

Project Development Phase

Sprint-1

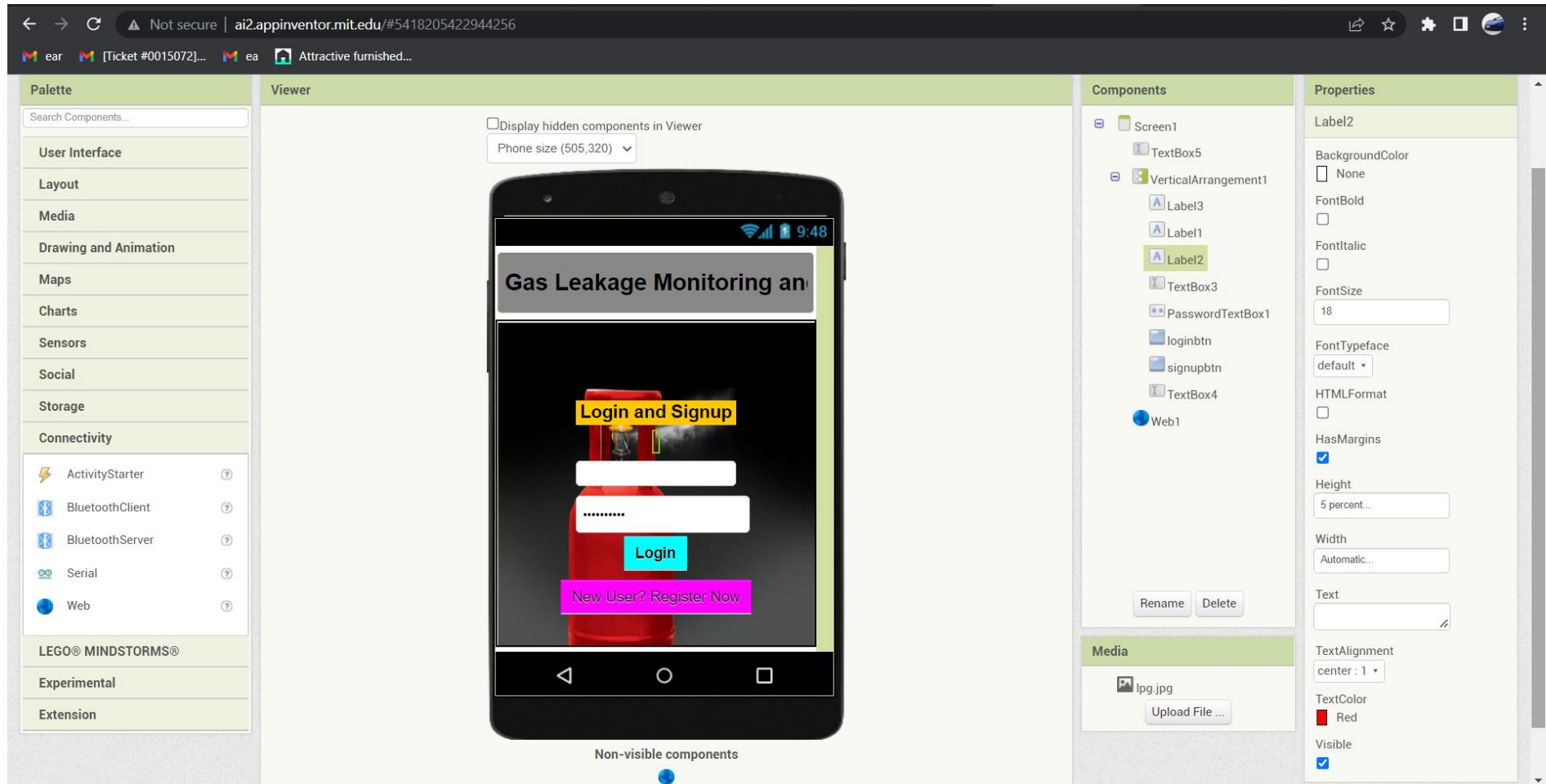
Date	2 November 2022
Team ID	PNT2022TMID35841
Project Name	Gas Leakage Monitoring and Alerting System

