VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

System Access Control Using Smart Fencing at Stockyards

By

Kulvir Singh (19BCE2074)
Aryan Bhatt (19BCE2208)

Project Report
on
CSE3009 – Internet of Things

Fall Semester 2021-22

Submitted to
Faculty: Dr. Deepa K
Date: 10-12-2021
Slot : B2+TB2

CONTENT	PAGE No.

1.Abstract

This project work is made for Stockyards where there is constant risk of injury and fatality to the people working on field. The project deals with an IOT device which gets triggered when the magnetic field of a magnet comes in contact with a magnetic switch. A smart fence is made for the people to cross into and work in the danger prone area. When a person crosses the fence, a signal is triggered by the IOT device to the central access control system which disables the machinery and rail cables. The connection is re-established only when the fence is closed by the person who crossed it earlier.

Therefore, the project comprises of a smart fence which can access the system and control the machines accordingly in order to prevent human fatality.

Keywords: Internet of Things. Wireless Sensor Network, Raspberry PI

2.Introduction

Theoretical Background

An occupancy sensor is a piece of electronic equipment that detects the presence of people. Occupancy sensing technologies include magnetic, microwave, ultrasonic, and video image processing. The sensors are typically linked to a building's Internet of Things (IoT) network and feed data back to building management systems and booking systems, which can automate systems for lighting, HVAC, and ventilation control, as well as provide data for occupancy analytics systems to understand desk usage, meeting room efficiency, and space utilisation.

Aim of the Proposed Work

In this project we will be making a smart fence that will be coded in Universal Windows Platform which will be installed in the Raspberry PI.3. There will be a web application interface which will act as a frontend and the full application will be hosted in the backend of a cloud service.

The IOT device will trigger and the Raspberry PI 3 will call for an API with a particular input. The API will return and perform the function according to the input given to it. This will allow us to take control of the entire system.

Scope

The proposed work can be applied across various manufacturing and production industries where a logistics unit is present. It can be used in places where man and machine interaction is fairly high and hence the risk of accidents and fatal injury increases. Apart from accident and fatality prevention, the proposed work can be issued in places where security systems can be implemented.

3.Literature Review

S.No.	Reasearch Paper	Proposed Work	Gaps Identified
1	Wireless sensor network system using Raspberry Pi and zigbee for environmental monitoring applications By Sudhir G Nikhade 2015 International Conference on Smart Technologies and Management for Computing, Communication,	The paper deals with the Wireless Sensor Network system that has been made using the hardware platforms Raspberry Pi and zigbee. It is low cost, low power consuming and makes it much more efficient to be used across various environments. The research deals with various sensor nodes connected via ZigBee protocol and interacts with the client through a command line interface	The paper has given us a fair insight on the way to use the sensors for our application. However, its command line interface does not have high scalability. The client can only interact through a local setup network. Our project would like to implement the client

	Controls, Energy and Materials (ICSTM)		side across a cloud service system to increase the ease of access.
2	Implementation of Internet of Things for Home Automation By Mamata Khatu, Neethu Kaimal, Pratik Jadhav, Syedali Adnan Rizvi, International Journal of Emerging Engineering Research and Technology , Volume 3, Issue 2, February 2015 .	This paper deals with the automation of sensors for the implementation of IOT for home automation. The sensors are secured by SHA algorithm and Naive Bayes Classifier Algorithm is used for automated decision making. The Naive Bayes Classifier algorithm is fed with big data to get maximum accuracy and the sensors interact with a cloud frontend.	This paper uses a robust cloud architecture and a secure connection using SHA algorithm. However, the Naive Bayes Algorithm for automation brings a risk factor as its accuracy will never be 100%. Since our project deals with minimizing fatal accidents the algorithm does not solve our problem statement.
3	Intrusion detection systems for IoT-based smart environments: a survey By Mohamed Faisal Elrawy, Ali Ismail Awad & Hesham F. A. Hamed Journal of Cloud Computing volume 7, Article number: 21 (2018)	The paper suggests the use of various sensors which can be implemented to make efficient smart environments. However, the authors have suggested that the IOT system's software is not robust and identify the susceptibility of IOT systems to malware and DDOS attacks. They propose to solve this by creating a cybersecurity feature of IDS. They have elaborated about Host Based Intrusion Detection System.	The paper successfully deals with various threats that occur due to hackers and malware but is not fail safe. As the full functioning environment can come to a standstill due to a power outage, malfunction is still possible. Hence our project will focus to deal with this situation and create a failsafe.
4	IoT based Indoor Air Quality Monitoring System By Ravi Kishore Kodali; Sasweth C. Rajanarayanan	This paper deals with the indoor air quality monitoring system and uses Raspberry Pi to sense the same. The air quality index is set and if the air quality drops down the trigger level, an alert is raised. The paper	The project can only be useful in an enclosed environment, with least external influence. If there is a constant fluctuation due to installment of

