UNIVERSITÀ DEGLI STUDI DI PALERMO

Metodi Avanzati per la Programmazione

ClockTask

Studente
Giulia Maraventano

Docente
Prof. Gabriele Fici

30/03/2021

A.A. 2020/2021

Abstract

Il progetto "ClockTask" è stato da me realizzato per l'esame della materia "Metodi avanzati per la Programmazione", insegnata dal Prof. Gabriele Fici.
Il software che ho implementato permette di registrare il tempo impiegato per svolgere una o più attività e di inserire, per ciascuna attività, delle note datate. Inoltre permette di generare una fattura proforma relativa al tempo impiegato e al prezzo orario del lavoro del professionista.

Indice

Introduzione

Il Software descritto nel seguente documento può essere utilizzato da un libero professionista che svolge una o più attività, per monitorare il tempo impiegato ed eventualmente emettere una fattura proforma.
Come è noto, la fattura proforma è un documento privo di valore fiscale. Essa viene emessa prima della fattura fiscale vera e propria, per verificare il prezzo e la correttezza dei dati, evitando, in caso di errori, di dover emettere note di credito per stornare la fattura precedente.
L'utilizzatore del software deve inserire i dati relativi alle sue attività in corso, quelli relativi al cliente e il valore orario del proprio lavoro. Tali dati possono essere modificati anche in corso d'opera.

Inizialmente viene descritto il modello generale del software, ideato e progettato con gli strumenti appresi nello studio della disciplina "Ingegneria del Software".
Verso la fine vengono evidenziati alcuni dettagli tecnici, elaborati dopo l'implementazione del codice sorgente.

Analisi dei requisiti

Requisiti funzionali

- Creare, modificare il profilo Freelance.
- Creare, modificare e cancellare un Cliente.
- Creare, modificare e cancellare un Task.
- Gestire il tempo di svolgimento di un Task, avviando e fermando il timer all'interno del Task.
- Inserire delle note relative al Task.
- Salvare le note e il tempo impiegato.
- Aprire uno storico del Task.
- Generare una fattura proforma, attraverso un'anteprima, con i dati del profilo e del Task, calcolata in base al valore orario del lavoro del freelance e al tempo impiegato per lo svolgimento del Task.

Requisiti non funzionali

- Interfaccia grafica semplice.
- Utilizzo del formato PDF per la fattura proforma.
- Utilizzo del formato .json e .txt per la memorizzazione dei dati. Trattandosi di un programma stand-alone, non ci sono rischi sulla sicurezza e quindi non è necessario né un form di login per l'accesso né una connessione ad un database, che appesantirebbe programma. È sufficiente il salvataggio dei dati in un file di testo, in formato json per una manipolazione più avanzata di tali dati.

Modello del sistema

Scenario di successo

L'utente crea un nuovo Cliente, inserendo i dati relativi (Nome, Ragione Sociale, Partita IVA, Indirizzo e Telefono). Il Cliente viene inserito nella rubrica dal sistema, previo controllo della correttezza dei campi inseriti (ES.: La partita IVA deve contenere precisamente 11 cifre numeriche che iniziano con 0, ecc).

L'utente crea un nuovo Task, dandogli un nome associandolo ad uno dei Clienti della rubrica, per il quale svolge il lavoro. Il sistema inserisce il task nella lista dei Tasks.

L'utente seleziona e apre un Task. La schermata relativa al task selezionato mostra un riepilogo dell'attività (nome del Task, cliente associato, valore orario e costo attuale del task), un timer con i tasti "Play" e "Stop" , uno spazio per le note e un riquadro con i pulsanti (Salva Task, Storico, Genera Fattura, Torna Indietro).

L'utente avvia il tempo di svolgimento dell'attività, premendo il pulsante "Play". Il sistema attiva il timer e lo mostra a video.

Durante l'attività lavorativa, l'utente inserisce delle note all'interno dell'area di testo del Task.

Al termine della sua attività, l'utente ferma il tempo, premendo il pulsante "Stop". Il sistema disattiva il timer.