Sprint Target:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can start by registering my credentials so that I get notifications in case of a gas leakage	10	High	SUDARSHAN A R, B ADITYA, C PRADUMNA
		USN-2	As a user, I will receive confirmation once I have registered for the application.	5	High	SUDARSHAN A R, B ADITYA, C PRADUMNA
	Login	USN-3	As a user, I can log into the application by entering my credentials	5	High	SUDARSHAN A R, B ADITYA, C PRADUMNA

Introduction:

In this Sprint-1, we have created a Gas Leakage Monitoring and Leakage Detection application using MIT app inventor 2 and Node-RED API which allows user registration and login.



Registration/Signup

New users to the app can enter their username, password and mobile number to register.

The screenshot displays the MIT App Inventor web interface in a browser. The address bar shows the URL `ai2.appinventor.mit.edu/#5418205422944256`. The page title is "Gas_leakage". The interface is divided into three main sections: "Blocks", "Viewer", and "Media".

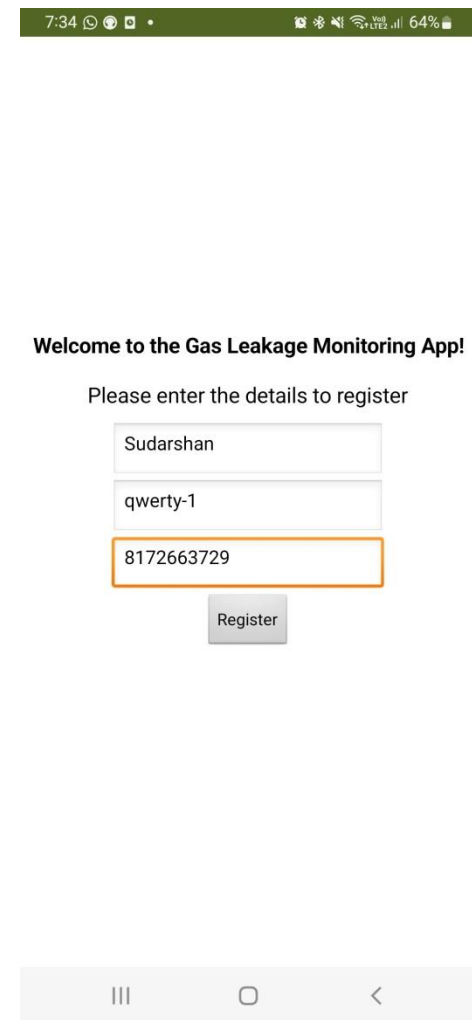
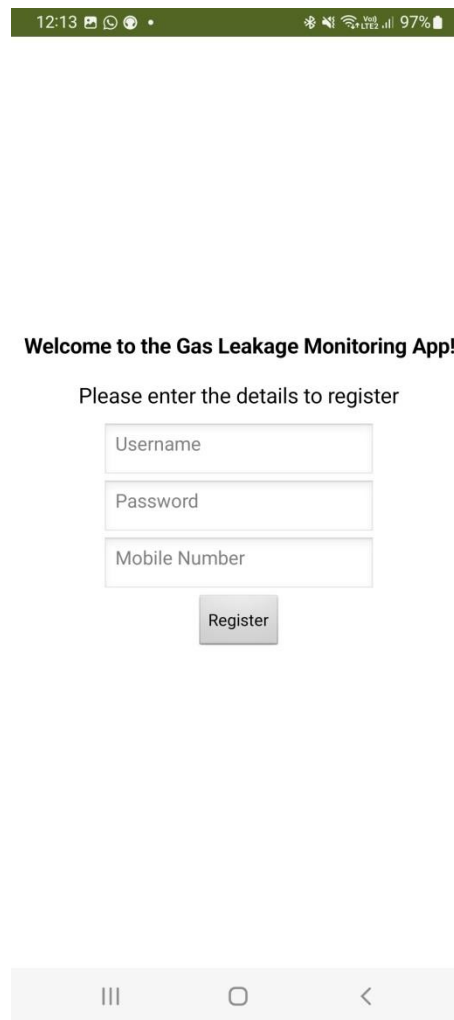
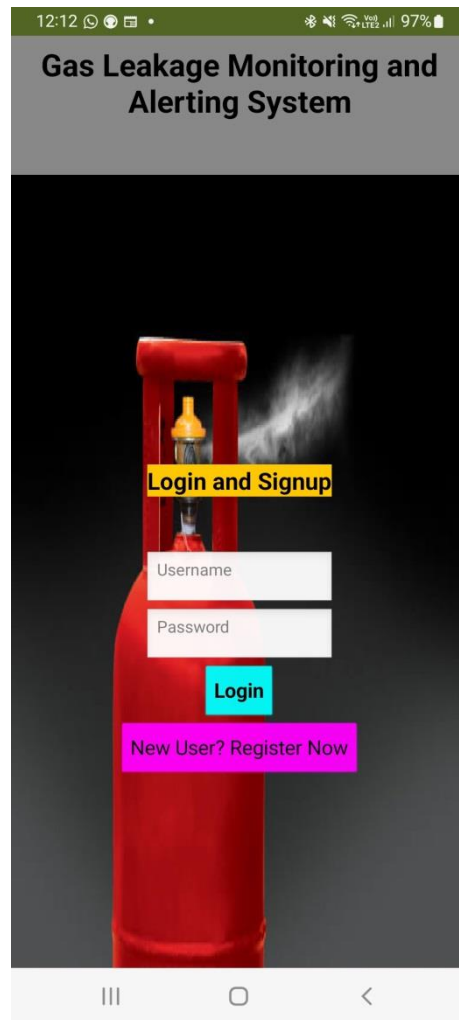
Blocks Section: The "Built-in" category is selected, showing various block types like Control, Logic, Math, Text, Lists, Dictionaries, Colors, Variables, and Procedures. The "Screen3" category is also visible, containing a "VerticalArrangement1" block with sub-components like "TextBox1", "TextBox2", "username", "pw", and "mobno".

