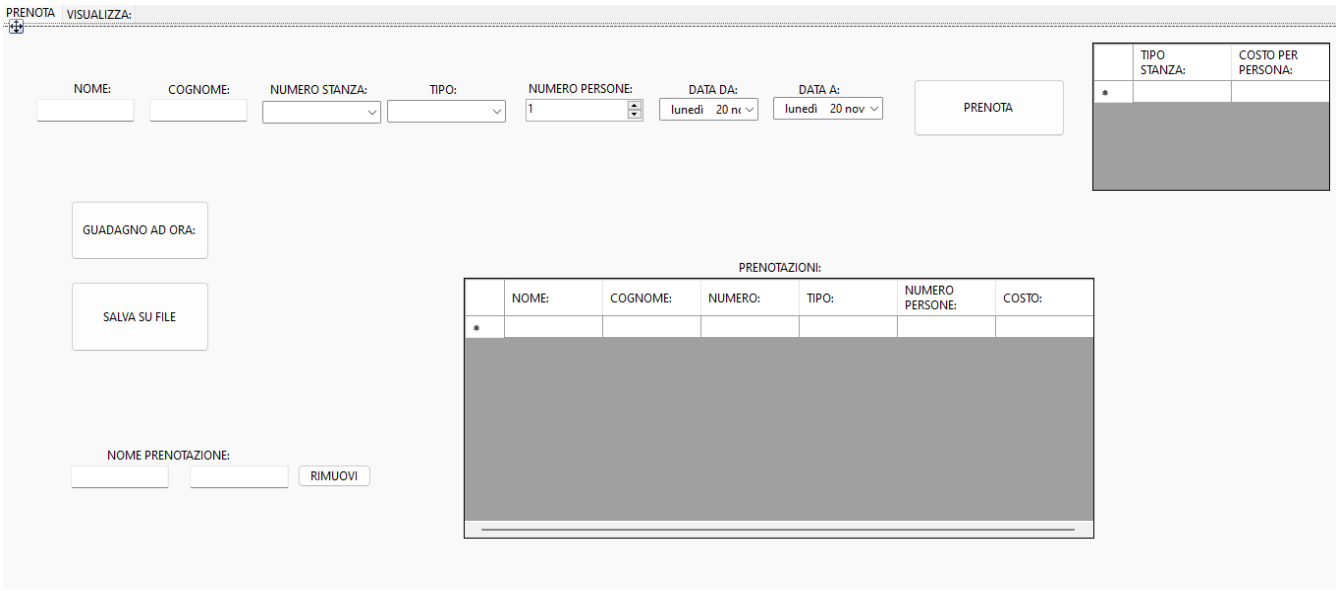
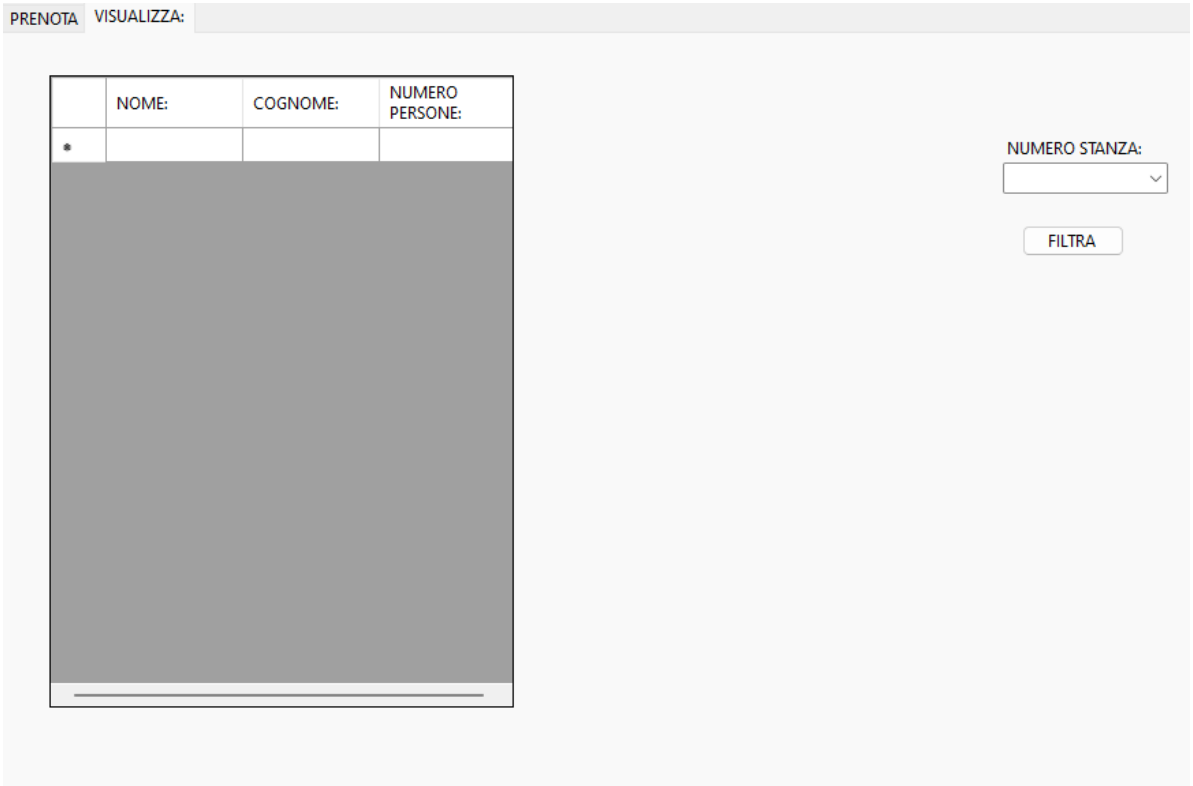
GEP - TESTING SOFTWARE

