SCHOOL OF DIGITAL MEDIA AND INFOCOMM TECHNOLOGY (DMIT)

IOT CA2
Step-by-step Tutorial

DIPLOMA IN BUSINESS INFORMATION TECHNOLOGY
DIPLOMA IN INFORMATION TECHNOLOGY
DIPLOMA IN INFOCOMM SECURITY MANAGEMENT

ST0324 Internet of Things (IOT)

Date of Submission:

Prepared for: Ms Dora Chua

Class:

Submitted by: Leyond Lee

Student ID	**Name**
	Leyond Lee
	Titus Fung
	Daryl Loh

Table of Contents

Section 1
Overview of project

A. Where we have uploaded our tutorial

-

B.Why have we chosen to upload to this site

-

C. What have we uploaded

-

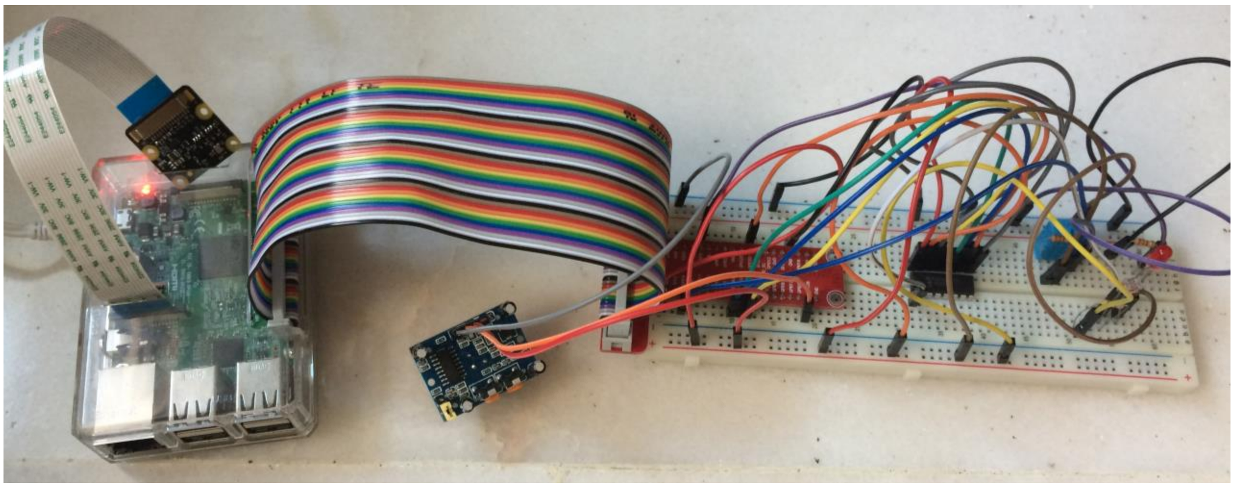
D. What is the application about?