Viewer Section: This section shows the logic for the "register" button's click event. The logic is as follows:

- When **register** .Click
- do
- set **Web1** . Url to `join (" https://node-red-rdrxa-2022-11-09.us-east.myblue..."`
- make a list
 - username . Text
 - pw . Text
 - mobno . Text
- set **Label1** . Text to `" Registration Successful! Redirecting to Login.... "`
- set **Clock1** . TimerEnabled to `false`
- call **Web1** .Get
- open another screen screenName **Screen1**

Media Section: This section is currently empty, showing a placeholder for an image.

The interface also includes a top navigation bar with links like "Projects", "Connect", "Build", "Settings", and "Help". A right sidebar contains a "Designer" tab, a "Blocks" tab, and a "Hide Warnings" button.



Using Node-RED the entered details are stored in the Cloudant Database.

The screenshot displays the Node-RED web interface in a browser. The address bar shows the URL: `node-red-rdrxa-2022-11-09.us-east.mybluemix.net/red/#flow/c113cc75604a8191`. The interface includes a left sidebar with node categories (common, function, network), a central workspace with a flow diagram, and a right sidebar with a debug console.

Flow Diagram:

- Flow 1:** Starts with a `[get] /login` node. It branches into two paths:
 - Path 1: `[get] /login` → `Function 1` → `msg.payload` (output).
 - Path 2: `[get] /login` → `Login` node → `Function 2` → `http` node → `msg.payload` (output).
- Flow 3:** Starts with a `[get] /reg` node. It branches into two paths:
 - Path 1: `[get] /reg` → `http` node → `msg.payload` (output).
 - Path 2: `[get] /reg` → `Function 3` → `Registration` node → `msg.payload` (output).