	IEEE 2019	also shows how the trigger causes a safety measure to get activated.	ACs etc. around the sensor, the sensor can raise false alarms and hence it requires the environment variables to change the least for maximum accuracy.
5	IoT-based occupancy detection system in indoor residential environments By Yunwan Jeon, Chanho Cho, Jongwoo Seo, Kyunglag Kwon (2018)	This research paper uses particulate matter and air quality index to determine the occupancy of a room. It also uses detection algorithms such as a point extraction algorithm in tandem with the sensors to further improve accuracy and determine presence.	The paper discusses various algorithms that are highly accurate and create graphs to interpret results. This at times can be time taking and there may be delay to sense the occupancy of a certain area. Therefore we would require a faster technique.

4. Proposed Work

Our proposed system includes using a Raspberry Pi as a controller which reads. the data from a magnetic switch to determine if there is fatality risk inside the smart fenced area. The data is then sent to the Azure server with our custom API. Ultimately with the help of the API an output will be given whether to close the fence and activate the machine or to leave the fence open with no machinery inside the fenced zone on

The architecture of the proposed system has been explained through the representational state diagram (see Fig. 1). The start of the application can be visualized with the initialization of the Raspberry PI and a click of the button in the application which would be present on site. The magnetic switch will be read and according a status code will be generated. The status code would

be binary with 1 indicating that the fence is closed and 0 indicating that the fence will be open. The data would be sent to the Azure cloud backend as an API request.

The button click in application signifies that an API request has been sent to the Azure cloud which would in turn check the status of the fence table. The fence table would be populated according to the data received from the Raspberry PI. Once checked, the fence status can be displayed to the user and changes can be made accordingly.

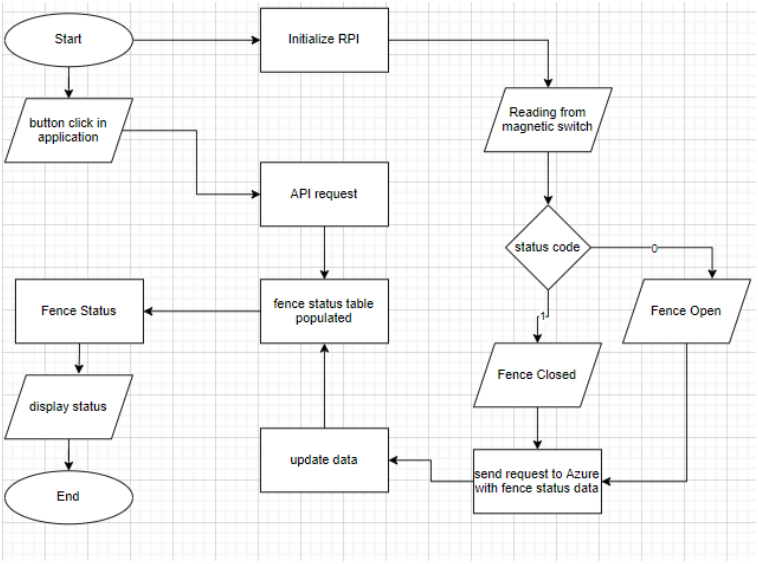


Fig. 1. Architecture Diagram of the proposed system. Each module and the flow of control can be seen with the help of unidirectional arrows.

