16.04 server. The bash script calls an R script to write the observation data into LITTLE_R format. The LITTLE_R Generators runs as a daemon on the server.
Installation	The weather data messaging application shall be deployed on the server

3. Design and implementation process

3.1 Design inputs and outputs

The design output must meet the design input requirements, contain or make references to acceptance criteria, and identify those characteristics of the design that are crucial to the safe and proper functioning of the product. The design output should be validated prior to releasing the system for final inspection and testing.

Table 2. Design output checklist

Topics	**Design output**	
Implementation (coding and compilation)	The implementation tools used to implement the Weather API include Xampp MySQL Database Management System, Atom IDE was used for coding, Laravel php framework was used for compilation, Chrome browser and postman testing tool were used for testing.	
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
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	

Utilities for validation and testing	Linux Server used to host the weather data API, Weather Messenger Application developed by the same team to send emails of weather data to the Entebbe NMC message switching System using this developed API, Modelling data transmitter Application developed by the same team which still uses this API to send weather data in format called little R required for data assimilation done to provide more accurate results in weather forecasting, postman a common testing tool for testing API endpoints.
Inactive code	None
Documentation	Documentation section is provided on the Weather API web interface and shows the steps the user passes through to use this API.

4. Inspection and testing

The inspection and testing of the system is planned and documented in a test plan. The extent of the testing is in compliance with the requirements, the system acceptance test specification, the approach, complexity, risks, and the intended and expected use of the system.

The test plan is created during the development or reverse engineering phase and identify all elements that are about to be tested. The test plan should explicitly describe what to test, what to expect, and how to do the testing. Subsequently it should be confirmed what was done, what was the result, and if the result was approved.

4.1 Test Objectives and types.

Topics	**Test plan and performance**
Sequence of tests	Test Case: Enter valid Username, email and password. Procedure: 1. Enter User Name 2. Enter Email.

	3. Enter Password. 4. Click sign in: Test Data: TC001_TD001 Expected output: Successfully registered message is displayed. Test Case: Enter invalid email. Procedure: 1. Enter User Name 2. Enter Email. 3. Enter Password. 4. Click sign in: Test Data: TC001_TD002 Expected output: Invalid email message. Test Case: Enter valid User Name and valid Password. Procedure: 1. Enter User Name 2. Enter Password. 3. Click Login: Test Data: TC001_TD003 Expected output: Logged in successfully.

	Test Case: Enter invalid User Name and invalid Password. Procedure: 1. Enter User Name 2. Enter Password. 3. Click Login Test Data: TC001_TD004 Expected output: A message " invalid username or password " is shown. Test Case: Click the copy below the API Key provided. Procedure: Click copy button Test Data: TC001_TD005 Expected output: A message " invalid username or password " is shown. Test Case: Enter valid admin Name and valid Password. Procedure: 1. Enter Admin Name. 2. Enter Password. 3. Click "Login" button. Test Data: TC001_TD006

	Expected output: Access to recipients' page. Test Case: Enter invalid admin Name and Valid Password. Procedure: 1. Enter User Name 2. Enter Password. 3. Click Login Test Data: TC001_TD007 Expected output: Invalid user name or password. Test Case: Enter valid Recipient Name, Email and valid Phone number. Procedure: 1. Enter Recipient Name. 2. Enter Recipient email. 3. Enter valid phone number. 4. Click "Login" button. Test Data: TC001_TD008 Expected output: recipient successfully registered. Test Case: Enter valid Recipient Name, invalid Email and valid Phone number. Procedure:

	1. Enter Recipient Name. 2. Enter Recipient email. 3. Enter valid phone number. 4. Click "Login" button. Test Data: TC001_TD009 Expected output: A message "Invalid email" shown Test Case: Enter valid Recipient Name, valid Email and invalid Phone number. Procedure: 1. Enter Recipient Name. 2. Enter Recipient email. 3. Enter valid phone number. 4. Click "Login" button. Test Data: TC001_TD010 Expected output: A message "Invalid phone number" shown Test Case: Automatically Send email registered recipients. Procedure: Automatic sending of the email to registered recipient. Test Data: TC001_TD011

	Expected output: Recipient receives email containing TDCF data. Test Case: Provide a valid api url and key to access weather data. Procedure: Run the R Script. Test Data: TC001_TD012 Expected output: Successful creation of little R file. Test Case: Make an api request that returns a json object with some null values. Procedure: 1. Select a station, and date to retrieve weather data from. 2. Run the R Script. Test Data: TC001_TD012 Expected output: Display no data found message. Test Case: Provide a wrong/incorrect url for accessing weather data. Procedure: Run the R Script. Test Data: TC001_TD014 Expected output: Display invalid url message.

Configuration tests	Configuration are made on Wimea Linux server to run the Weather data API, the Weather Data Messaging Application and the Little_R Generator.
Action if errors	The API user receives error messages in case he or she makes mistakes while signing in for API KEY.

4.2 Test Results

TEST CASE ID	**TEST CASE**	**TEST RESULTS**
TC_SIGN IN FOR API KEY_001	Enter valid Username, email and password.	Successfully registered message is displayed.
TC_SIGN IN FOR API KEY_002	Enter invalid email.	Invalid email message.
TC_LOGIN_003	Enter valid User Name and valid Password.	Logged in successfully.
TC_LOGIN_004	Enter invalid User Name and invalid Password	A message " invalid username or password " is shown.
TC_GET API Key_005	Click the copy below the API Key provided.	API is being highlighted and copied.
TC_ADMIN_LOGIN_006	Enter valid admin Name and valid Password.	Successful login.
TC_ADMIN_LOGIN_007	Enter invalid admin Name and Valid Password	Invalid username or password.