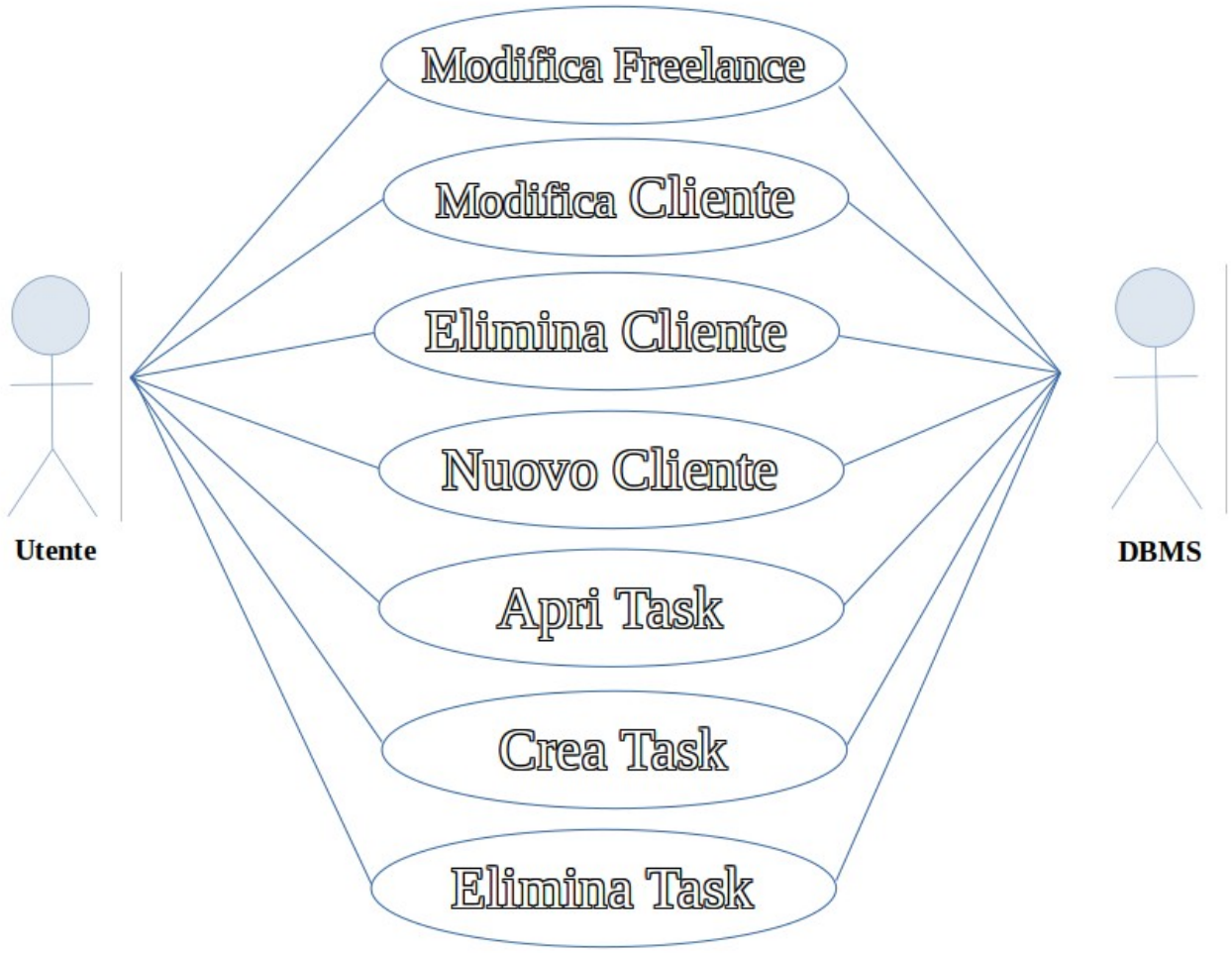
L'utente salva il Task. Il sistema memorizza la data, il valore del tempo impiegato e la nota inserita.

L'utente apre lo storico note relativo al Task. Il sistema apre un file PDF che mostra tutte le note.

L'utente genera la fattura proforma. Il sistema apre il file PDF della fattura, contenente i dati del freelance e del committente e il prezzo totale del task comprensivo di IVA.