Proposed System Module Workflow Model. The module workflow would give a detailed explanation of how each module in the proposed system will traverse and interact with the other modules and services. From the architecture, it can be concluded that the model would be divided into three components which would be hardware (Raspberry PI), cloud (Microsoft Azure) and frontend (Webpage). As seen in the workflow model (see Fig. 2), the proposed system would interact in the following manner discussed below. The hardware component would send data to the cloud service and on successful transmission of data an acknowledgement would be received to the hardware system which would stop the transmission. The frontend of the proposed system would receive data from the cloud service and send an acknowledgement back to it in a similar

fashion. Thus, communication is done in a handshaking way and a fully secure transmission is created with error handling inbuilt within the cloud service.

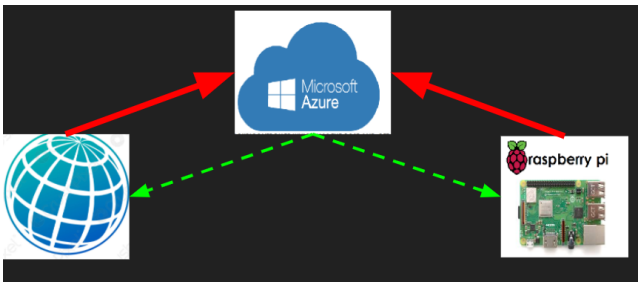


Fig. 2. Workflow diagram of the proposed system.

5. System Design and Analysis

Introduction

In this section, the system model and the analysis of the proposed system will be discussed. The proposal of making and testing of the IOT device is explained and elaborated. The UWP (Universal Windows Platform) code has to be installed in the Raspberry PI 3. The web application can be coded in C# ASP.NET and shall be tested in an IDE like Visual Studio using swagger UI and published in Azure Cloud. The webpage for the frontend would be made in HTML and can be integrated to any web service to host it.

System Specification

To build a model of the proposed system, certain number of hardware components are required. For installation, Raspberry PI 3, GPIO extension ribbon cable, magnetic switch, strong round neodymium magnets, stopper and screws, epoxy adhesive is required. For testing, a monitor, keyboard, hdmi cable and a power cable is required. As for the software requirements, visual studio and UWP is required.

Design Approach

Raspberry PI 3. The hardware component of Raspberry PI 3 installation will be divided into two sub components. Firstly, connect the magnetic switch to the raspberry pi via GPIO pin no.40 and GPIO pin no.30. This magnetic switch will act as a sensor for the magnetic field. The code in this UWP is for sensing the magnet on the stopper and consume a post API (Application Programmable interface) of the web application which is published in the Azure cloud. While consuming the POST API it sends a parameter to the API function according to the status of the fence. Also a main page or landing page coded as an xaml file. This is the frontend and used in testing purpose.

Web Application. This is a web application which is published in the azure cloud and has 2 APIs GET and POST. The post api takes in a parameter and sends that parameter to the azure cloud table storage. And the get API retrieves that particular parameter from the azure cloud table storage. Similarly, a web form needs to be designed which will be integrated with a major system which controls the entire system. According to the data retrieved, it will display the fence status and the application can be programmed to disable or enable the system control.

6. Implementation

Hardware Requirements

1. Raspberry PI 3
2. GPIO extension ribbon cable
3. Magnetic switch
4. Strong round neodymium magnets
5. Stopper and screws
6. Epoxy adhesive (to stick the magnet with the stopper)
7. Other cables and basic input output devices for a computer

Software Requirements

For Installation

1. Visual Studio 2019(UWP, web application, web form)

For Testing

1. Code in UWP (Universal windows platform)

RASPBERRY PI

Main Page.xaml.cs

• Firstly, connect the magnetic switch to the raspberry pi via GPIO pin no.40 and GPIO pin no.30. This magnetic switch will act as a sensor for the magnetic field.

• The code in this UWP is for sensing the magnet on the stopper and consume a post API (Application Programmable interface) of the web application which is published in the Azure cloud. While consuming the POST API it sends a parameter to the API function according to the occupancy of the fence.

