

WIDS-USSD APPLICATION TECHNICAL DOCUMENT

Version 1



Table of Contents

INTRODUCTION	3
How to use the application	3
Accessing daily or Seasonal Forecast via USSD APP.....	3
Accessing Marine Forecast via USSD APP.....	4
GUIDELINES FOR TECHNICAL CUSTOMIZATION	5
Inception-Technical Requirements	5
Adding new languages	5
Add new display menu.....	6
Handle menu options processing	7
Add logging to a function.....	8
Sending SMSs	9

INTRODUCTION

The Weather Information Dissemination System (USSD-APP) is one of the research projects of WIMEA-ICT and was initially designed to be used in Uganda, and the whole functionality and structure of the application was implemented and designed based on forecast structures provided by Uganda National Metrological Authority (UNMA).

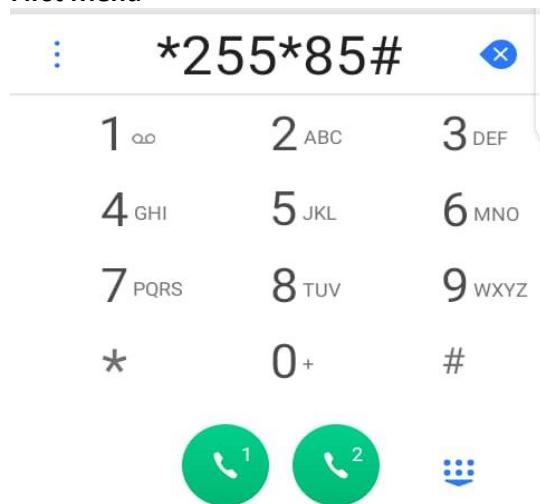
The major goal of this project module was to provide rapid dissemination of weather information to the stakeholders especially farmers in the local areas of Uganda (or farmers who can't access our web version). This was achieved based on the current USSD users that assess the system, via a pull request.

How to use the application

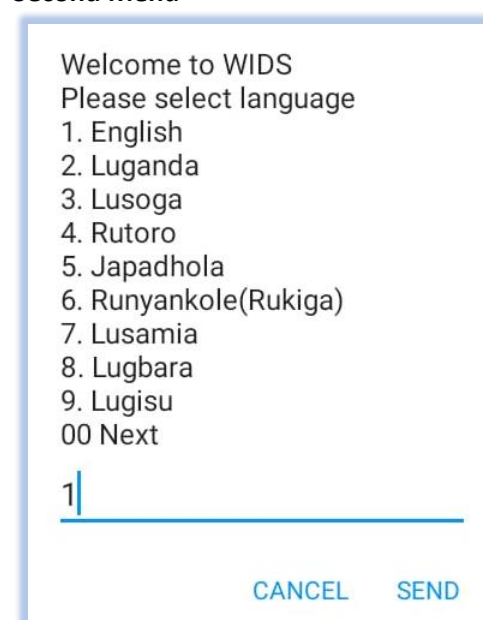
The system can be accessed by dialing *255*85# and following the prompts as shown in figure 1.

Accessing daily or Seasonal Forecast via USSD APP

First Menu



Second Menu



Third Menu

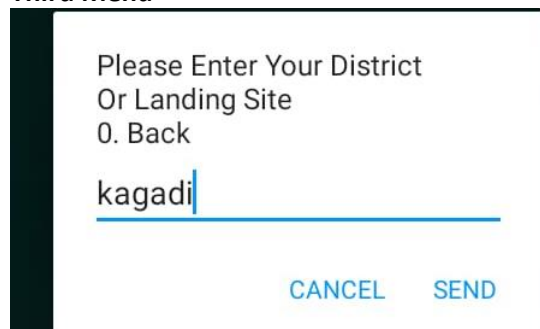
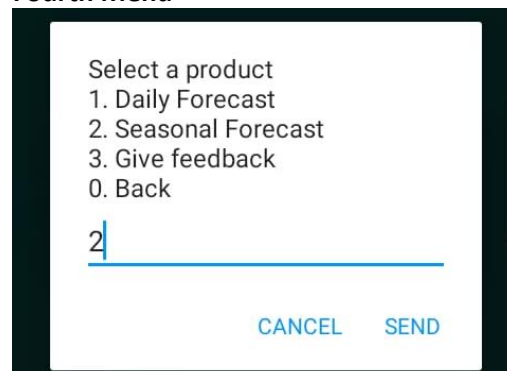


Figure 1: Enter your district

Fourth Menu



Fifth menu

Seasonal Forecast
Select a Sector
1. Water
2. Health
3. Infrastructure, works & transport
4. Harvesting
0. Back

4

CANCEL SEND

Sixth Menu

KAGADI, Seasonal Forecast
1. Confirm Submission
2. Back

1

CANCEL SEND

Figure 2: These sectors vary with the season.

Seventh Menu

You will receive a message shortly
Thank you for Contacting Us.

