**1 Introduzione**

* 1. **Scopo del sistema**

La nostra applicazione è stata pensata per poter permettere la gestione di sistema che automatizza i pagamenti dei caselli autostradali e altri servizi associati. La gestione può essere vista in due grandi aree: quella dell’utilizzatore e quella gestione, dove in quest’ultima troviamo l’amministratore e helpdesk. Le funzionalità per far sì che questo avvenga, sono le seguenti:

* Gestione dispositivi
* Gestione abbonamenti
* Visualizzazione viaggi e statistiche
* Assistenza online
  1. **Obiettivi di progettazione**
* **Usabilità:** UI facile e intuitiva.
* **Familiarità:** UI pensata seguendo i canoni della progettazione moderna.
* **Response time:** La richiesta viene soddisfatta in modo rapido ed efficiente.
* **Estensibilità:** Sviluppo che segue i principi base del software design.
* **Piattaforma di destinazione:** Web. Implementazione tramite Java + Html-Css-Js + JSP.
* **Portabilità:** La natura web fa sì che l’app possa essere utilizzata da tutti i dispositivi dotati di un’interfaccia web.
* **Gestione dei dati persistenti:** Per la memorizzazione dei dati è stato utilizzato un database SQLDeveloper.
  1. **Riferimenti**
* Accesso in varie modalità.
* Algoritmi per operazioni base e semplici.
* Adesione ai principi SOLID.
* Utilizzo dei Design Pattern per la struttura della Web-App.
* Pattern MVC.
  1. **Accesso in varie modalità**

La Web App si basa su un sistema composto in modo che al momento dell’ingresso nell’App effettuando il login, c’è una modalità di visualizzazione e di operazioni consentite diverse in base al tipo di utente connesso.

La modalità di accesso Utente permette di far le seguenti operazioni:

* Visualizzare i viaggi effettuati, con i dettagli del caso.
* Promuovere il proprio abbonamento, nella sezione gestione dispositivo, passando al Plus, che oltre all’automatizzazione del pagamento si dà accesso all’assistenza autostradale.
* Rimuovere o Inserire un nuovo veicolo da associare/associato al proprio dispositivo, così da poter automatizzare i pagamenti anche di un altro veicolo.
* Contattare l’helpdesk per inviare segnalazione o richiedere assistenza.

La modalità di accesso Amministratore, invece, permette di far le seguenti operazioni:

* Visualizzare le statistiche di accesso dei vari caselli presenti sulle diverse autostrade.
* Visualizzare e revocare i transponder attivi e associati ai vari utenti abbonati alla piattaforma.
* Inserire nuovi dispositivi che verranno utilizzati quando un nuovo utente si registrerà alla web-app.

La modalità di accesso Helpdesk permette di utilizzare le seguenti funzionalità:

* Visualizzare i ticket in pending, con i dettagli dell’utente mittente, la data in cui è stato inviato e l’utente di helpdesk a cui è assegnato. Se non è stato assegnato a nessuno è possibile autoassegnarselo.
* Rispondere a un ticket che non è stato assegnato a un altro utente Helpdesk oppure autoassegnarselo per rispondere in un secondo momento.
  1. **Algoritmi per operazioni base**

La Web App utilizza vari algoritmi sviluppati in modo da poter permettere ai diversi tipi di utenti di effettuare le varie operazioni previste per ogni modalità. Le operazioni sono quelle sopra elencate per ogni utente

* 1. **Adesione ai principi SOLID**

La Web App è stata sviluppata in modo da aderire perfettamente a quelli che sono i principi SOLID. Tali

principi fanno in modo che l’App sia realizzata con una struttura solida, estendibile e facilmente leggibile ed interpretabile da chiunque debba apportare modifiche. La parola SOLID descrive vari principi da rispettare durante lo sviluppo di un App con un linguaggio di programmazione orientato ad oggetti. I vari principi, che se rispettati rendono il codice estendibile e mantenibile, sono:

* Single Responsability
* Open/Closed
* Liskov Substitution
* Interface Segregation
* Dependency Inversion
  1. **Utilizzo dei Design Pattern per la struttura base della Web App**

Nello sviluppo della Web App sono stati usati Design Pattern per facilitare la creazione e l’uso di nuove strutture mantenendo il codice pulito e sintetico. I Design Pattern usati sono:

* **Singleton**: Questo modello consente di garantire che una sola istanza di una classe possa esistere in un’intera applicazione Java. Ciò significa che l’istanza Singleton è accessibile da qualsiasi punto dell’applicazione e rappresenta un modo efficiente e sicuro per gestire le risorse che devono essere condivise in tutta l’applicazione. E’ stato implementato per gestire la connessione al database SQLDeveloper.
* **Command**: Questo modello consente di disaccoppiare l’invocazione di un comando dai suoi dettagli implementativi, separando colui che invoca il comando da colui che esegue l’operazione. E’ stato usando per il calcolo delle tariffe dei caselli in base all’autostrada e alla classe del veicolo.
* **Decorator**: Questo modello consente permette di estendere le funzionalità di un oggetto senza modificare la sua struttura. E’ stato usato per implementare la caratteristica aggiuntiva del transponder nel caso in cui si promuove l’abbonamento a Plus.
* **Builder**: Questo modello separa la creazione di un oggetto complesso dalla sua rappresentazione, consentendo la costruzione step-by-step dell'oggetto. E’ stato usato per la costruzione del viaggio, che contenendo diverse istanze di diverse classi è ritenuto un oggetto piuttosto complesso.
  1. **Pattern MVC**

