|  |  |  |
| --- | --- | --- |
| |  | | --- | | *18/07/2022*  *DOCUMENTAZIONE* | | *SABINA*  *MARGHERITA* | |
|  |
|  |
|  |

***Indice dei Contenuti***

[***I PARTE – Progetto***](#_1fob9te)***3***

[*1. Contesto*](#_2et92p0) *3*

[*2. Progettazione architetturale*](#_1t3h5sf) *3*

[***II PARTE – Tecnologie utilizzate***](#_wd27185wjkx9) ***4***

[*1.*](#_lnxbz9) *Programmazione scheda elettronica 4*

[*2.*](#_35nkun2) *Realizzazione Gateway 5*

[*3.*](#_1y810tw) *Gestione sistema nel cloud 6*

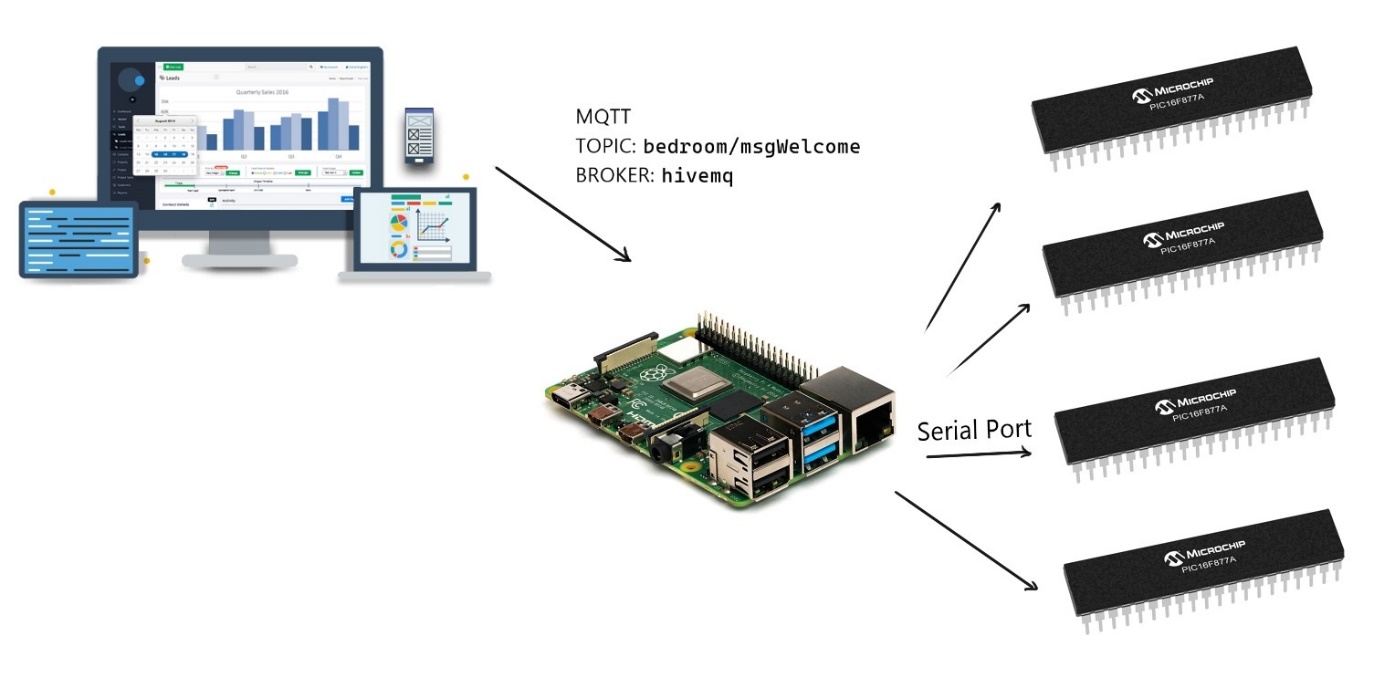
# I PARTE - Progetto

## 1. Contesto

Il progetto consiste nella realizzazione di un sistema che visualizzi dei messaggi, in delle camere di un hotel, tramite display LCD. Si richiede l’utilizzo di un PIC16 che comunica tramite BUS RS485 con un gateway che funziona da bridge con il cloud.

Dovrà venire, infine, visualizzata un’interfaccia Web che potrà impostare il messaggio da inviare e il rispettivo tempo di visualizzazione.

## 2. Progettazione architetturale



# II PARTE – Tecnologie utilizzate

## 1. Programmazione scheda elettronica

È stato usato un PIC16F877A ad una velocità di 8 MHz per la visualizzazione dei messaggi, ricevuti dal Cloud, su displayLCD.

|  |  |
| --- | --- |
| void main(void) | Se ho ricevuto un dato, lo visualizzo sul displayLCD |
| void LCD\_Send(char data, char mode) | Invio i comandi al displayLCD |
| void LCD\_Write(char \*text) | Invio il testo al diplayLCD |
| void UART\_init(long int); | Inizializzo comunicazione attraverso la seriale |
| void clearBuff(char \*, char, char \*); | Svuoto il buffer |
| void \_\_interrupt() ISR() | 1. Ricevo il dato 2. Imposto il tempo di visualizzazione del dato |

I dati che riceve dal Gateway sono così strutturati:

dataReceivedBuffer[32]

|  |  |  |  |
| --- | --- | --- | --- |
| 1 bit | 2 bit | 3 – 31 bit | 32 bit |
| Id del Pic | Numero tra 1 e 4 che rappresenta il tempo di visualizzazione del messaggio | Sono utilizzati per il testo da visualizzare, non sono fissi ma variano a seconda della lunghezza del messaggio (massimo 29 caratteri). | Alla fine c’è un carattere che indica il termine del messaggio “\n”. In base alla lunghezza del messaggio prende posto nell’ultimo bit. |

## 2. Realizzazione Gateway

È stato usato un Raspberry Pi che comunica in via Seriale con il PIC [COM5 - COM6] e tramite MQTT con il cloud (Broker hivemq).

Il gateway riceve dal cloud un JSON:

Message: {

"idGateway": 1,

"idRoom": 9,

"message": "WELCOME",

"time": 2

}

Questo contiene:

1. idGateway diverso per ogni gateway dei diversi hotel, e serve appunto a smistare il messaggio al giusto gateway
2. idRoom che identifica il PIC che deve ricevere il messaggio e quindi quale camera
3. message il messaggio che si vuole visualizzare
4. time durata del messaggio da visualizzare

Dopo aver smistato il messaggio al gateway corretto, scrive una stringa contenente:

idRoom + time + message + “\n”.

Viene aggiunto il carattere “\n” per indicare la fine della frase.

Infine, la stringa viene tramutata in bytes e inviata al PIC tramite seriale.

## 3. Gestione sistema nel cloud

L’interfaccia è stata creata tramite ASP.NET Razor Pages.

Puoi scegliere a che camera mandare il messaggio, il messaggio e la durata di esso tramite un valore da 1 a 4.