TC_REGISTER_RECIPIENT_008	Enter valid Recipient Name, Email and valid Phone number.	recipient successfully registered.
TC_REGISTER_RECIPIENT_009	Enter valid Recipient Name, invalid Email and valid Phone number.	Invalid email error message.
TC_REGISTER_RECIPIENT_010	Enter valid Recipient Name, valid Email and invalid Phone number.	Invalid phone number error message.
TC_SEND_WEATHER DATA_011	Automatically Send email registered recipients.	Recipient receives email containing TDCF data.
TC_LG_012.	Provide a valid api url and key to access weather data.	Successful creation of little R file.
TC_LG_013	Make an api request that returns a JSON object with some null values.	Display no data found message.
TC_LG_014	Provide a wrong/incorrect url for accessing weather data.	Display invalid url message.

5. Installation and system acceptance test

The validation of the installation process ensures that all system elements are properly installed in the host system and that the user obtains a safe and complete installation, especially when installing software products.

5.1 Installation method

A manual method was used to install the weather data API on an Ubuntu server running Ubuntu 16.04.4 LTS operating system. The weather data messaging application and Little_R generator are also installed on this server.

5.2 Installation media

A git repository of the WDR API project which consist of weather data API, weather data messaging application and Little_R generator was created and changes pushed (uploaded) to the cloud. The same git repository was cloned(downloaded) and configured to run on the server environment.

5.3 Input file

Weather data from the Weather Data Repository is main input for the Weather Data API. The latter obtains both manual weather observations and automatic weather observations. Users of the API also provide data to the system as they register to use its services.

5.4 Supplementary files

A user manual that will guide the users on how to operate the system shall be provided, together with a readme file for quick system configuration.

5.5 Installed components

The project was developed based on the laravel php framework v5.5. Among the components that require to be on the server is php v7.0.30 or higher. The project depends on the following libraries that need to be installed as well: -

- passport
- tinker
- Mailgun-php

5.6 Installation qualification

A checklist of the Installation and system acceptance test

Table 3 Installation summary.

Topics	**Installation summary**
Installation method	☐ Automatic - auto deployment of the project on the server. ☑ Manual - upload project files on to the server. Comments: Used github to upload the project files to the server.
Installation media	☑ Download from the internet ☐ Diskette(s) ☐ CD-ROM

	☐ Flash Disk ☑ Source disk folder(PC or Network) Comments: Files were uploaded to a GitHub repository from a source folder and downloaded from the internet using GitHub.
Input files	● Weather data repository ● User input
Supplementary files	● User manual ● Readme file
Installed components	● Passport ● Tinker ● Composer. Json ● Mailgun-php ● Php 7.0.30 or higher

Installation Procedure Check

Topics	**Installation procedure**	*Date/initials*
Authorization	Person responsible: JOVAN MUTESASIRA SARAH KYEBAMBE STANLEY OKWII MOSES MULONDO	20/05/2018

Installation test	☑ 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: System was tested in a copy of the true environment in order to protect original data from possible fatal errors due to using a new system.	27/05/2018

System Acceptance test

The system acceptance test is carried out in accordance with the system acceptance test specifications after installation. The software product may subsequently be approved for use.

Topics	**System Acceptance Test**	*Date/initials*
Test environment	☑ The actual operating environment (site test) ☐ A true copy of the actual environment ☐ External environment (supplier factory test) Comments: The system has been tested on Windows and Linux operating systems.	OS
Test performance	☑ Installation and version ☐ Startup and shutdown ☐ Selected or critical requirements ☑ Selected inputs ☐ Selected outputs ☑ Selected functionality ☐ Performance vs. user instructions	KS

	Comments: As system functional requirements, the system's components were tested one at a time right away from installation. Some inputs and some selected outputs were also tested.	
User level test	☑ Tested on operator user level ☐ Tested on super-user level ☑ Tested on system administrator level ☐ Tested on overall system manager level ☑ Education and training documented ☑ System user manuals available	KMM
Result of testing	☐ Testing approved Comments: Tests were not fully approved because the system was still under development.	MJ

6. Performance, servicing, maintenance, and phase out

6.1 Service and maintenance

6.1.1 Support concerning maintenance:

The system source code contains well documented comments which can be helpful during the maintenance of the system.

6.1 .2 System upgrades

Using a version control tool like Git, the system can be updated easily. Git also offers distributed version control and source code management functionality.

6.2 Performance and Maintenance

Table Performance and maintenance details.

Topics	**Performance and maintenance**
Problem / solution	**Problem:** System generates a long API key of more than 200 character based on Laravel passport' Bearer token which make is difficult to use and chances of incorrectly copying the key also arise. **Solution:** Dropped the Bearer token and adopted a Url parameter passed token.
Functional expansion and performance improvement	● The Weather API retrieves a small but specific set of data for a given purpose using the specified arguments. ● Allow all endpoints to specify a target station using an optional station parameter, reducing data traffic of selecting weather data for all stations. ● An additional endpoint for retrieving data for all active stations such that an API user can select the right region/station to focus on. ● Need to allow for zipping of large JSON data to increase its download speed in future.

7. Conclusion

The development of the Weather data API, Weather data Messaging application and the Little_R Generator application will enhance to stream line the data flow from Weather Data Repository to the weather modelers, Nation Meteorological center and to any other developer interested in making applications that need the use of Ugandan weather data.