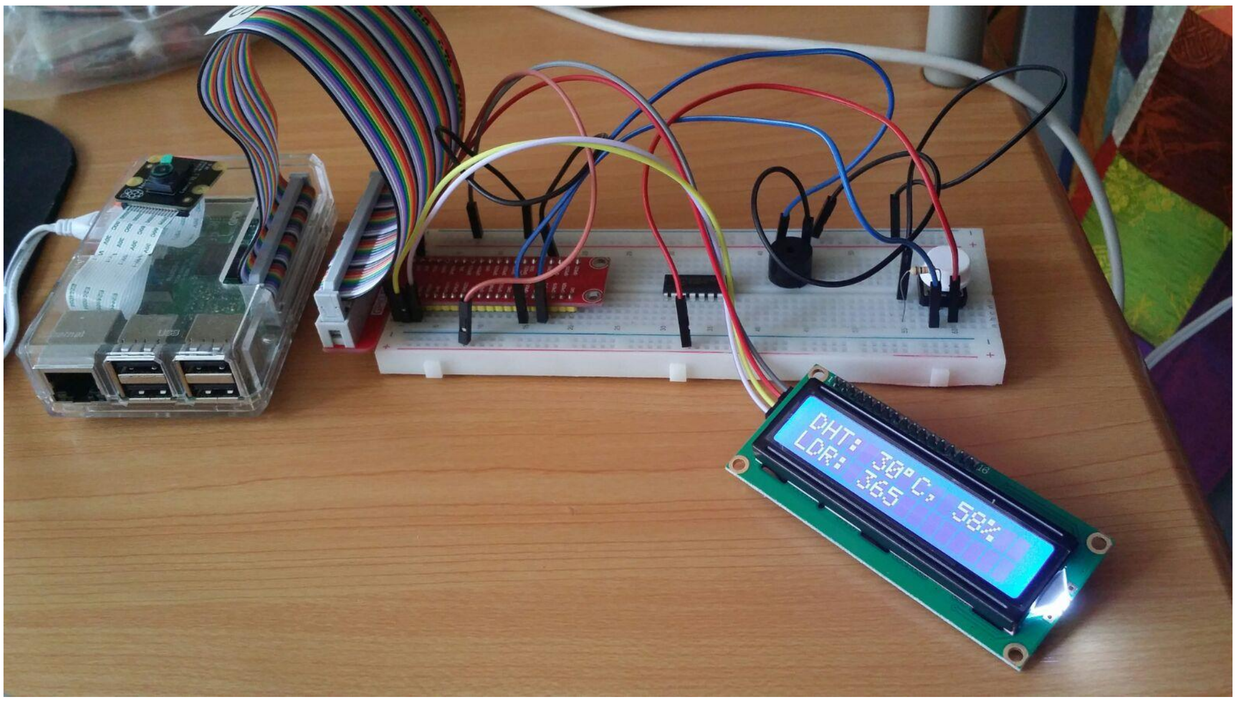
-

E. Summary of the steps that will be described

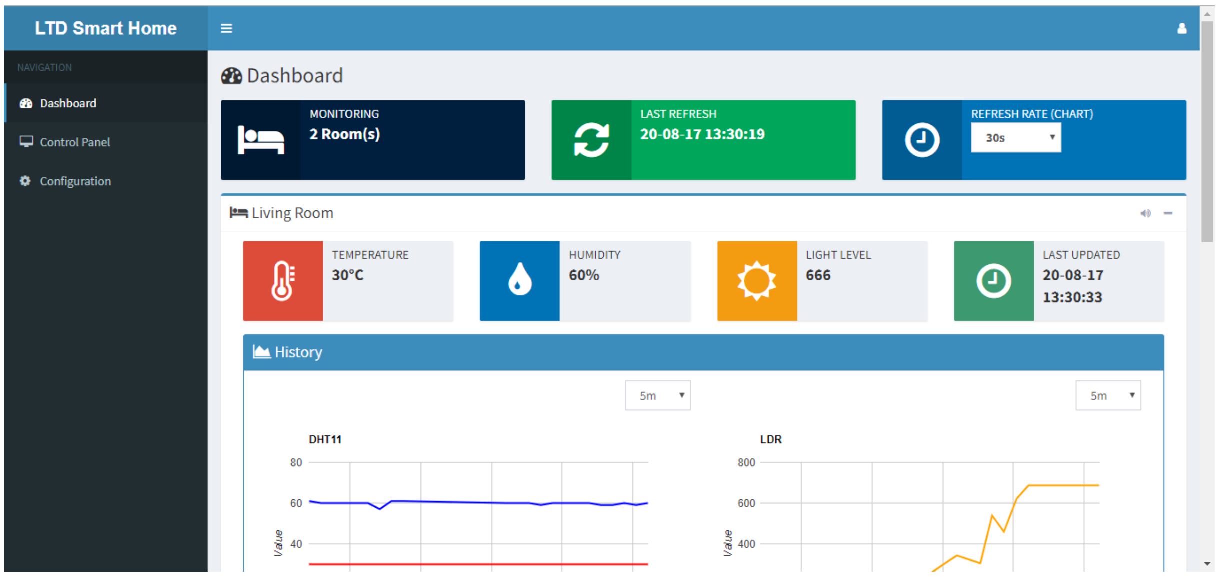
	Section	Description
1)	Overview	Provides overview of the application
Sections 2 to 5 provides the step-by-step instructions to set up the application		
2)	Prerequisites	Provides overview of hardware and software requirements
3)	AWS and certificates setup	Provides instructions on how to setup AWS IoT, Lambda, IAM as well as certificates
4)	Server setup	Provides instructions on how to setup server
5)	RaspberryPi setup	Provides instructions on how to setup the RaspberryPi based on various configurations
6)	Additional information	Provides additional information about the application

F. How does the final RPI set-up looks like?





G. How does the web or mobile application look like?



MONITORING
2 Room(s)