Main Page.xaml

• This is the xaml file. this is basically the frontend of the xaml.cs file and it is only used for testing purposes that whether the code in xaml.cs is getting executed or not.

WEB APPLICATION

Occupicontroller.cs

• This is a web application which is published in the azure cloud and has 2 APIs GET and POST. The post api takes in a parameter and sends that parameter to the azure cloud table storage.

• And the get API retrieves that particular parameter from the azure cloud table storage.

** In the azure cloud there is a table created by the name ‘occupiable’ in the resource group ‘OccuPI’. This table can be created in an azure storage account.

7. Code Snippets

```
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Configuration;
using System.Text;
using System.Threading.Tasks;
namespace occupiservices.Controllers
{
public class occupancystat : TableEntity
{
public occupancystat()
{
}
public occupancystat(int fencestatusindicator)
{
occupancystatus = fencestatusindicator;
PartitionKey = "IOT";
RowKey = "IOT";
}
public int occupancystatus { get; set; }
}
public class occupiController : ApiController
{
public static occupancystat RetrieveRecord(CloudTable table, string
partitionKey, string rowKey)
{
TableOperation tableOperation =
TableOperation.Retrieve<occupancystat>(partitionKey,rowKey);
```

```
TableResult tableResult = table.Execute(tableOperation);
return tableResult.Result as occupancystat;
}
[HttpPost]
[System.Web.Http.Route("Api/post")]
public string poststatus(int statusindicator)//a is the parameter
which the
// api link will give to the function
{
string connStr =
ConfigurationManager.ConnectionStrings["ConnString"].ConnectionStri
ng;
CloudStorageAccount storageAccount =
CloudStorageAccount.Parse(connStr);
CloudTableClient client = storageAccount.CreateCloudTableClient();
CloudTable table = client.GetTableReference("status");
table.CreateIfNotExists();
var stat = new occupancystat(statusindicator);
occupancystat OCCUPANCYEntity = RetrieveRecord(table, "iot",
"iot");
if (OCCUPANCYEntity == null)
{
TableOperation tableOperation = TableOperation.Insert(stat);
table.Execute(tableOperation);
}
else if (OCCUPANCYEntity != null)
{
OCCUPANCYEntity.occupancystatus = statusindicator;
OCCUPANCYEntity.PartitionKey = "IOT";
OCCUPANCYEntity.RowKey = "IOT";
TableOperation tableOperation =
TableOperation.Replace(OCCUPANCYEntity);
table.Execute(tableOperation);
}
if (statusindicator == 1)
{
return "Occupied";
}
else
{
return "Vacant";
}
```

```
}
```

