PROGETTO GESTIONE ALBERGHIERA [OOP + APP]

Il mio progetto di software delle telecomunicazioni riguarda un applicazione web gestionale per la prenotazione/gestione di stanze in un albergo.

**1) SCOPO DEL PROGETTO**

* **Aggiunta di nuove prenotazioni:** I clienti possono inserire i propri dati, selezionare le date di check-in e check-out, scegliere una camera e aggiungere servizi extra. Il sistema verifica la disponibilità in tempo reale.
* **Gestione delle prenotazioni esistenti:** Una sezione dedicata consente al personale di visualizzare, modificare ed eliminare le prenotazioni in corso.
* **Visualizzazione delle statistiche:** L'applicazione fornisce una panoramica finanziaria, che include entrate lorde, costi di gestione e profitto netto per un periodo specificato. Offre anche un riepilogo dei tipi di camera e della loro disponibilità.
* **Persistenza dei dati:** Tutti i dati delle prenotazioni vengono salvati automaticamente in un file JSON locale (**gestione\_alberghiera.json**) per garantire che le informazioni non vengano perse alla chiusura e al riavvio dell'applicazione.

**2)SVILUPPO PROGETTO**

**Il progetto è stato sviluppato utlizzando :**

**BACKEND :**

**Python** con programmazione ad oggetti**(OOP) :**Questa parte, situata all'inizio del file, implementa la logica principale utilizzando un approccio orientato agli oggetti. Questa scelta rende il codice modulare e riutilizzabile. **Le classi implementate sono 5**:

* **Albergo:** Questa è la classe principale che gestisce tutte le altre. Si occupa della creazione delle camere, del caricamento e del salvataggio dei dati dal file JSON, e contiene i metodi per aggiungere, eliminare e modificare le prenotazioni, verificare la disponibilità delle camere e calcolare le statistiche finanziarie.

**Camera:** Questa classe rappresenta una camera d'albergo con attributi come numero, tipo (es. Singola, Doppia, Suite), capacità e prezzo per notte.

* **ServizioAggiuntivo:** Questa classe definisce i servizi aggiuntivi come la colazione o il parcheggio, ciascuno con un nome e un prezzo.
* **Prenotazione:** È la classe base per una prenotazione. Memorizza le informazioni del cliente, le date di check-in/check-out, la camera assegnata e un elenco di servizi selezionati. Include un metodo per calcolare il costo totale del soggiorno.
* **PrenotazioneVip:** Questa classe eredita da Prenotazione e dimostra i concetti di **ereditarietà** e **polimorfismo**. Introduce un codice VIP, applica uno sconto sul costo totale e fornisce una descrizione specifica sovrascrivendo dei metodi presenti in **Prenotazione**

**FRONTEND/UI :**

**Bootstrap(CSS)**: Organizzare il layout della pagina (griglie, colonne, contenitori),creazione interfaccia responsive(adattabile in base al display)

**Dash(HTML)** : Costruire la struttura della pagina web, titoli, paragrafi, pulsanti e collegare componenti grafici e interattivi

**Interfaccia Web (Framework Dash)**

Questa sezione è responsabile dell'interfaccia utente. Utilizza il framework **Dash** insieme a **Dash Bootstrap Components (dbc)** per costruire un'applicazione web reattiva. Il layout è strutturato usando un componente, che organizza il contenuto in **3 TAB** distinte:

1. **Nuova Prenotazione:** Un modulo per la creazione di nuove prenotazioni.
2. **Prenotazioni Esistenti:** Una tabella dati per visualizzare, filtrare e gestire le prenotazioni esistenti.
3. **Statistiche:** Un pannello di controllo per i resoconti di costi e guadagni.

Le **callback** collegano il frontend e il backend. Queste funzioni reagiscono alle azioni dell'utente (come il clic su un pulsante o la modifica di una data) e aggiornano i componenti della pagina in tempo reale senza doverla ricaricare. Ad esempio, quando un utente clicca su "Verifica Disponibilità", una funzione di callback chiama il metodo relativo alle camere del backend e aggiorna l'elenco delle camere disponibili sulla pagina.