LAST REFRESH
20-08-17 13:30:19

REFRESH RATE (CHART)
30s

Living Room

TEMPERATURE
30°C

HUMIDITY
60%

LIGHT LEVEL
666

LAST UPDATED
20-08-17 13:30:33

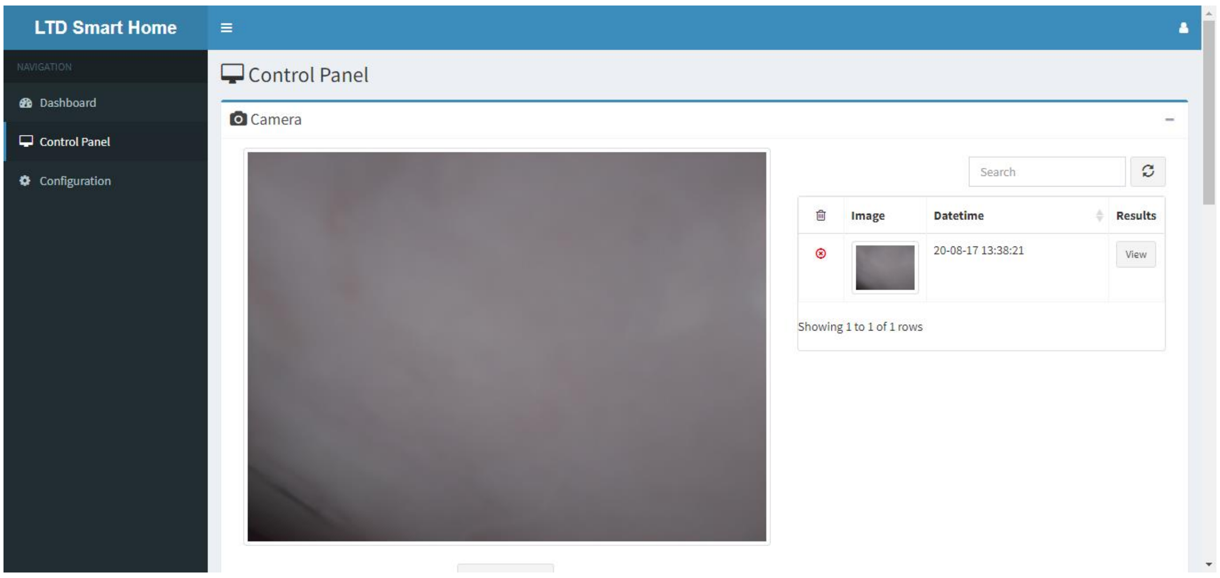
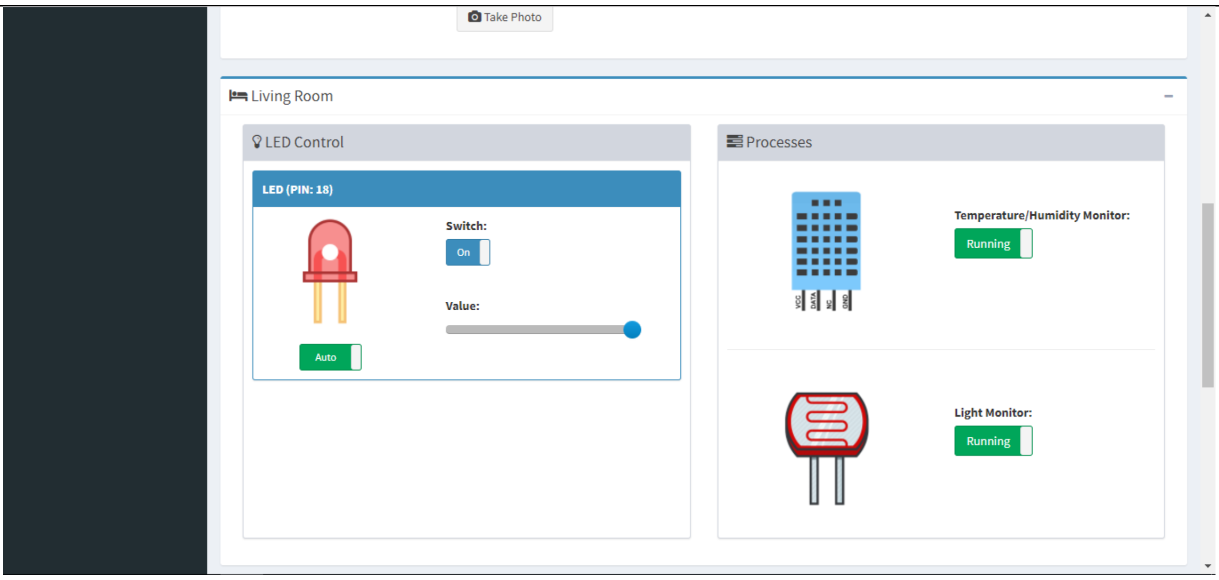
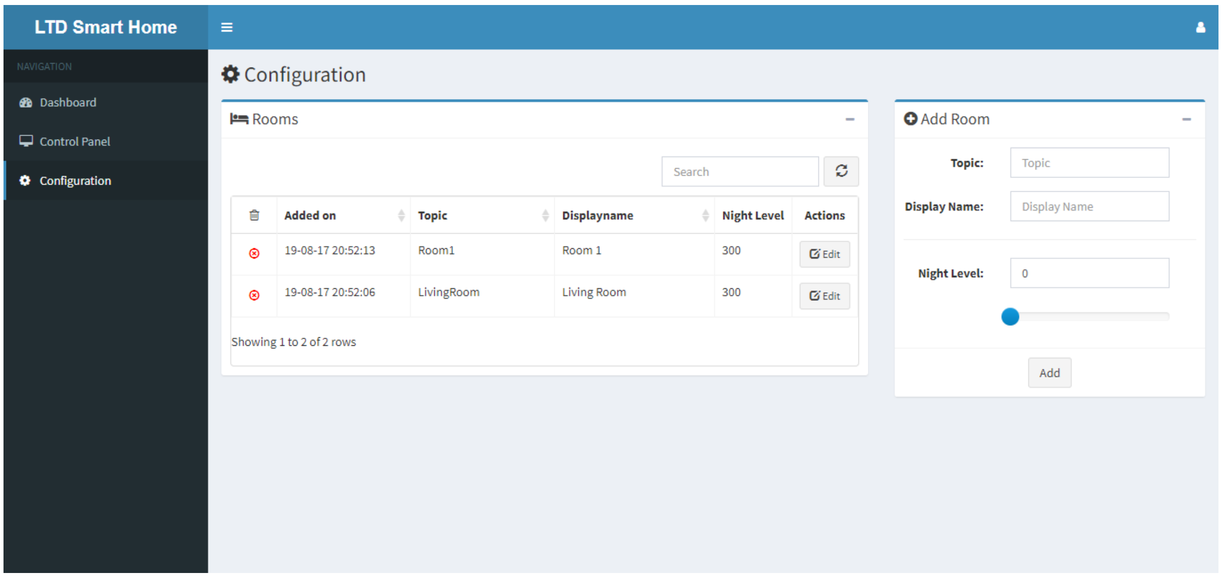