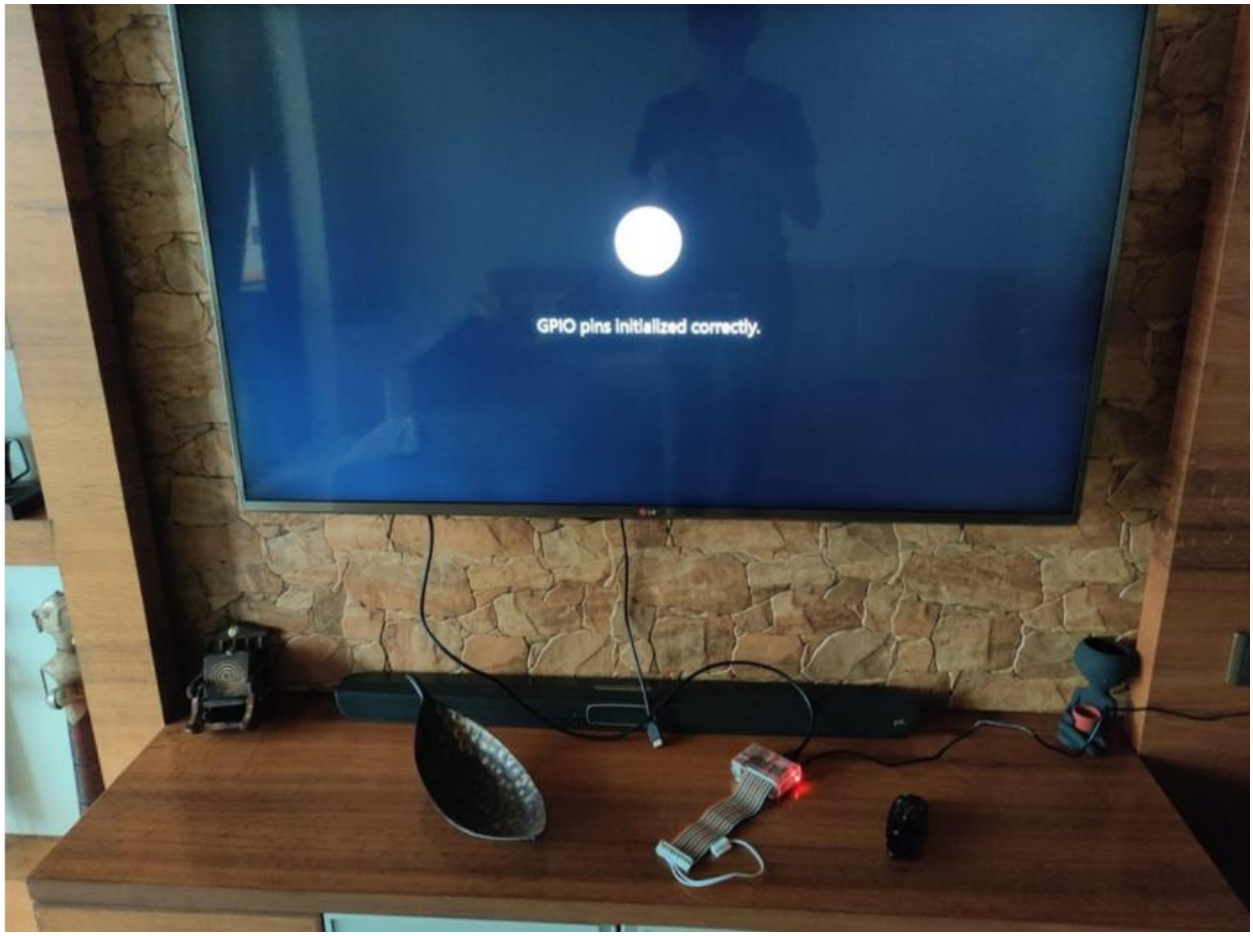
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="occupiserviceswebform.aspx.cs"
Inherits="occupiservices.occupiserviceswebform" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
</head>
<center><body background="vit2.png" style="background-
repeat:norepeat; background-position-x:center;background-size:60%
145%"></center>
<form id="form1" runat="server">
<div>
<center><asp:Button ID="clickforstatus"
OnClick="clickforstatus_Click" runat="server" Text="IOT -
FENCESTATUS" Font-Size ="20.667" ForeColor="White"
Style="backgroundcolor:red; margin-top:25%" Height="100%"
/></center><br />
<center> <asp:Label ID="lbldisplaystatus" runat="server" Font-Size
="26.667" Style="margin-top:50%" ></asp:Label></center>
<center><asp:Button ID="clickforstatus"
OnClick="clickforstatus_Click" runat="server" Text="IOT -
FENCESTATUS" Font-Size ="20.667" ForeColor="White"
Style="backgroundcolor:red; margin-top:25%" Height="100%"
/></center><br />
<center> <asp:Label ID="lbldisplaystatus" runat="server" Font-
Size="26.667" Style="margin-top:50%" ></asp:Label></center>
</div>
</form>
</body>
</html>
```

```
using System;
using Windows.Devices.Gpio;
using Windows.UI.Core;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Media;
using System.Net;
```