Diagrammi e descrizioni dei casi d'uso

Vista di insieme



Nome caso d'uso	Modifica Freelance
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente clicca sul pulsante "Modifica Profilo", nella schermata principale.
Flusso di eventi	**2.** DBMS apre una nuova finestra con un form contenente i campi del profilo del freelance. **3.** L'utente edita i campi e clicca su "OK" per salvare i dati. **4.** DBMS modifica in memoria i dati del freelance.

Nome caso d'uso	Nuovo Cliente
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente clicca sul pulsante "Nuovo Cliente", nella schermata principale.
Flusso di eventi	**2.** DBMS apre una nuova finestra con un form contenente i campi del cliente. **3.** L'utente compila i campi e clicca su "Inserisci" per salvare i dati. **4.** DBMS modifica in memoria i dati del freelance.

Nome caso d'uso	Modifica Cliente
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente seleziona dalla lista un Cliente e clicca sul pulsante "Modifica Cliente", nella schermata principale.
Flusso di eventi	**2.** DBMS apre una nuova finestra con un form contenente i campi del cliente, contenenti i dati relativi. **3.** L'utente modifica i campi e clicca su "Inserisci" per salvare i dati. **4.** DBMS modifica in memoria i dati del freelance.

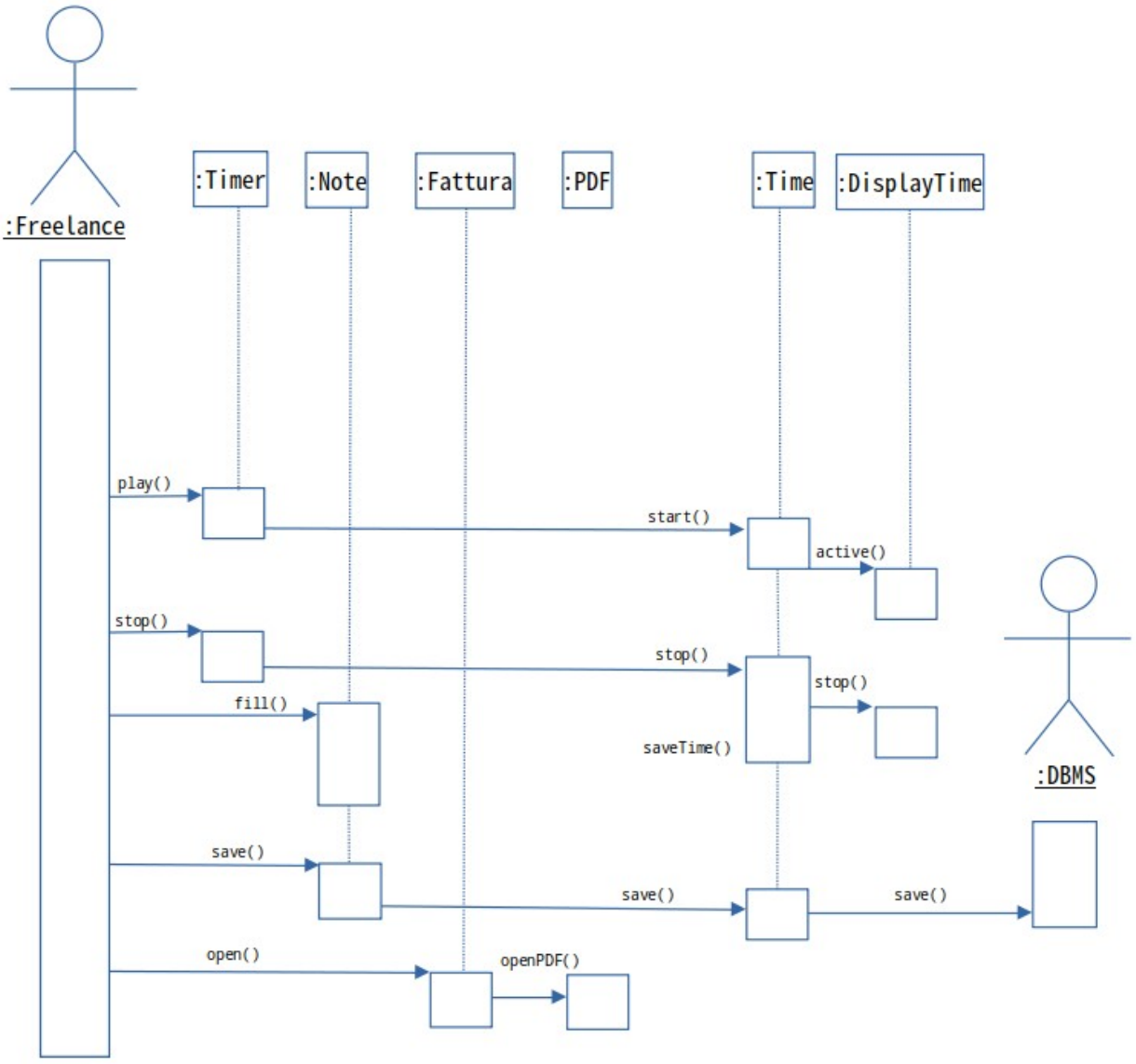
Nome caso d'uso	Elimina Cliente
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente seleziona dalla lista un Cliente e clicca sul pulsante "Elimina Cliente", nella schermata principale.
Flusso di eventi	**2.** DBMS, dopo aver chiesto conferma, cancella il cliente e lo elimina dalla lista.