Il pattern MVC (Model View Controller) `e stato utilizzato nella Web App per gestire lo scambio di dati che avviene tra l’utente che utilizza l’app e i dati presenti nel sistema. Il pattern MVC `e un pattern dedicato allo sviluppo orientato agli oggetti. L’architettura di questo pattern prevede l’utilizzo di questo tramite tre componenti: MODEL, VIEW e CONTROLLER, che rappresentano i 3 oggetti (Entity, Boundary, Control) che rendono i modelli più resilienti alle modifiche.

* MODEL(Oggetto Entity): è una classe che incorpora la logica di business dell’applicazione relativa alla gestione dei dati del programma. Un esempio di utilizzo del Model `e la lettura o scrittura di dati da un database.
* VIEW(Oggetto Boundary): `e una classe che gestisce la GUI del programma e che ritorna all’utente i dati ricevuti dal Controller.
* CONTROLLER(Oggetto Control): `e la classe che elabora tutti i dati inseriti dall’utente nella GUI ed è la classe che ritorna i dati elaborati alla View, per poi mostrarli all’utente.

Immagine che contiene testo, schermata, Carattere, diagramma

Descrizione generata automaticamente

Figure 1: Pattern MVC

1. **Sistema Proposto**
   1. **Panoramica**

Il sistema proposto permette la gestione del proprio account a seconda del tipo di accesso effettuato. Gli utenti possono tenere traccia dei viaggi effettuati su ogni veicolo associato al proprio dispositivo, oltre all’interazione con l’utenza di Helpdesk che può dare supporto se si presenta un problema o un dubbio. Gli utenti amministratore possono controllare i dispositivi in uso e revocarli in caso di problemi, inserirne di nuovi se necessario e visualizzare le statistiche dei caselli. Gli utenti amministratore possono interagire con gli utenti tramite i ticket aperti da quest’ultimi.

* 1. **Decomposizione del sistema**

1. UserInterface: Consente di scegliere la modalità d’accesso.
2. Modalità d’accesso Amministratore

* Revoca transponder: permette di revocare un transponder associato a un utente.
* Inserisci nuovo transponder: permette di inserire un nuovo transponder.
* Visualizza statistiche: permette di visualizzare le statistiche di entrate/uscite dai caselli delle varie autostrade.

1. Modalità d’accesso Utente

* Inserisci nuovo veicolo: permette di inserire un nuovo veicolo da associare al proprio abbonamento/dispositivo.
* Passa a Plus: permette di promuovere il proprio abbonamento da semplice a Plus (assistenza stradale).
* Visualizza viaggi: permette di visualizzare i viaggi effettuati con i propri veicolo associati.
* Apri ticket: permette di inviare una segnalazione all’helpdesk inviando un ticket.

1. Modalità d’accesso Helpdesk

* Visualizza ticket: permette di visualizzare i ticket in pending aperti dagli utenti.
* Rispondi ticket: permette di inviare una risposta all’utente mittente del ticket e chiuderlo.

1. SQLDeveloper : è lo spazio del database dove tutti i dati vengono salvati per poi essere letti dalle view che devono mostrare questi dati all’utente.
   1. **Hardware/Software mapping**

Il sistema è progettato per garantire la massima disponibilità e può essere utilizzato da qualsiasi PC o altro dispositivo connesso ad internet, a condizione che l’utente (amministratore, utente semplice, helpdesk) abbia i permessi per utilizzare la propria tipologia di utenza. Il sistema può essere utilizzato in qualsiasi momento.

**2.4 Gestione dei dati persistenti**Telepass prevede la memorizzazione persistente di una grande quantità di dati mediante l’utilizzo di un database relazionale. Ciò permette di mantenere traccia di tutti i dati generati, richiesti, da elaborare e da trasmettere. Molte delle informazioni visualizzabili dagli utenti vengono ottenute eseguendo query sul database, il che comporta un alto tasso di utilizzo del database stesso. Per questo motivo, molte delle classi del modello a oggetti sono state progettate tenendo presente il mapping con le tabelle di un eventuale database.

* 1. **Controllo accessi e sicurezza**

Poiché il progetto prevede tre diverse tipologie di utenze, è possibile avere a seconda dell’utenza associata, di operare in modalità Amministratore, modalità utente semplice o modalità Helpdesk. Possiamo quindi definire ciò tramite tre aree:

**Area Utente semplice (Utente):** Area di un utente semplice che può avere accesso solo ai propri dati.

**Area Amministratore(Amministratore):** Area di amministratore dove è possibile gestire i diversi dispositivi e informazioni riguardo l’utilizzo.

**Area Helpdesk(Helpdesk):** Area di un utente Helpdesk dove c’è accesso solo ai ticket inviati dai clienti e permette l’interazione con un utente con la risposta alla segnalazione.

|  |  |
| --- | --- |
| ATTORE | AZIONI |
| Amministratore | Gestione dispositivi |
| Utente | Gestione abbonamento |
| Helpdesk | Gestione assistenza |