```
using System.Threading.Tasks;
using System.Collections.Generic;
using System.Collections.Concurrent;
using System.Linq;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;
using System.Collections.ObjectModel;
namespace gpiotest
{
public sealed partial class MainPage : Page
{
static HttpClient client = new HttpClient();//created an obj of
class
// httpclient
public MainPage()
{
InitializeComponent();
// function for initialization of componenets
InitGPIO();
// initialization of gpio function
}
public void InitGPIO()
{
var gpio = GpioController.GetDefault();//**1
// returns the default GPIO controller for the system or null if
the
// system has no GPIO controller
if (gpio == null)//**1 condition for the absence of gpio
// controller(means if there is no gpio controller then the
following set of
// statements is executed)
{
GpioStatus.Text = "There is no GPIO controller on this device.";
return;//in the xaml file name of the textblock is gpiostatus and
the
// default text is waiting to initialise gpio so if there is no
gpio then how will it
// do the following .....thus the text below will be brinted if
there is no gpio
}
buttonPin = gpio.OpenPin(BUTTON_PIN); //The pin number of the
```

```
// GPIO pin that you want to open
ledPin = gpio.OpenPin(LED_PIN);//The pin number of the GPIO pin
// that you want to open
//hence buttonpin is 6 & ledpin is 5
ledPin.Write(GpioPinValue.High);//current should flow in the
// (ledpin)pin numbered 5
ledPin.SetDriveMode(GpioPinDriveMode.Output);//The drive mode
// specifies whether the pin is configured as an input or an
output, and
// determines how values are driven onto the pin
if
(buttonPin.IsDriveModeSupported(GpioPinDriveMode.InputPullUp))
buttonPin.SetDriveMode(GpioPinDriveMode.InputPullUp);
else
buttonPin.SetDriveMode(GpioPinDriveMode.Input);
// 'Check if input pull-up resistors are supported
buttonPin.DebounceTimeout = TimeSpan.FromMilliseconds(50);//Set a
debounce timeout to filter out switch bounce noise from a button
press
buttonPin.ValueChanged += buttonPin_ValueChanged;
GpioStatus.Text = "GPIO pins initialized correctly.";
}
```

8. Screenshots of the Prototype



The GPIO Controller is initialized

When the magnet is not connected it shows vacnt

When the magnet is connected it shows occupied

```
_id: ObjectId("61adecc2d2f63b11f7320386")
status: "CLOSED"
binary: 1
system: "OFF"
lastLogin: 2021-12-06T10:58:10.356+00:00
createdOn: 2021-12-06T10:58:10.356+00:00
__v: 0
```

```
_id: ObjectId("61adeccbd2f63b11f7320388")
status: "OPEN"
binary: 0
system: "ON"
lastLogin: 2021-12-06T10:58:19.182+00:00
createdOn: 2021-12-06T10:58:19.182+00:00
__v: 0
```

```
_id: ObjectId("61aded65d2f63b11f732038a")
status: "CLOSED"
binary: 1
system: "OFF"
lastLogin: 2021-12-06T11:00:53.983+00:00
createdOn: 2021-12-06T11:00:53.983+00:00
__v: 0
```


The database hosted on cloud with the data and populated tables.

9. Conclusion

The proposed system is based on simple hardware which is easily available and cheap. It provides a cheap solution to real world problems and can be fairly used in accident prone regions to prevent fatality. The system is optimized than most other fencing system present. Overall, more research and a heavy implementation can make this system more optimized and more feasible for industry level implementation.

Future Scope

The proposed system can be improved and be used in various real world application including smart fencing for security, smart parking applications, smart fencing applications. The proposed architecture can be extended to be used in other products such as smart locking system and smart safes.

10. References

1. Nikhade, Sudhir G. "Wireless sensor network system using Raspberry Pi and zigbee for environmental monitoring applications." 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM). IEEE, 2015.
2. Khatu, Mamata, et al. "Implementation of internet of things for home automation." International Journal of Emerging Engineering Research and Technology 3.2 (2015): 7-11.Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999).
3. Elrawy, Mohamed Faisal, Ali Ismail Awad, and Hesham FA Hamed. "Intrusion detection systems for IoT-based smart environments: a survey." Journal of Cloud Computing 7.1 (2018):
4. Kodali, Ravi Kishore, and Sasweth C. Rajanarayanan. "IoT based indoor air quality monitoring System." 2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET). IEEE, 2019.
5. Jeon, Yunwan, et al. "IoT-based occupancy detection system in indoor residential environments." Building and Environment 132 (2018): 181-204.