	Image	Datetime	Results
		20-08-17 13:38:21	View

Showing 1 to 1 of 1 rows

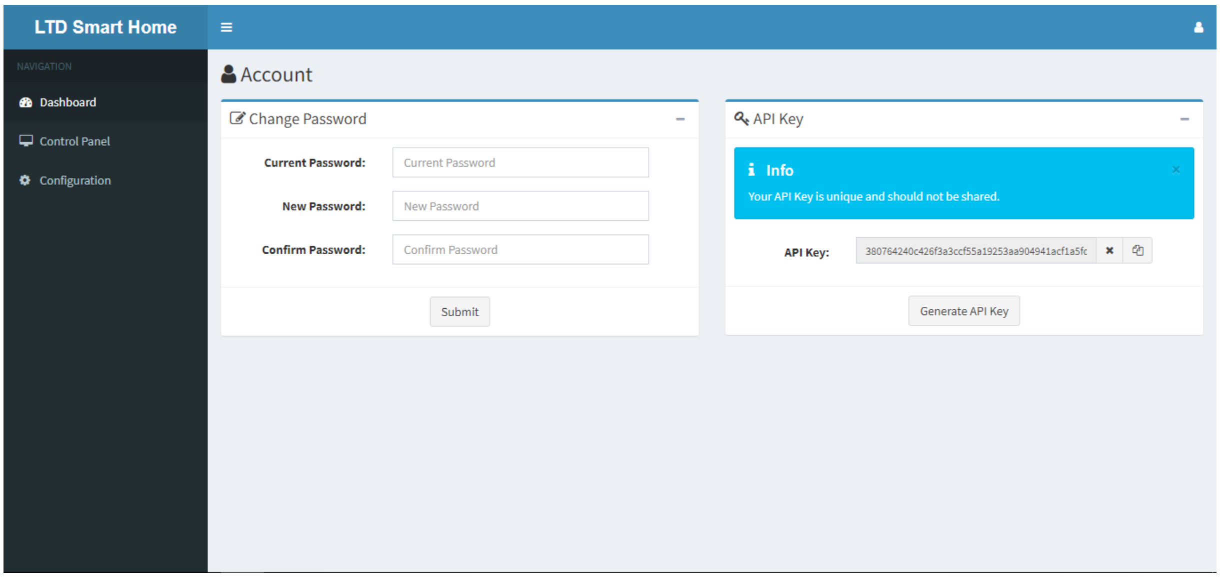




	Added on	Topic	Displayname	Night Level	Actions
⊗	19-08-17 20:52:13	Room1	Room 1	300	Edit
⊗	19-08-17 20:52:06	LivingRoom	Living Room	300	Edit

Showing 1 to 2 of 2 rows

Topic:	Topic
Display Name:	Display Name
Night Level:	0



Current Password:	Current Password
New Password:	New Password
Confirm Password:	Confirm Password

Info

Your API Key is unique and should not be shared.

API Key: 380764240c426f3a3ccf55a19253aa904941acf1a5fc

Section 2
Prerequisites

A. Checklist

Main server	
Hardware	Any (AWS EC2, Virtual Machine, your laptop, Raspberry Pi, etc)
Operating System	Any (Preferably Ubuntu Server)
Applications	python python-pip MySQL (optional if using other hosting solutions such as AWS RDS)
Python Libraries	virtualenv **To be installed in virtual environment:** - gevent - flask - flask-sqlalchemy - flask-wtf - flask-login - watson_developer_cloud - MySQL-python - AWSIoTPythonSDK - boto3 - python-dateutil - flask-socketio

Camera server	
Hardware	Any (AWS EC2, Virtual Machine, your laptop, Raspberry Pi, etc)
Operating System	Any (Preferably Ubuntu Server)
Applications	python python-pip
Python Libraries	virtualenv **To be installed in virtual environment:** - tornado - pillow - requests

Camera client	
Hardware	Raspberry Pi 3 **Additional Hardware:**

Quantity	**Type**
1	PiCamera

Operating System	Raspbian
Applications	python python-pip
Python Libraries	picamera

Sensor	
Hardware	Raspberry Pi 3 **Additional Hardware:**

Quantity	**Type**
1	Breadboard (with T-Cobbler)
1	MCP3008
1	Light Dependent Resistor (LDR) Sensor
1	DHT11 Sensor
1	PIR Motion Sensor
1	10kΩ Resistor
~2	220Ω Resistor
~ 30	Jumper cables/wires

Operating System	Raspbian
Applications	python python-pip
Python Libraries	AWSIoTPythonSDK gpiozero Adafruit_Python_DHT

Doorbell	
Hardware	Raspberry Pi 3 **Additional Hardware:** **Quantity** – **Type** 1 – Breadboard (with T-Cobbler) 1 – Buzzer 1 – Button 1 – 220Ω Resistor ~20 – Jumper cables/wires
Operating System	Raspbian
Applications	python python-pip
Python Libraries	AWSIoTPythonSDK gpiozero

Additional Hardware (Doorbell):

Quantity	Type
1	Breadboard (with T-Cobbler)
1	Buzzer
1	Button
1	220Ω Resistor
~20	Jumper cables/wires

LCD	
Hardware	Raspberry Pi 3 **Additional Hardware:**
Operating System	Raspbian
Applications	python python-pip
Python Libraries	AWSIoTPythonSDK rpi-lcd

Additional Hardware (LCD):

Quantity	Type
1	Breadboard (with T-Cobbler)
1	IC2 LCD Screen
4	Jumper cables/wires

Section 3
AWS and certificates setup

A. AWS IoT

TASK

1. Login to AWS and navigate to AWS IoT console.

2. Register a device
   a. On the navigation panel, click **Registry** to expand the choices, and then choose **Things**.
   b. Click **Register a thing** or **Create**.
   c. Enter a name and click **Create thing**.

3. Create certificates
   a. On the navigation panel, click **Security**, and then click **Certificates**.
   b. Click **Create**.
   c. Click **Create certificate** for "One-click certificate creation (recommended)".
   d. Once the certificate is created, **Download** them (including RootCA), then click **Activate**.
   e. Click **Attach a policy** then **Create new policy**.
   f. For name, enter "Webapp".
   g. Under "Add statements" switch the mode to **Advanced mode** and enter the following:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": "iot:*",
   "Resource": "*"
  }
 ]
}
```

   h. Go back to Certificates and attach the policy.
   i. Next, repeat the certification creation steps and create a new policy.
   j. For name, enter "RPi".
   k. Under "Add statements" switch the mode to **Advanced mode** and enter the following:
      Note – Change <region> and <account ID>