OK

Accessing Marine Forecast via USSD APP

The 24 hours marine forecast released twice a day gives people guidance and weather information about when its safe to travel on the lake and when to stay on the shores. When there is severe weather, people planning to use the lake will have information to take proactive actions whether to wear a life jacket, carry enough fuel with them when they decide to sail on the water.

This forecast information can be accessed via the USSD-APP as follows:

First Menu

*255*85#

1 op 2 ABC 3 DEF
4 GHI 5 JKL 6 MNO
7 PQRS 8 TUV 9 WXYZ
* 0+ #

1 2

Figure 3: Make a pull request

Third Menu

Second Menu

Welcome to WIDS
Please select language
1. English
2. Luganda
3. Lusoga
4. Rutoro
5. Japadhola
6. Runyankole(Rukiga)
7. Lusamia
8. Lugbara
9. Lugisu
00 Next

1

CANCEL SEND

Figure 4: Forecast is always Disseminated in English. Choose option 1

Fourth Menu

Please Enter Your District
Or Landing Site
0. Back

Ggaba

CANCEL SEND

Figure 5: Enter destination- landing site

Marine Forecast
Select a period
1. Today Morning
2. Today Afternoon
3. Give feedback
0. Back

1

CANCEL SEND

Figure 6: Select period of the day

Fifth Menu

GGABA, Sunday Night before
midnight
1. Confirm Submission
2. Back

CANCEL SEND

Figure 7: Confirm your request

Sixth Menu

You will receive a message
shortly
Thank you for Contacting Us.

OK

GUIDELINES FOR TECHNICAL CUSTOMIZATION

Inception-Technical Requirements

Before thinking about implementation of the USSD-APP the following inception requirements must be fulfilled.

1. An Unstructured Supplementary Service Data (**USSD**) Code (e.g., *284#) registered with legal service provider in Uganda.
2. Short Message Service (SMS) service provider. For example, SimplySMS or egoSMS
3. Voice message Service provider, in case of disseminating audio forecasts.
4. WIDS-Web system. The USSD-APP works on top of the robust WIDS-web system.

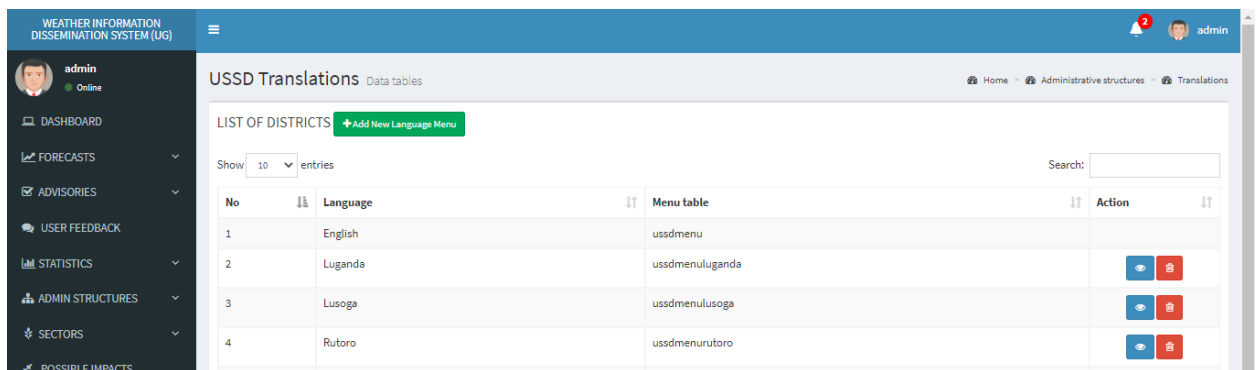
Adding new languages

This can be archived via the WIDS web admin panel.

Login to the Web as the Admin user. Go to “**USSD Management**” menu then “**USSD Menu Settings**”

“**USSD Management > USSD Menu Settings**”

From the menu that displays, click on “**Add new Language Menu**”



You are provided with an input field for the language name *<<language name>>*, a table having first column with the English version of the words to be translated and the other column with empty spaces for their corresponding translations in the specified language.

Fill all the fields with the translations and save the changes.

This creates a table “*<<language_name>>menulanguage*” in the database, holding the translations specified in the form fields.

NOTE: The Language option will only show up on the USSD if there is a seasonal forecast entered for that language.

Add new display menu

Add the ussdmenu in the ussdmenu table in the database

In the **UssdLogic.php** file, under the function “Display”

Create an “else if” statement clause at the end before the “else” clause , checking if the \$menuName == ‘menu_name of your choice’, e.g

```

else if($menuName == "menu_name"){
    $responseMsg = $this->DBQueryFunctions->loadUssdMenu("menu_name", $menu_table);
    $responseMsg .= "-0. ".$this->DBQueryFunctions->loadUssdMenu("back", $menu_table);
}

```

The function in the “if” clause, retrieved the menu to be displayed from the database with a menu name passed in as an argument.