Nome caso d'uso	Nuovo Task
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente clicca sul pulsante "Nuovo Task", nella schermata principale.
Flusso di eventi	**2.** DBMS apre una finestra contenente un campo di testo per assegnare un nome al task e un menu a tendina per selezionare il Cliente della rubrica per il quale si svolge il task. **3.** L'utente inserisce i valori e clicca sul pulsante "Inserisci" per salvare i dati. **4.** DBMS salva i dati del Task e inserisce quest'ultimo nella lista dei Tasks.

Nome caso d'uso	Elimina Task
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente seleziona un Task e clicca sul pulsante "Elimina Task"
Flusso di eventi	**2.** DBMS elimina il Task selezionato, previa richiesta di conferma.

Nome caso d'uso	Apri Task
Attori	Utente, DBMS
Condizione di ingresso	**1.** L'utente seleziona un Task e clicca sul pulsante "Apri Task"
Flusso di eventi	**2.** DBMS apre una nuova finestra contenente: - Un campo e un pulsante per modificare il nome del Task. - Un Timer con i pulsanti "Play" e "Pausa" per l'avvio e lo stop del tempo dello svolgimento del Task. - Un' area di testo per inserire delle note. - Un pulsante per salvare la nota. - Un pulsante "Genera Fattura Proforma". 3.**1.** L'utente edita il campo Nome Task e clicca sul pulsante "Modifica Task" 3.**2. DBMS** aggiorna il nome del Task in memoria. **4.1.** L'utente clicca sul pulsante "Play" del timer. 4.**2.** DBMS avvia il tempo del timer, partendo dal punto in cui da dove lo aveva lasciato. 5.**1.** L'utente clicca sul pulsante "Pausa" del timer. **5.2.** DBMS ferma il timer a video e salva il valore del tempo in memoria. 6.**1**. L'utente inserisce una nota (facoltativo) e clicca sul pulsante "Salva". **6.2** DBMS salva la nota insieme alla data di salvataggio. 7.**1.** L'utente clicca sul pulsante "Genera Fattura Proforma" 7**,2.** DBMS apre una schermata di anteprima in PDF con tutti i dati e i relativi prezzi, calcolati in base al tempo impiegato allo svolgimento del Task.

Diagramma di sequenza del ClockTask



Diagrammi delle classi

I diagrammi delle classi sono illustrati nei seguenti file:

FILE **Main Class Diagram.png**:

(Diagramma generale delle classi)

FILE FileIO Class Diagram.png:

(Diagramma delle classi relative al salvataggio dei dati in formato .json)

Design Pattern e classi coinvolte

Le Classi utilizzate per lo sviluppo del software sono le seguenti:

Package objects:

Freelance - Classe con gli attributi relativi al profilo del Freelance.