```
{
        "Version": "2012-10-17",
        "Statement": [
    {
                    "Effect": "Allow",
                    "Action": "iot:Connect",
                    "Resource": "*"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Publish",
                    "Resource": "arn:aws:iot:<region>:<account ID>:topic/room/*"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Publish",
                    "Resource": "arn:aws:iot: <region>:<account ID>:topic/doorbell"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Subscribe",
                    "Resource": "arn:aws:iot: <region>:<account
                    ID>:topicfilter/room/*"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Receive",
                    "Resource": "arn:aws:iot: <region>:<account
                    ID>:topic/room/*/controls/led/*"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Receive",
                    "Resource": "arn:aws:iot: <region>:<account
                    ID>:topic/room/*/controls/process/*"
    },
    {
                    "Effect": "Allow",
                    "Action": "iot:Receive",
                    "Resource": "arn:aws:iot: <region>:<account
                    ID>:topic/room/*/sensors"
    }
    ]
 }
```

l. Go back to Certificates and attach the policy.

4. Place the certificates in the correct folders.
   Note – Configure settings.ini
   Certificate with Webapp policy

- server/aws

Certificate with RPi policy
- doorbell/aws
- lcd/aws
- sensor/aws

B. AWS Lambda

1. Login to AWS and navigate to AWS Lambda.

2. In AWS Lambda Dashboard page, click **Create function**.

3. For blueprint, click **Author from scratch**.

4. For trigger, click **Next**.

5. Under Basic information:
   a. Name – SensorData
   b. Runtime – Node.js 6.10

6. Under Lambda function code, enter the following:

```
var AWS = require('aws-sdk');

AWS.config.update({
  region: 'ap-southeast-1'
});

var docClient = new AWS.DynamoDB.DocumentClient();

exports.handler = (event, context, callback) => {
  var room = event.Room;
  var timestamp = event.Timestamp;
  var dht11 = event.DHT11;
  var ldr = event.LDR;

  var humidity = dht11.Humidity;
  var temperature = dht11.Temperature;
  var light = ldr.Value;

  docClient.get({
    TableName: 'Subscription',
    Key: {
      'Topic': room
```

```
        }
    }, function(err, data) {
        if (!err) {
            if ('Item' in data) {
                if (light) {
                    light = Math.round((1024 / light) % 1024);
                }

                docClient.put({
                    TableName: 'Sensor',
                    Item: {
                        'Room': room,
                        'Timestamp': timestamp,
                        'Humidity': humidity,
                        'Temperature': temperature,
                        'Light': light
                    }
                }, function(err, data) {});
            }
        }
    });
};
```

7. Follow through till the end.

8. Navigate to AWS IoT console.

9. Click **Rules**, then click **Create**.

10. Enter the following:
    a. Name - SensorRule
    b. Attribute - topic(2) as Room, Timestamp, DHT11, LDR
    c. Topic filter - room/+/sensors

11. Then click **Add action** and select **Invoke a Lambda function passing the message data**.

12. Click **Configure action** and choose the function "SensorData".

13. Click **Add action** and then **Create rule**.

C. AWS IAM

TASK
<u>**Lambda Role**</u>

1. Login to AWS and navigate to AWS IAM console.

2. On the navigation panel on the left, click on **Policies**.

3. Then **Create policy**.

4. Click **Select**, for “Create Your Own Policy” and enter the following:
    a. Policy Name - Sensor_Lambda
    b. Policy Document
    Note – Change <region> and <account ID>

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1503053280000",
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:<region>:<account ID>:table/Sensor"
      ]
    },
    {
      "Sid": "Stmt1503053309000",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:<region>:<account ID>:table/Subscription"
      ]
    }
  ]
}
```

5. Click **Create policy**.

6. Next, on the navigation panel on the left, click on **Roles**.

7. Click on the role associated to the “SensorData” Lambda function.

8. Click **Attach Policy** and select “Sensor_Lambda”.

Webapp user

1. On the navigation panel on the left, click on **Users**.

2. Click **Add user**.

3. Enter the following:
    a. Name - DynamoDB_Webapp
    b. Access type - Programmatic access

4. Click **Next: Permissions**.

5. Select **Attach existing policies directly**, then click **Create policy**.

6. Click **Select**, for “Create Your Own Policy” and enter the following:
    a. Policy Name - Webapp_policy
    b. Policy Document
    Note – Change <region> and <account ID>

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Stmt1502635912000",
         "Effect": "Allow",
         "Action": [
            "dynamodb:CreateTable",
            "dynamodb:DeleteItem",
            "dynamodb:DeleteTable",
            "dynamodb:Query"
         ],
         "Resource": [
            "arn:aws:dynamodb:<region>:<account ID>:table/Sensor"
         ]
      },
      {
         "Sid": "Stmt1502636185000",
         "Effect": "Allow",
         "Action": [
            "dynamodb:CreateTable",
            "dynamodb:DeleteItem",
            "dynamodb:DeleteTable",
            "dynamodb:PutItem",
            "dynamodb:Query",
            "dynamodb:Scan",
            "dynamodb:UpdateItem"
         ],
         "Resource": [
            "arn:aws:dynamodb:<region>:<account ID>:table/Subscription"
         ]
      }
   ]
}
```

7. Click **Create policy**, then click **Next: Review**.

8. Click **Create user**.

9. Note down the “Access key ID” and “Secret access key” as it is needed in the server setup later.

D. Server and Client certificates

TASK

1. Download KeyStore Explorer (http://keystore-explorer.org/downloads.html)

2. Open KeyStore Explorer

3. Click **Create a new KeyStore**.
   a. Type – PKCS #12

4. Click **Tools** -> **Generate Key Pair**.
   a. Use the default values until “Name”.
   b. For “Name”, specify your details.
   c. Click **OK**.
   d. For alias, use “rootca”.

5. In the main keystore page, right click on “rootca”. **Sign** -> **Sign New Key Pair**.
   a. Follow the same steps as when creating “rootca”. **Note: Do not use same common name**
   b. For alias, use “server”.

6. Repeat step 5 to create a certificate for “server”.
   a. For alias, use “client”.

7. Export all certificates and private keys (excluding rootca).
   a. Right click -> **Export** -> **Export Certificate Chain**.
      i. Use the default values
   b. Right click -> **Export** -> **Export Private Key**.
      i. Private Key Type – OpenSSL
      ii. Encrypt
      iii. Encryption Algorithm **PBE with 256 bit AES CBC**
   c. (Recommended) Renamed the certificate and private key accordingly:
      i. <type>_certificate.cer
      ii. <type>_private.key

8. Place the certificates in the correct folders.
   Note – Configure settings.ini
   <u>server/certs</u>
   - rootca (certificate)
   - server (certificate and private key)
   - client (certificate and private key)

   <u>camera_server/certs</u>
   - rootca (certificate)
   - server (certificate and private key)

   <u>camera_client/certs</u>
   - rootca (certificate)
   - client (certificate and private key)

Section 4
Server setup

A. MySQL Database

TASK

METHOD 1: Host locally on server

1. Install MySQL-Server