Consider the sample below as shown from the database

<input type="checkbox"/>	Edit	Copy	Delete	district	13	Please Enter Your District-Or Landing Site
<input type="checkbox"/>	Edit	Copy	Delete	invalididistrict	14	District Unknown

argument

Menu displayed

Handle menu options processing

For each menu display created and to be used as a standalone display with some functionality, the backend functionality for a display menu is first declared in another file.

That is to say, to create a backend functionality for a menu, you need to declare the function for the specified menu_name in the **UssdAppMain.php** file.

This can be done in the “switch” statement.

Create a new “switch case” with the <<menu_name>>, then in it, create a function that will handle processing of the menu displayed on mobile devices.

Consider the sample below.

```

case "repeat":
    $_SESSION['menu-Opt'] = $logic->ProcessRepeated($receiver->getInput(), $sessionId, $msisdn);
    break;

case "menu_name":
    $_SESSION['menu-Opt'] = $logic->ProcessMenuOptions($receiver->getInput(), $sessionId, $msisdn);
    break;
}

```

UssdAppMain.php file

Finishing up with the function

Under the **UssdLogic.php** file, within the “class”, create a function with a name similar to that declared in the **UssdAppMain.php** file, for the menu to be displayed.

The function will handle the processing of user input as they traverse the different menu options if given any.

Consider the sample below.

User input inform of numbers
or text

Session Id

User's phone number

```
function processMenuOptions($input, $sessionid, $msisdn){
    $menuname = "";
    .....
    $this->Display($menuName);
    return $menuname;
}
```

Optional to only display that are not final in the USSD cycle

NOTE: For every function created, there must be a return value, unless it's the last function to be processed in the USSD. The return value is always a menu name for the next menu to be displayed for the user

For extra sample code on how to handle the menu options processing, refer to the available coded functions in the same file, as each **processing function** has a prefix “**Process**”.

Add logging to a function

This can be easily achieved in the **UssdLogic.php** file, with the lines of code as shown below

```
function processMenuOptions($input, $sessionid, $msisdn){
    $menuname = "";
    .....

    $Log = $this->DBQueryFunctions->LogUpdates($msisdn,$sessionid, $_SESSION['district'], $menu, $menuOption );

    $Log2 = $this->DBQueryFunctions->LogSessions($msisdn, $sessionid, $_SESSION['language'], $_SESSION['district'],
        NULL, NULL, NULL, NULL, NULL);

    $this->Display($menuName);
    return $menuname;
}
```

Logging is carried out using 2 functions as seen above, its done to ensure consistency and accuracy of information stored in both data tables based on the USSD usage.

Function 1

`$Log = $this->DBQueryFunctions->LogUpdates($msisdn,$sessionid, $_SESSION['district'], $menu, $menuOption);`

\$msisdn	represents the user phone number
\$sessionid	the current user session
\$_SESSION['district']	district selected, if any, otherwise stores null
\$menu	takes in a string, represents the name for the current menu that the user is currently accessing.
\$menuOption	the option a user chooses or an input taken from the user while accessing the menu display

Function 2

The kind of logging is applicable to the different USSD menus, as each argument passed in, represents a specific Menu display of the USSD.



```

$Log2 = $this->DBQueryFunctions->LogSessions($msisdn, $sessionid,
$_SESSION['language'], /** the language selected by the user from the starting point **/
$_SESSION['district'], /** District selected after the language menu **/
NULL, /** add if menu reached, product selected from the product menu **/
NULL, /** add if menu reached, period selected while handling subscription menu **/
NULL, /** add if menu reached, advisory selected after selecting the seasonal
forecast, then from the advisory menu displayed, can add option if applicable **/

NULL, /** add if menu reached, message type to be received by the user **/
NULL); /** add if menu reached, final user confirmation on the USSD final menu**/

```

The function can be modified in the **DBQueryFunctions.php** file as more menu displays may come up. Below is the sample database structure based on all USSD menu levels

	#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	1	id 	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/>	2	phone	varchar(15)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	3	sessionId	varchar(200)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	4	language	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	5	district	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	6	product	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	7	period_selected	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	8	advisory	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	9	msg_type	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	10	confirmation	varchar(200)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/>	11	date	timestamp			No	CURRENT_TIMESTAMP	

Ussd_transactions_2021

Sending SMSs

Sending an SMS requires you to have an account with a bulk messaging service provider say simplySMS, egoSMS and others.

The *messages()* function in *DBQueryFunctions.php* file is used to hit the API provided by the bulk messaging service provider.

```

function Messages($message,$phoneNumber){

    $resp = "";
    try{
        $ch = curl_init();
        curl_setopt_array($ch,array(
            CURLOPT_RETURNTRANSFER =>1,

            CURLOPT_URL =>'http://simplysms.com/
                getapi.php?email=YOUR_API_EMAIL&password=YOUR_API_PASSWORD&sender=8777&message='.$
                message.'&recipients='.$phoneNumber,
            CURLOPT_USERAGENT =>'Codular Sample cURL Request'));

        $resp = curl_exec($ch);

        curl_close($ch);
    }catch(Exception $e){}
    return $resp;
}

```

The *messages()* function takes in two arguments; message you want to send to the user and the user's phone number.