Customer - Classe con gli attributi relativi al profilo del Cliente.

Task - Classe con gli attributi relativi al Task.

Package filePDF:

PDF - Interfaccia per la creazione e l'apertura di un file PDF.

Fattura – Interfaccia che estende l'interfaccia PDF, per la creazione della fattura.

FatturaPDF - Classe che implementa l'interfaccia Fattura, per la creazione e la visualizzazione di una fattura proforma.

Note - Interfaccia che estende l'interfaccia PDF, per la creazione delle note.

NotePDF Classe che implementa l'interfaccia Note, per la visualizzazione delle note in un file PDF.

Package fileIO:

FileIO – Interfaccia per l'esecuzione di un comando su un file json.

FileJson - Classe per la gestione dei dati in memoria, che contiene tutti i metodi necessari per il salvataggio, la lettura e la cancellazione dei dati in un file .json.

JsonSave, JsonAdd, JsonRead, JsonDelete, JsonReplace, JsonReplace, JsonReadObject, JsonDeleteObject – Classi che implementano l'interfaccia FileIO per la gestione dei dati in memoria.

Package gui:

L'interfaccia grafica e i suoi oggetti, come per esempio i campi di testo, i bottoni, le labels ecc., sono costruiti in classi statiche, estensioni della classe Jframe, appartenenti allo specifico package "gui", al fine di separare la componente grafica da quella funzionale, che invece è implementata in altre classi, estensioni a loro volta di quelle statiche.

Home – Classe interfaccia grafica funzionale estensione di:

JHome - Interfaccia grafica statica

CustomerForm - Classe Interfaccia grafica funzionale per la creazione o la modifica del Cliente, estensione di:

Jform - Interfaccia grafica statica

TaskForm - Classe Interfaccia grafica funzionale per la creazione di un task, estensione di:

Jform - Interfaccia grafica statica

JTaskPanel - Interfaccia grafica relativa ad uno specifico Task.

ClockTaskPanel – Classe che implementa l'interfaccia JTaskPanel.

MyTimer - Una classe estensione di JPanel per la visualizzazione del timer. Si tratta di un oggetto timer indipendente, utilizzabile anche in altre applicazioni e che sfrutta la classe JavaTime.

Pattern Creazionali

Per la creazione del Freelance, del Cliente e della Fattura, ho scelto il pattern "Builder", in quanto, le classi relative presentano numerosi attributi. Nelle classi Freelance e Customer ho inserito un metodo per il salvataggio degli attributi su file: rispettivamente salvaFrelance() nella classe Freelance e salvaCustomer() nella classe Customer. Questo consente di rendere l'inizializzazione dell'oggetto e il suo salvataggio indipendenti tra loro. Entrambe le classi presentano due costruttori: uno avente tutti i parametri di ingresso e uno senza nessuno di essi. Questo ulteriore costruttore serve soprattutto per inizializzare l'oggetto Freelance senza alcun parametro in ingresso e potere così chiamare il metodo getFreelance(), che preleva i dati dal file, in cui è contenuta l'unica istanza dell'oggetto. Lo stesso tipo di logica è stata applicata alla classe Task, anche se essa non ha bisogno del pattern Builder per costruire un oggetto.

Pattern Strutturali

Per la creazione e la visualizzazione di un file PDF, che contiene dei valori specifici, ho fatto ricorso al pattern "Decorator". Infatti, l'interfaccia principale (PDF), che implementa il metodo di apertura del file PDF (open()), può essere estesa da altre interfacce che implementano anche altri medodi.

Nel caso specifico, ho creato due interfacce, Note e Fattura, estensioni della classe principale. Esse implementano in più il metodo create(), per creare il contenuto del file PDF.

Le classi che eseguono concretamente questo metodo sono NotePDF e FatturaPDF.

Pattern Comportamentali e Refactoring

Per evitare il problema di ridondanza di porzioni di codice, durante il salvataggio dei dati nei relativi file .json, ho scelto come soluzione quella del pattern "Command".

Ogni tipo di comando, come il salvataggio, la lettura, la cancellazione di un oggetto JSON, viene eseguito nella classe FileJson, attraverso i rispettivi metodi.