```
sudo apt-get update
sudo apt-get install mysql-server
sudo mysql_secure_installation
```

2. Move on to **Configure Database**.

METHOD 2: AWS RDS

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.

2. In the top right corner of the Amazon RDS console, choose the region in which you want to create the DB instance.

3. In the navigation pane, choose **Instances**.

4. Choose **Launch DB Instance**. The Launch DB Instance Wizard opens on the Select Engine page.

5. Choose **MySQL** and click **Select**.

6. Configure the instance and follow through till the end.

7. Move on to **Configure Database**.

Configure Database

1. Connect to MySQL-Server

```
mysql -h <endpoint> -P <port> -u root -p
```

2. Create database

```
mysql> CREATE DATABASE assignment;
```

3. Create user for database

```
mysql> USE assignment;
mysql> CREATE USER ‘assignmentuser’@’localhost’ IDENTIFIED by ‘<NEW
PASSWORD>’;
mysql> GRANT ALL PRIVILEGES ON assignment.* TO ‘assignmentuser’@’localhost’;
mysql> FLUSH PRIVILEGES;
```

4. Exit MySQL console

```
mysql> quit
```

B. Setup files

TASK

METHOD 1: RaspberryPi

1. Access the RaspberryPi.
2. Move on to **Configure Files**.

METHOD 2: AWS EC2

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the console dashboard, click **Launch Instance**.
3. Choose your desired image (preferably "Ubuntu Server").
4. Follow through till the end.
5. Access the instance through SSH.
6. Move on to **Configure Files**.

Configure files

1. On the server, create a folder named "LTD_SmartHome".

```
mkdir -p ~/LTD_SmartHome
```

2. From the zip file, copy the folders "server" and "camera_server" (located within "source") into the "LTD_SmartHome" folder.

3. Configure settings.ini for servers

server

Class	Setting	Type	Description
Main	debug	BOOLEAN	Debug mode
Server	server_port	INTEGER	Flask server port
	rootcapath	STRING	RootCA certificate path
	certificatepath	STRING	Server certificate path
	privatekeypath	STRING	Server private key path
	privatekeypassword	STRING	Server private key password
Database	db_host	STRING	Database address
	db_port	INTEGER	Database port
	db_user	STRING	Database user
	db_password	STRING	Database password
	db_database	STRING	Database to use

AWS	aws_iot_endpoint	STRING	AWS IoT endpoint address
	aws_iot_rootcapath	STRING	AWS IoT RootCA path
	aws_iot_certificatepath	STRING	AWS IoT certificate path
	aws_iot_privatekeypath	STRING	AWS IoT private key path
	aws_dynamodb_access_key_id	STRING	AWS IAM access key (for dynamodb)
	aws_dynamodb_secret_access_key	STRING	AWS IAM secret access key (for dynamodb)
	aws_dynamodb_region_name	STRING	AWS Region
IBM	ibm_visualrecognition_apikey	STRING	IBM API key (For Visual Recognition)
	ibm_texttospeech_username	STRING	IBM TextToSpeech username
	ibm_texttospeech_password	STRING	IBM TextToSpeech password
Camera	stream_host	STRING	Camera server address
	stream_rootcapath	STRING	RootCA certificate path
	stream_certificatepath	STRING	Client certificate path
	stream_privatekeypath	STRING	Client private key path
	stream_privatekeypassword	STRING	Client private key password

camera_server

Class	Setting	Type	Description
Server	server_port	INTEGER	Tornado server port
	main_host	STRING	Main server address
	rootcapath	STRING	RootCA certificate path
	certificatepath	STRING	Server certificate path
	privatekeypath	STRING	Server private key path
	privatekeypassword	STRING	Server private key password
Camera	stream_port	INTEGER	Stream server port
	stream_rootcapath	STRING	RootCA certificate path
	stream_certificatepath	STRING	Server certificate path
	stream_privatekeypath	STRING	Server private key path
	stream_privatekeypassword	STRING	Server private key password

C. Run servers

TASK

1. Create virtual environments
   a. server
      ```
      cd ~/LTD_SmartHome/server
      virtualenv env
      env/bin/pip install -r requirements.txt
      ```
   b. camera_server
      ```
      cd ~/LTD_SmartHome/camera_server
      virtualenv env
      env/bin/pip install -r requirements.txt
      ```
2. Start servers
   a. server
      ```
      cd ~/LTD_SmartHome/server
      nohup env/bin/python run.py &
      ```
   b. camera_server
      ```
      cd ~/LTD_SmartHome/camera_server
      nohup env/bin/python run.py &
      ```
3. Additional information
   a. Check processes
      ```
      ps aux | grep python
      ```
   b. Kill process
      ```
      kill -9 <pid>
      ```
4. Access servers
   a. server
      https://<server ip>:<server_port in "server">
   b. camera_server
      https://<server ip>:<server_port in "camera_server">

Section 5
RaspberryPi setup

A. Camera client

TASK

1. Create a folder named “LTD_SmartHome”.