Immagine che contiene testo, schermata, Carattere, numero

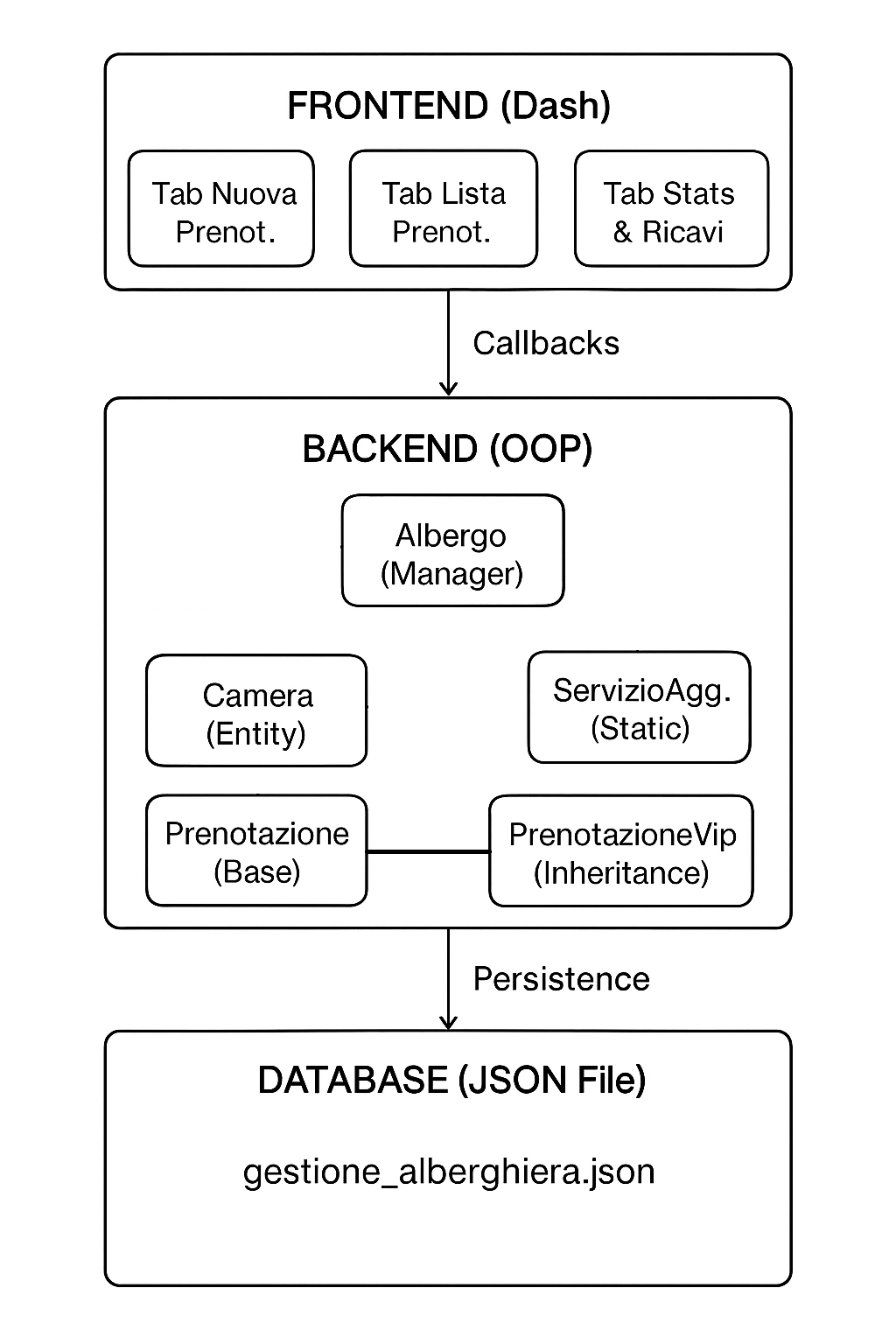
Il contenuto generato dall'IA potrebbe non essere corretto.

DIAGRAMMA SINTETICO CHE DESCRIVE L’INTERAZIONE TRA IL BACKEND E IL FRONTEND . IL SITO è COMPOSTO DA 3 PAGINE PRINCIPALI . NELLA PRIMA SI POSSONO EFFETTUARE NUOVE PRENOTAZIONI, VERIFICARE LA DISPONIBILITà DI STANZE E IMPLEMENTARE EXTRA E SCONTI. LA SECONDA è UNA TABELLA CON TUTTE LE PRENOTAZIONI EFFETTUATE, E PERMETTE L’ELIMINAZIONE, E LA MODIFICA DI DATE E EXTRA. LA TERZA PAGINA INFINE è UN SEMPLICE ELABORATORE DI COSTI SOSTENUTI E RICAVI DEFINITI IN UN CERTO PERIODO TEMPORALE . IL TUTTO VIENE SALVATO IN UN SEMPLICE DATABASE JSON PERSISTENTE.