Debug Console:

The debug console shows the following log entries:

- 11/15/2022, 7:33:26 PM node: 91cf557ba6154e80
msg.payload : Object
▶ { User Name: "yukta", Password: "556_!34", Mobile Number: "8895984733" }
- 11/15/2022, 7:35:28 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User Name: "aditya", Password: "tuyedhs", Mobile Number: "6277488282" }
- 11/15/2022, 7:35:29 PM node: 34788d1d77dec690
msg.payload : number
0
- 11/15/2022, 7:36:13 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudarshan", Password: "qwerty-1" }
- 11/15/2022, 7:36:13 PM node: 34788d1d77dec690
msg.payload : number
0
- 11/15/2022, 7:37:05 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudarshan", Password: "qwerty-1" }
- 11/15/2022, 7:37:05 PM node: 34788d1d77dec690
msg.payload : number
1

Function 3 and the change node modify the message payload to store in proper format in the database.

The screenshot shows the Node-RED web interface. The 'Edit function node' panel is open for a node named 'Function 3'. The 'Properties' section shows the name 'Function 3'. The 'Setup' section has 'On Message' selected. The 'Code' section contains the following JavaScript code:

```
1 a=msg.payload.reg //same procedure as function 1
2 s=a.split(",")
3 name = s[0]
4 pass = s[1].trim()
5 mmo = s[2].trim()
6 msg.payload={
7   "User Name": name,
8   "Password": pass,
9   "Mobile Number": mmo,
10 }
11 return msg;
```

The 'debug' console on the right shows the message flow. The first two messages are objects with 'User name' and 'Password'. The third message is a number. The fourth message is an object with 'User name' and 'Password'. The fifth message is a number. The sixth message is an object with 'User name' and 'Password'. The seventh message is a number. The eighth message is an object with 'User name' and 'Password'. The ninth message is a number. The tenth message is an object with 'User name' and 'Password'.

The screenshot shows the Node-RED web interface. The 'Edit change node' panel is open. The 'Properties' section shows the name 'Name'. The 'Rules' section has three rules defined:

- Rule 1: Change `msg.payload.login` to `msg.payload.login`. Search for `^`, Replace with `^`.
- Rule 2: Change `msg.payload.login` to `msg.payload.login`. Search for `^`, Replace with `^`.
- Rule 3: Change `msg.payload.login` to `msg.payload.login`. Search for `^`, Replace with `^`.

The 'debug' console on the right shows the message flow. The first two messages are objects with 'User name' and 'Password'. The third message is a number. The fourth message is an object with 'User name' and 'Password'. The fifth message is a number. The sixth message is an object with 'User name' and 'Password'. The seventh message is a number. The eighth message is an object with 'User name' and 'Password'. The ninth message is a number. The tenth message is an object with 'User name' and 'Password'.

The updated database is shown below:

←

→

↻

🔒 e05d580a-6996-4dc2-85f2-fa1371318e10-bluemix.cloudant.com/dashboard.html#/database/gasdb/_all_docs

🔗

☆

⚙️

🗖

🌐

⋮

📧 ear

📧 [Ticket #0015072]...

📧 ea

🏠 Attractive furnished...

↔️

⏪

gasdb

⋮

Document ID

⌵

⚙️ Options

{ } JSON

📖

🔔

📊 All Documents

+

Query

Permissions

Changes

Design Documents

+

☐

Table

Metadata

{ } JSON

🔍

Create Document

	Mobile Number	Password	User Name	_id
<input type="checkbox"/>	📄 6277488282	tuyedhs	aditya	0b86b26e85d44884f90aac6521...
<input type="checkbox"/>	📄 7166367718	126_4q	Akash	2c42b50a3b6a26ae52d0c951fcb...
<input type="checkbox"/>	📄 8172663729	qwerty-1	Sudarshan	6e6127af8112423ada32ff89a2c0...
<input type="checkbox"/>	📄 8895984733	556_l34	yukta	6e6127af8112423ada32ff89a2d...
<input type="checkbox"/>	📄 7761664478	76hy	ajay	94e11cc75de7deb6c2a200cdb80...

Showing 4 of 5 columns.

☐ Show all columns.

Showing document 1 - 5.