Il file preso in considerazione riguarda un progetto c# che permette di prenotare le stanze all'interno di un hotel. L'interfaccia presenta anche la possibilità al gestore dell'hotel di poter visualizzare chi ha prenotato le stanze, per quanti giorni e le caratteristiche della prenotazione.

L'interfaccia si presenta nel seguente modo:





Il modulo di prenotazione presenta diverse caselle di input per raccogliere informazioni importanti. In particolare, sono fornite caselle separate per inserire il nome e il cognome (entrambi devono essere stringhe), il numero della stanza (deve essere un numero intero), il tipo di stanza (selezionabile tramite un menu a discesa), il numero di persone (deve essere un numero intero), e infine, due selettori di data per specificare la data della prenotazione.

Assicurarsi che gli inserimenti rispettino il tipo di dato richiesto per ciascun campo. Ad esempio, i campi nome e cognome devono contenere testo, il numero della stanza deve essere un numero intero, il tipo di stanza deve essere scelto da un elenco prestabilito, il numero di persone deve essere un valore numerico, e le date devono essere selezionate tramite appositi selettori di data.

Negli input si possono presentare vari problemi:

1. L'utente inesperto potrebbe inserire all'interno della casella di input di nome e cognome dei numeri mentre ciò che il sistema si aspetta è solo caratteri alfabetici

2. Il numero di stanza non deve ripetersi per le prenotazioni su un giorno che è già stato prenotato, una volta che la stanza è stata prenotata non deve essere possibile che un altro utente la prenoti nella stessa data.

3. Nella scelta del tipo di stanza l'utente deve poter solo selezionare i tipi di data definiti dal sistema, dunque LUXURY, BASE, BUSINESS.

4. E' necessario che la data di inizio prenotazione non sia dopo quella di fine prenotazione e che le due date non coincidono. Inoltre il sistema dovrà controllare che la data selezionata non sia già passata in quanto le prenotazioni devono partire solo dai giorni successivi.