TAB 1

Come mostrato nella tabella sotto inserendo **nome , cognome e data** in cui si desidera prenotare e eventualmente **extra**(che mostrano il costo complessivo sulla permanenza), e cliccando il pulsante Verifica Disponibilità vengono mostrate le stanze disponibili e quelle non disponibili (o perché occupate nel periodo selezionato o per un limite di numero di clienti per stanza). Immagine che contiene testo, schermata, software, Pagina Web

Il contenuto generato dall'IA potrebbe non essere corretto.ora sarà possibile selezionare tra le stanze che soddisfano i requisiti. Premendo Conferma prenotazione

Immagine che contiene testo, elettronica, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto. TAB 2 La seconda tabella è dove si trova la lista con tutte le prenotazioni effetuate

Immagine che contiene testo, schermata, numero, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.come mostrato in questa immagine vi sono tutti i dati necessari per avere prenotazioni univoche (id a 8 cifre) per evitare omonimi e semplificare le ricerche), inoltre selezionando il tasto Modifica Selezionata si aprirà la seguente schermata.

Immagine che contiene testo, schermata, numero, software

Il contenuto generato dall'IA potrebbe non essere corretto.



Che permette di modificare la data e aggiungere eventuali extra. Inoltre come possiamo vedere nella lista sotto la camera 101 è occupata da da 2 persone diverse dal18/08 al 24/08 e dal 24/08 al 27/08 Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.provando a modificare l’utente Emanuel Rosas che ha la stanza 101 a partire da 01/09 e impostando una data già occupata verrà segnalato l’errore , per evitare overfitting. Ovviamente si possono eliminare prenotazioni con il tasto Elimina Selezionata



Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.



TAB 3

La tabella 3 mostra un resoconto di un periodo determinato dall’utente con il calendario , e va a calcolare per quel range di periodo la somma dei ricavi **Lordi** delle prenotazioni. Successivamente con il tasto Calcola Ricavi dopo aver selezionato un valore verosimile dei costi giornalieri si potrà calcolare il **ricavo netto** e il **Margine** complessivo nel periodoImmagine che contiene testo, schermata, numero, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

3)**L’organizzazione logica del codice è così composta**

La prima parte riguarda l’implementazione dell’OOP con 5 classi (prime 500 righe di codice)

**Classe Camera** definisce il numero prezzo tipo (suite,singola,doppia)

**Classe servizio aggiuntivo**(wifi, piscina…) non necessità di parametri

**Classe Prenotazione** per calcolare il totale di giorni, nome cognome,id …)

**Classe Prenotazione Vip** analoga ma implementa sconto

**Classe Albergo** : creare camere,salvare e caricare dati,calcolare ricavi incassi,verifica disponibilità per modifica, trovare camere disponibili,aggiungere,rimuovere, modificare prenotazioni

Esempio di prenotazioni effettuato sul programma invece che su applicazione

**Prenotazione.descrizione** stampa una stringa del tipo:   
 return f"{self.nome} {self.cognome} - Camera {self.camera.numero}"

Immagine che contiene testo, schermata, software, Software multimediale

Il contenuto generato dall'IA potrebbe non essere corretto.Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Un ulteriore esempio in cui utilizziamo i metodi definiti dalla classe madre **Albergo** per verificare il funzionamento solo a livello di codice : con **servizi1** creo una lista che contiene solo 2 servizi e poi creo una nuova prenotazione “Alice Bianchi …, verifico le camere libere e infatti la 101 non è nell’elenco in quanto occupata da Alice nel periodo verificato. Con la funzione **Albergo.prenotazioni** stampa tutte le prenotazioni inclusa la nuova di Alice con richiami ai metodi di **calcolo del totale** e **descrizione camera**

Immagine che contiene testo, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, software, Software multimediale

Il contenuto generato dall'IA potrebbe non essere corretto.

**PARTE DI LAYOUT**

Come spiegato a inizio file il layout frontend è stato realizzato interamente in python utlizzando **Dash e Bootstrap**. Sotto, la parte di codice relativa alla prima pagina in cui come possiamo vedere si utilizza la key dbc(dash bootstrap component), ovvero la libreria che permette di formattare in modo semplice e automatico una pagina web . **dbc.row** suddivide ogni riga in 12 colonne e come possiamo vedere sotto nome e cognome occupano 6 colonne ciascuno