Documents per page:

20

⌵

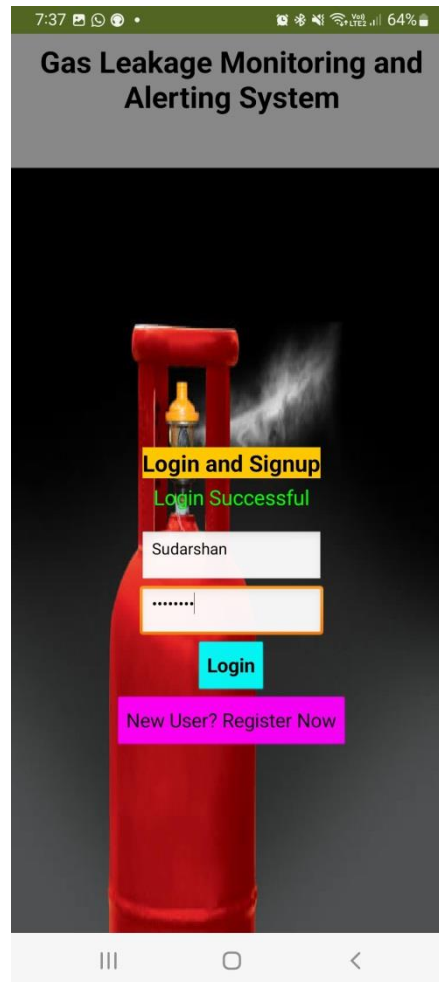
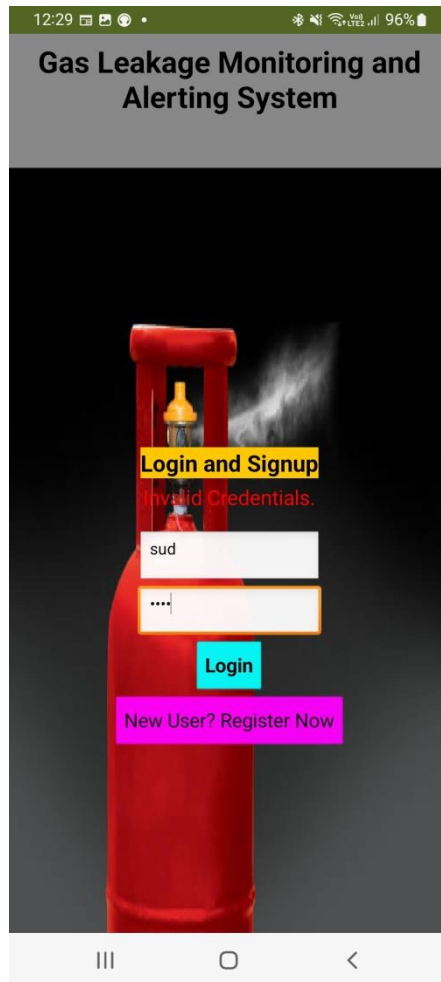
⏪

⏩

Log Out

Login

Once registered, the users can enter their newly created credentials to login. For now, they are redirected to a blank screen after successful login. We will be adding further details in this screen in future sprints. The first image shows invalid credentials entered. If correct credentials are entered, then the login is successful.



For user login we used the following blocks in MIT App Inventor 2

The screenshot displays the MIT App Inventor 2 web interface. The browser address bar shows the URL `ai2.appinventor.mit.edu/#5418205422944256`. The project name is **Gas_leakage**. The interface is divided into three main sections: **Blocks**, **Viewer**, and **Media**.