5. Il form permette inoltre di cancellare una prenotazione tramite l'inserimento di nome e cognome, e' necessario dunque anche in questo caso un controllo sul tipo di dato inserito che dovrà essere una stringa di soli caratteri alfabetici.

6. Nella pagina visualizza sarà possibile inserire solo le stanze prenotate che l'amministratore potrà andare a selezionare per visualizzare chi ha prenotato una determinata stanza .

SOLUZIONI:

1. Assicuriamoci che nelle caselle di input siano inseriti soltanto caratteri alfabetici, senza la presenza di spazi vuoti. Questo significa che l'utente deve fornire un nome e un cognome costituiti esclusivamente da lettere dell'alfabeto, senza alcun carattere numerico o spazio. Qualsiasi inserimento non conformante a queste condizioni verrà considerato come un errore.

La verifica che non siano presenti spazi vuoti viene fatta nel seguente modo:

```
if (textBox1.Text.Trim().Length * textBox2.Text.Trim().Length != 0)
```

Utilizzando la seguente condizione, verifichiamo che le caselle di testo del nome e del cognome non siano vuote. Nella pratica, estraiamo il testo da entrambe le caselle di testo, eseguiamo la rimozione degli spazi iniziali e finali mediante la funzione Trim, calcoliamo la lunghezza di entrambe le stringhe ottenute e moltiplichiamo questi valori. Se una delle due caselle di testo è vuota, il risultato della moltiplicazione sarà 0, e la condizione non sarà soddisfatta.

Per verificare che la casella di testo non contenga caratteri numerici ma solo alfabetici utilizziamo un espressione regolare:

```
Regex regex = new Regex("^[a-zA-Z]+$");
```

Il regex ^[a-zA-Z]+$ è una espressione regolare utilizzata per validare una stringa che deve consistere esclusivamente di caratteri alfabetici senza spazi. Ecco una descrizione dettagliata dei componenti di questa espressione regolare:

- ^: Indica l'inizio della stringa. La corrispondenza deve iniziare dall'inizio della stringa.
- [a-zA-Z]: Questo è un set di caratteri che indica che il carattere corrispondente deve essere una lettera minuscola (da "a" a "z") o una lettera maiuscola (da "A" a "Z").
- +: Indica che il set di caratteri [a-zA-Z] deve comparire una o più volte. In altre parole, la stringa deve contenere almeno una lettera.
- $: Indica la fine della stringa. La corrispondenza deve arrivare fino alla fine della stringa.

Dopo aver dichiarato il regex dovremo andare a fare un test sulle caselle di input:

```
Regex regex = new Regex("^[a-zA-Z]+$");
string testoTextBox1 = textBox1.Text;
string testoTextBox2 = textBox2.Text;
if (regex.IsMatch(testoTextBox1))
{
    MessageBox.Show("TextBox1: La stringa è valida.");
}
else
{
    MessageBox.Show("TextBox1: La stringa non è valida. Deve contenere solo lettere senza spazi.");
    return;
}
if (regex.IsMatch(testoTextBox2))
{
    MessageBox.Show("TextBox2: La stringa è valida.");
}
else
{
    MessageBox.Show("TextBox2: La stringa non è valida. Deve contenere solo lettere senza spazi.");
    return;
}
```

La condizione in questione verifica se entrambe le TextBox contengono soltanto caratteri alfabetici. Nel caso in cui tale condizione non sia soddisfatta, viene generato un errore e la funzione termina immediatamente tramite l'istruzione return.

2. La prima verifica da effettuare è quella che la data di inizio deve essere minore della data di fine prenotazione:

```
DateTime dataInizio = dateTimePicker1.Value;
DateTime dataFine = dateTimePicker2.Value;
if(dataInizio >= dataFine)
{
    MessageBox.Show("La data di inizio non può essere superiore o uguale a quella di fine");
    return;
}
```

Andiamo a prendere le due date e le salviamo in due specifiche variabili. Tramite la condizione verifichiamo poi se la data di inizio è superiore a quella di fine e in tal caso il form restituisce un messaggio di errore e termina l'esecuzione della funzione.