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, linea, Carattere, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Abbiamo inoltre implementato il calendario con la funzione **dbc.DatePickerSingle** che importa un calendario a tendina già completo, in cui si possono fare modifiche sulla data inziale e primo giorno della settimana. **dbc.Label** si utilizza per testi non cliccabili e dbc.Input invece si usa per i pulsanti come nel nostro caso il pulsante per selezionare il num. Degli ospiti. **dbc.Dropdown** per menu a tendina come quello dei servizi extra Button per pulsanti come quello di verifica disponibilità . Questo approccio permette di creare siti ben formattati e graficamente ottimizzati su diverse risoluzioni e formati. Con semplice codice in cui si possono selezionare **margini, distanze, colori** e **forme** è possibile in poche linee di codice creare siti interattivi completi. Ulteriori descrizioni dei comandi dbc sono presenti nei commenti del codice python **gestione\_alberghiera.py**

Questi codici html invece sono cio che interfaccia il frontend con i backend tramite i **callback**

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Ad esempio:

**html.Div(id="anteprima-servizi", className="mb-3")**

Contenitore per l'anteprima dei servizi selezionati

**html.Div(id="camere-disponibili", className="mb-3")**

Sezione che mostra le camere disponibili dopo una ricerca

Inizialmente vuota, si popola dopo una ricerca/callback

**dbc.Select(id="selezione-camera", options=[], style={"display": "none"})**

Menu dropdown per selezionare una camera

Le opzioni si popolano dinamicamente quando ci sono camere disponibili

**dbc.Button("Conferma Prenotazione", id="btn-conferma", color="success", disabled=True)**

Pulsante per confermare la prenotazione

**html.Div(id="output-conferma", className="mt-3")**

Area per mostrare messaggi di conferma o errore dopo la prenotazione

CALLBACK

Le callback costituiscono il cuore dell'interattività dell'applicazione. Sono funzioni Python decorate che si attivano automaticamente quando l'utente interagisce con i componenti.

Il codice relativo alle callbacks parte dalla linea 700 del file python

Le callback Dash permettono di:

* Aggiornare dinamicamente i componenti (es. lista camere disponibili)
* Evitare ricaricamenti della pagina
* Collegare direttamente le azioni dell’utente alla logica OOP

È un metodo rapido per rendere un sito funzionante e reattivo a quelli che sono diversi input provenienti da diverse fonti(esempio pulsanti , stringhe , selezione date)

Esempio:

Immagine che contiene testo, schermata, software, Software multimediale

Il contenuto generato dall'IA potrebbe non essere corretto.

**Decoratore @app.callback**

* Dice a Dash "quando succede qualcosa agli Input, esegui questa funzione"
* **Output**: Lista di 4 componenti che verranno aggiornati
* **Input**: 2 pulsanti che possono scatenare la funzione
* **State**: Dati attuali che la funzione può leggere (ma non la scatenano)

**Output - Cosa viene aggiornato:**

[Output('modal-modifica', 'is\_open'),

Output('modal-nome-input', 'value'),

Output('modal-check-in', 'date'),

Output('modal-servizi-checklist', 'value')]

**. Input - Cosa scatena la funzione:**

[Input('modifica-btn', 'n\_clicks'),

Input('salva-modifiche-btn', 'n\_clicks')]

**. State - Dati che leggiamo:**

[State('tabella-prenotazioni', 'selected\_rows'),

State('tabella-prenotazioni', 'data')]

**ctx.triggered** dice quale componente ha scatenato la callback

Se nessuno l'ha scatenata, ritorna valori di default

**Identificazione del trigger:**

**trigger\_id = ctx.triggered[0]['prop\_id'].split('.')[0]** 🡪**Capisce se è stato premuto "Modifica" o "Salva**

**C) Caso 1 - Pulsante "Modifica":**

if trigger\_id == 'modifica-btn' and selected\_rows:

prenotazione\_data = table\_data[selected\_rows[0]]

return ( True,

*# Apre il modal* prenotazione\_data['nome'],

prenotazione\_data['check\_in'],

prenotazione\_data['servizi\_extra']

questo è il funzionamento per altre info nel codice python è commentato

**CONCLUSIONI:** il progetto è stato sviluppato concentrandomi prevalentemente nel creare classi intersecabili, facilmente leggibili e facilmente modificabili. L’uso delle callback ha permesso di creare un ponte diretto tra le azioni dell’utente e la logica backend, rendendo l’applicazione reattiva e dinamica. Grazie a strumenti come dash sono stato in grado di creare un’interfaccia semplice , proporzionata e intuitiva. Se in futuro ci sarà bisogno di implementare altre funzioni , sarà agevole in quanto la struttura gerarchica permette di aggiungere e modificare elementi in modo semplice e funzionale.