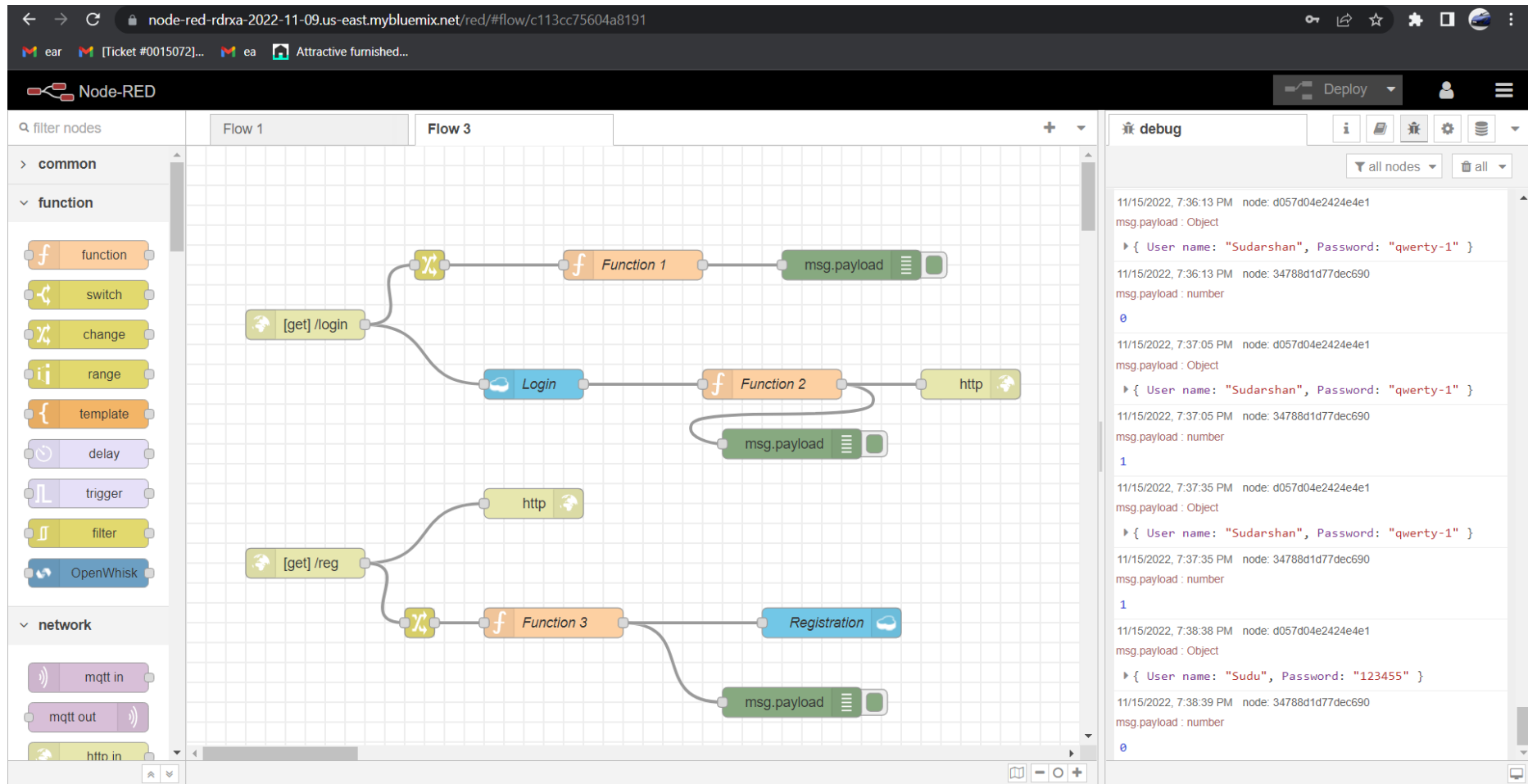
Blocks Panel: The left sidebar shows the 'Built-in' blocks category. Under 'Screen1', the following components are listed: `TextBox5`, `VerticalArrangement1`, `Label3`, `Label1`, `Label2`, and `TextBox3`. Buttons for 'Rename' and 'Delete' are visible at the bottom of the list.

Viewer Panel: The central workspace shows the visual representation of the app screen. It includes a teal backpack icon in the top right corner and a vertical arrangement of three circular buttons (with a target icon, a plus sign, and a minus sign) and a trash can icon on the right side. At the bottom left of the viewer, there are warning icons (a yellow triangle and a red X) both showing a count of 0, and a 'Hide Warnings' button.