```
mkdir ~/LTD_SmartHome
```

2. Copy the folder “camera_client” (located within “source”) into “LTD_SmartHome” folder.

3. Install python packages

```
cd ~/LTD_SmartHome/camera_client
pip install -r requirements.txt
```

4. Configure settings.ini

Class	Setting	Type	Description
Camera	stream_host	STRING	Stream server address
	stream_port	INTEGER	Stream server port
	stream_rootcapath	STRING	RootCA certificate path
	stream_certificatepath	STRING	Client certificate path
	stream_privatekeypath	STRING	Client private key path
	stream_privatekeypassword	STRING	Client private key password

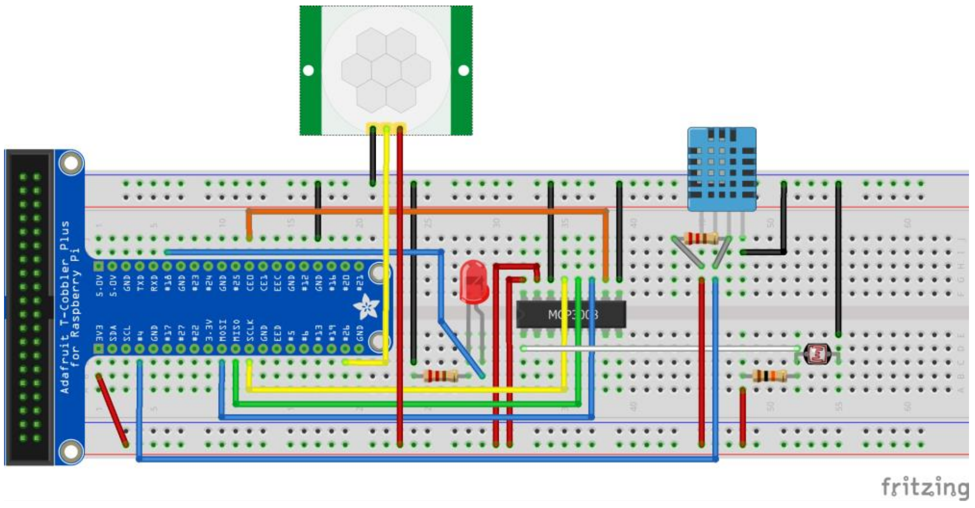
5. Start camera client script

```
cd ~/LTD_SmartHome /camera_client
python run.py
```

B. Sensor

TASK

1. Complete the Fritzing Diagram.



2. Create a folder named “LTD_SmartHome”.

```
mkdir ~/LTD_SmartHome
```

3. Copy the folder “sensor” (located within “source”) into “LTD_SmartHome” folder.

4. Install python packages

```
cd ~/LTD_SmartHome/sensor
pip install -r requirements.txt
```

5. Configure settings.ini

Class	Setting	Type	Description
Main	topic	STRING	MQTT Topic
	dht11_pin	INTEGER	DHT11 Sensor pin
	motionsensor_pin	INTEGER	PIR Motion Sensor pin
	ldr_channel	INTEGER	LDR channel on ADC
	led_pins	STRING	LED pins (Seperated by ,)
AWS	aws _endpoint	STRING	AWS IoT endpoint address
	aws _rootcapath	STRING	AWS IoT RootCA path
	aws _certificatepath	STRING	AWS IoT certificate path
	aws _privatekeypath	STRING	AWS IoT private key path

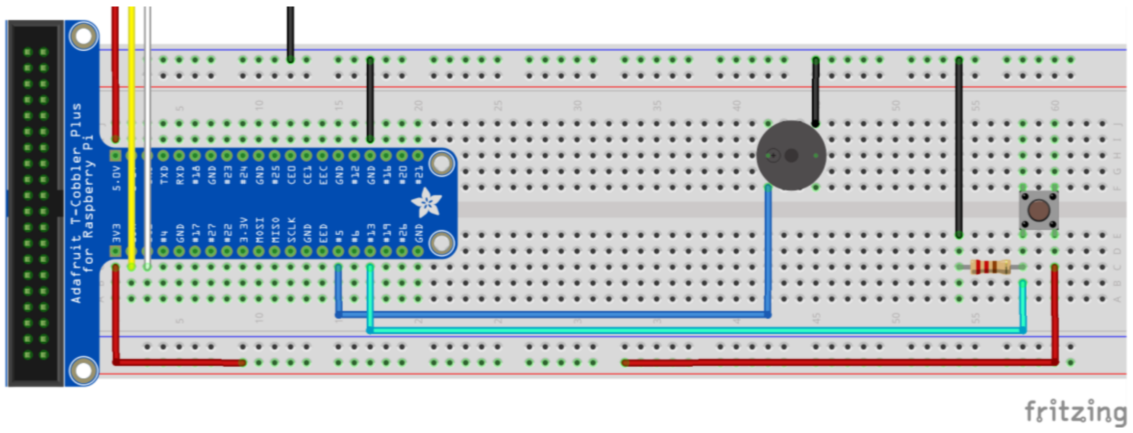
6. Start sensor script

```
cd ~/LTD_SmartHome/sensor
python run.py
```

C. Doorbell

TASK

1. Complete the Fritzing Diagram.



2. Create a folder named “LTD_SmartHome”.