Un secondo controllo necessario è relativo alla data di inizio che deve essere obbligatoriamente successiva alla data odierna:

```
if(dataInizio <= DateTime.Now)
{
    MessageBox.Show("la data di inizio deve essere successiva alla data odierna");
    return;
}
```

L'ultimo controllo sarà relativo alla disponibilità della stanza selezionata dall'utente in un certo intervallo di date:

```
public class GestorePrenotazioni
{
    private bool[] stanzeDisponibili;

    0 riferimenti
    public GestorePrenotazioni(int numeroTotaleStanze)
    {
        stanzeDisponibili = new bool[numeroTotaleStanze];
        for (int i = 0; i < numeroTotaleStanze; i++)
        {
            stanzeDisponibili[i] = true;
        }
    }

    0 riferimenti
    public bool VerificaDisponibilitaStanza(int numeroStanza, DateTime dataInizio, DateTime dataFine)
    {
        for (DateTime dataCorrente = dataInizio; dataCorrente < dataFine; dataCorrente = dataCorrente.AddDays(1))
        {
            int indiceStanza = numeroStanza - 1;
            if (indiceStanza >= 0 && indiceStanza < stanzeDisponibili.Length)
            {
                if (!stanzeDisponibili[indiceStanza])
                {
                    return false;
                }
            }
        }
        return true;
    }
}
```

La classe fornita contiene due funzioni fondamentali. La prima funzione ha il compito di impostare tutte le stanze dell'array come disponibili, rappresentando l'iniziale stato libero di tutte le stanze quando il software viene avviato. La seconda funzione, invece, si occupa di verificare la disponibilità di una specifica stanza in un determinato intervallo di date. Questa funzione restituisce un valore booleano che indica se la stanza è o meno disponibile per la prenotazione nel periodo specificato. In sintesi, la classe offre un meccanismo per gestire lo stato di disponibilità delle stanze e per effettuare controlli di prenotazione in base alle date desiderate.

```
public void PrenotaStanza(int numeroStanza, DateTime dataInizio, DateTime dataFine)
{
    for (DateTime dataCorrente = dataInizio; dataCorrente < dataFine; dataCorrente = dataCorrente.AddDays(1))
    {
        int indiceStanza = numeroStanza - 1;
        if (indiceStanza >= 0 && indiceStanza < stanzeDisponibili.Length)
        {
            stanzeDisponibili[indiceStanza] = false;
        }
    }

    MessageBox.Show("Prenotazione effettuata con successo!");
}
```

Attraverso questa funzione, è possibile aggiornare lo stato di disponibilità di una stanza nell'array, impostando il suo valore su "false" nel caso in cui la prenotazione risulti disponibile. In altre parole, la funzione consente di prenotare una stanza non rendendola più disponibile in un determinato intervallo di date.

3. Per garantire una corretta selezione del tipo di stanza in base ai servizi desiderati (BASE, BUSINESS, LUXURY) e per evitare errori durante l'inserimento, è stata implementata una combobox, ossia un elenco a discesa. Questo permette all'utente di scegliere solo tra i tipi di stanza proposti, garantendo una selezione accurata e prevenendo eventuali problemi derivanti da inserimenti errati.

```
foreach(string n in tipo)
{
    string[] col = n.Split(',');
    comboBox1.Items.Add(col[0]);
    dataGridView1.Rows.Add(col[0], col[1]);
}
```

4. (PUNTO 2)

5. Al fine di evitare problemi relativa alle stringhe inserite per eliminare una prenotazione dovremo fare il controllo del regex anche in questo caso così da evitare inserimento di numeri indesiderati.

```
Regex regex = new Regex("^[a-zA-Z]+$");
```

6. Nella pagina visualizza si potrà andare a vedere chi ha prenotato una determinata stanza, le stanze salvate vengono inserite dall'utente che utilizza il software. Al fine di evitare problemi anche in questo caso è stata inserita una combobox con l'elenco delle camere prenotate.