Code Blocks: The right side of the viewer shows the event-driven code blocks:

- when Screen1.Initialize**
 - do `set TextBox5.MultiLine to true`
- when signupbtn.Click**
 - do `open another screen screenName Screen3`
- when loginbtn.Click**
 - do `set Web1.Url to join ["https://node-red-rdrxa-2022-11-09-us-east.myblue...", ...]`
 - `make a list` containing `TextBox3.Text` and `PasswordTextBox1.Text`
 - `call Web1.Get`
- when Web1.GotText**
 - `url`: `responseCode`, `responseType`, `responseContent`
 - do `if call Web1.JsonTextDecode jsonText get responseContent == decimal 1`
 - then
 - `set Label2.TextColor to green`
 - `set Label2.Text to "Login Successful"`
 - `open another screen screenName Screen2`
 - else
 - `set Label2.Text to "Invalid Credentials"`

The data entered during login is then sent to Node-RED where it is compared with the database stored in the cloud. If the credentials match, 1 is returned by Node-RED else 0 is returned.



The functions used are shown below. Function 1 declares the received credentials as global so that it can be accessed by Function 2. Function 2 performs the comparison.

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow with two function nodes. The first function node is named 'Function 1' and is configured with the following JavaScript code:

```
1 a=msg.payload.login
2 s = a.split(",")
3 user = s[0]
4 pass = s[1].trim()
5 msg.payload={
6   "User name":user,
7   "Password":pass // convert to json format ,store in msg.payload
8 }
9 global.set('u',user) //the above things are used for checking in DB
10 global.set('p',pass)
11 global.set('m',msg.payload)
12 return msg;
```

The second function node is not shown in the screenshot. The debug console on the right shows the following messages:

```
11/15/2022, 7:36:13 PM node: d057d04e2424e4e1
msg.payload : Object
{ User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:36:13 PM node: 34788d1d77dec690
msg.payload : number
0

11/15/2022, 7:37:05 PM node: d057d04e2424e4e1
msg.payload : Object
{ User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:37:05 PM node: 34788d1d77dec690
msg.payload : number
1

11/15/2022, 7:37:35 PM node: d057d04e2424e4e1
msg.payload : Object
{ User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:37:35 PM node: 34788d1d77dec690
msg.payload : number
1

11/15/2022, 7:38:38 PM node: d057d04e2424e4e1
msg.payload : Object
{ User name: "Sudu", Password: "123455" }

11/15/2022, 7:38:39 PM node: 34788d1d77dec690
msg.payload : number
0
```

node-red-rdxa-2022-11-09.us-east.mybluemix.net/red/#flow/c113cc75604a8191

ear [Ticket #0015072]... ea Attractive furnished...

Node-RED

Deploy

filter nodes

Flow 1

Flow 2

common

function

function

switch

change

range

template

delay

trigger

filter

OpenWhisk

network

mqtt in

mqtt out

http in

[get] /login

[get] /reg

Edit function node

Delete Cancel Done

Properties

Name Function 2

Setup On Start On Message On Stop

```
1 var a=msg.payload
2 var e=global.get('u')
3 var p=global.get('p')
4 count=0
5 for (j in a){
6   count=count+1
7 }
8 flag=0
9 for (i=0;i<count;i++){
10   if((e==a[i]["User Name"])&(p==a[i]["Password"]))
11   {
12     flag = 1//if flag =1, credential availblle in DB ,send to App.
13   }
14 }
15 if (flag==1){
16   msg.payload = 1
17 }
18 if(flag==0){
19   msg.payload=0
20 }
21 return msg;
```

Enabled

debug

all nodes all

11/15/2022, 7:36:13 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:36:13 PM node: 34788d1d77dec690
msg.payload : number
0

11/15/2022, 7:37:05 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:37:05 PM node: 34788d1d77dec690
msg.payload : number
1

11/15/2022, 7:37:35 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudarshan", Password: "qwerty-1" }

11/15/2022, 7:37:35 PM node: 34788d1d77dec690
msg.payload : number
1

11/15/2022, 7:38:38 PM node: d057d04e2424e4e1
msg.payload : Object
▶ { User name: "Sudu", Password: "123455" }

11/15/2022, 7:38:39 PM node: 34788d1d77dec690
msg.payload : number
0