```
mkdir ~/LTD_SmartHome
```

3. Copy the folder “doorbell” (located within “source”) into “LTD_SmartHome” folder.

4. Install python packages

```
cd ~/LTD_SmartHome/doorbell
pip install -r requirements.txt
```

5. Configure settings.ini

Class	Setting	Type	Description
Main	buzzer_pin	INTEGER	Buzzer pin
	button_pin	INTEGER	Button pin
AWS	aws _endpoint	STRING	AWS IoT endpoint address
	aws _rootcapath	STRING	AWS IoT RootCA path
	aws _certificatepath	STRING	AWS IoT certificate path
	aws _privatekeypath	STRING	AWS IoT private key path

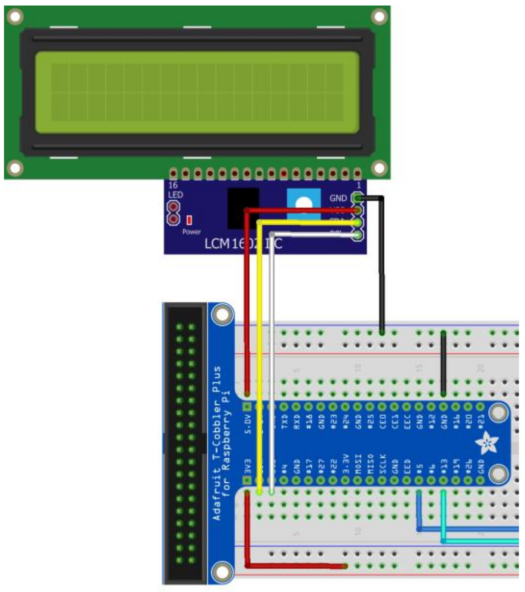
6. Start doorbell script

```
cd ~/LTD_SmartHome/doorbell
python run.py
```

D. LCD

TASK

1. Complete the Fritzing Diagram.



2. Create a folder named “LTD_SmartHome”.

```
mkdir ~/LTD_SmartHome
```

3. Copy the folder “lcd” (located within “source”) into “LTD_SmartHome” folder.

4. Install python packages

```
cd ~/LTD_SmartHome /lcd
pip install -r requirements.txt
```

5. Configure settings.ini

Class	**Setting**	**Type**	**Description**
Main	topic	STRING	MQTT Topic
AWS	aws _endpoint	STRING	AWS IoT endpoint address
	aws _rootcapath	STRING	AWS IoT RootCA path
	aws _certificatepath	STRING	AWS IoT certificate path
	aws _privatekeypath	STRING	AWS IoT private key path

6. Start doorbell script

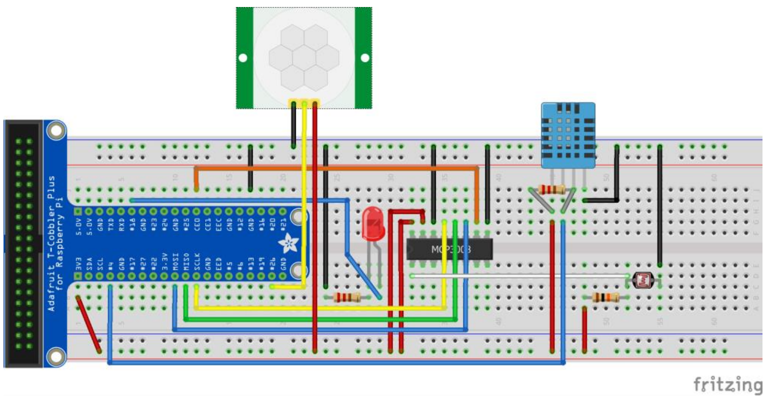
```
cd ~/LTD_SmartHome/lcd
python run.py
```

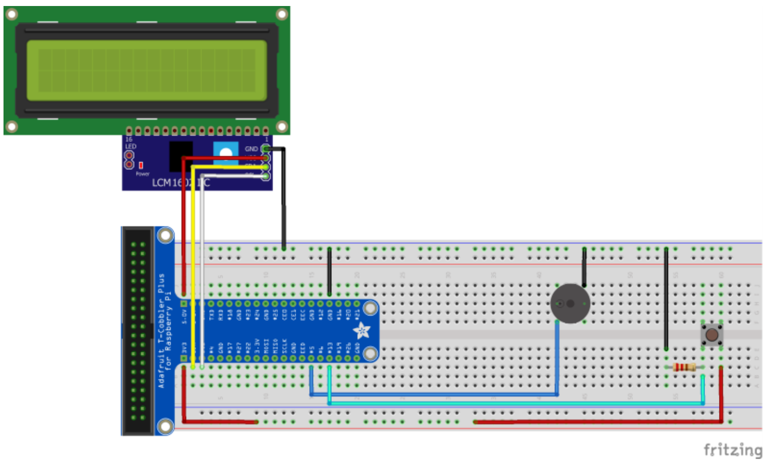
Section 6
Additional information

A. Server "manage.py"

TASK
1. Access the main server 2. cd ~/LTD_SmartHome/server 3. env/bin/python manage.py a. Options: i. (1) Create User ii. (2) Delete User iii. (3) List Users iv. (4) Delete Sensor Table v. (5) Delete Subscription Table vi. (q) Exit

B. Recommended Setup

Host	Type
Ubuntu Server	Main server Camera server
RaspberryPi – Sensor (at least 1)	Sensor Camera client (only 1 required) 

RaspberryPi	Doorbell LCD 

-- End of CA2 Step-by-step tutorial --