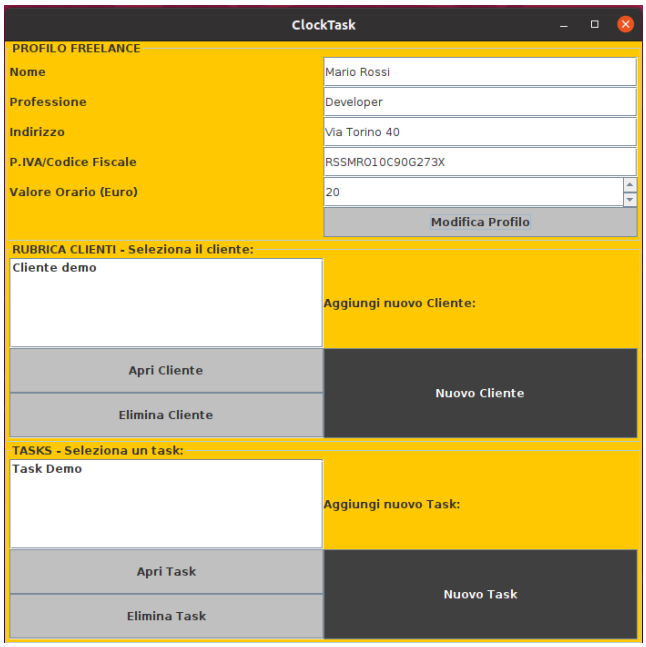
I metodi di questa classe vengono poi invocati, ciascuno, da una classe, la quale riveste il ruolo del metodo che invoca (es.: la classe JsonSave che invoca il metodo save(), avrà il ruolo di salvare il dato). Inoltre, ciascuna di queste classi, che invocano i rispettivi metodi, implementano l'interfaccia generale (JsonIO) che esegue il generico comando execute().

Il controllo dei dati inseriti nei campi, come la partita IVA o il codice fiscale, viene effettuato in una classe a parte, in modo che possa essere utilizzato da piu classi.

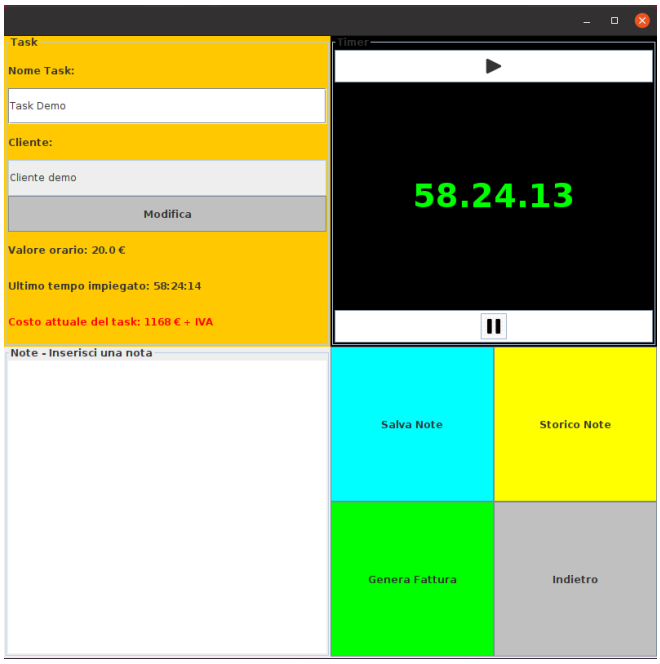
Descrizione del Software e mock-ups

All'avvio del programma, appare la schermata principale composta da:

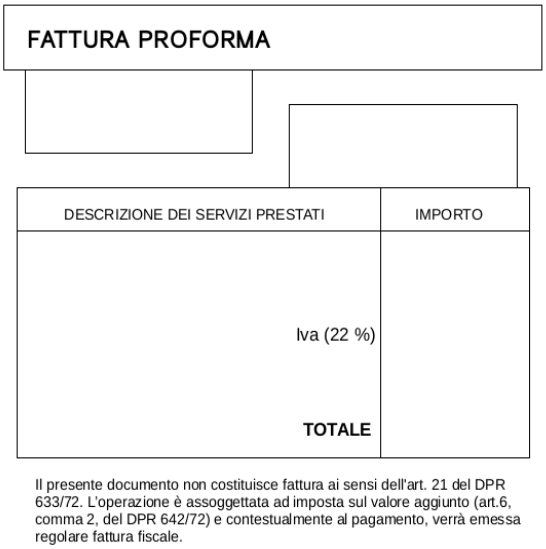
- Un riepilogo dei dati del profilo freelance, editabili e modificabili mediante l'apposito pulsante.
- Una lista di clienti, selezionabili. E' possibile aprire e modificare le informazioni di ogni cliente (il nome, la ragione sociale, l'indirizzo, la partita IVA e il numero di telefono). Inoltre è possibile creare un nuovo cliente o eliminarne uno esistente, mediante gli appositi pulsanti.
- L'elenco dei Task. Anche in questo caso è possibile creare un nuovo task o eliminarne uno esistente. Quando si crea un nuovo Task, si deve selezionare il cliente per il quale si svolge l'attività e dare il nome al task. Pertanto, prima di creare un task, bisogna avere in rubrica almeno un cliente.



Aprendo un Task, si apre una finestra con quattro riquadri. Nel riquadro in alto a sinistra ci sono le informazioni del task (il nome modificabile e il cliente). Sulla destra un timer, inizialmente spento, che si può avviare al momento dell'inizio dello svolgimento della sessione del task e stoppare alla fine. In basso c'è un'area di testo in cui poter inserire delle note riguardo alla sessione del task.
Infine, il riquadro in basso a destra mostra quatto pulsanti colorati.
Il pulsante "Salva", che permette salvare le eventuali note inserite nel riquadro delle note, con la data e l'orario in cui sono state inserite.



Il pulsante Storico apre una finestra che mostra tutte le note.
Il pulsante "Genera Fattura" apre un file PDF, con il prezzo della prestazione, calcolato in base all'ultimo tempo salvato e al valore orario del Freelance.
Il pulsante "Indietro" permette di tornare alla Home.


FATTURA PROFORMA

DESCRIZIONE DEI SERVIZI PRESTATI | IMPORTO

Iva (22 %)

TOTALE

Il presente documento non costituisce fattura ai sensi dell'art. 21 del DPR 633/72. L'operazione è assoggettata ad imposta sul valore aggiunto (art.6, comma 2, del DPR 642/72) e contestualmente al pagamento, verrà emessa regolare fattura fiscale.

E′ importante sottolineare che, quando il timer è in esecuzione, è possibile effettuare qualsiasi azione sul pannello del task, quindi è possibile scrivere una nota, guardare lo storico e generare la fattura. Tuttavia, non è possibile premere il pulsante "Indietro" per uscire dal task, in quanto l′oggetto timer verrebbe distrutto mentre è in attività.

Conclusioni e possibili sviluppi futuri

Questo software è ideale per chi dà un valore orario alla propria attività. Aiuta a tener traccia del tempo impiegato per tutto il periodo dello svolgimento dell'attività.

Una possibile versione avanzata di questo software, potrebbe prevedere il valore orario pesato per ogni genere di attività o per uno specifico argomento dell'attività. Si può così creare una fattura proforma più dettagliata, con tutte le voci riguardanti il progetto e il relativo prezzo. Inoltre si potrebbe creare anche un'apposita sezione per generare un preventivo.

Riferimenti Bibliografici

- Materiale didattico del corso di *Metodi Avanzati per la Programmazione* dell'anno accademico 2020/2021, tenuto dal Professore Gabriele Fici

- Cay S. Horstmann - Core Java SE 9 for the Impatient

- Raoul-Gabriel Urma, Mario Fusco, Alan Mycroft - Modern Java in Action

- Adrian Ianculescu, Kamalmeet Singh, Lucian - Paul Torje - Design Patterns

and Best Practices in Java

- Joshua Bloch - Effective Java (3rd edition)

- Bernd Bruegge & Allen H. Dutoit - Object-Oriented Software Engineering Using UML, Patterns, and Java

- Rohit Joshi – Java Design Patterns