Relazione del Progetto relativo l'insegnamento di Programmazione e Modellazione a Oggetti per la sessione Invernale 2020/2021.

Relatore:
Matteo Marco Montanari

Docente:
Prof. Saverio Delpriori

1) Specifica del software:

Si progetti un applicativo software dotato di semplice interfaccia grafica per gestire dei database di password.

1. I database dovranno essere costituiti da una serie di record composti da: nome utente e relativa password.
2. La memorizzazione dei database sarà effettuata in locale su file di testo in formato JSON.
3. Le password dovranno essere crittografate.
4. Sarà possibile cercare, inserire ed eliminare dei record all'interno dei database.
5. Il software dovrà includere una funzionalità per generare delle password casuali con un numero variabile di caratteri scelto dall'utente.
6. L'accesso al singolo database dovrà essere protetto da una Master Password.
7. L'applicativo supporterà la multiutenza, gestita tramite account (nome utente e password). La registrazione degli utenti sarà gestita tramite file analogamente ai relativi database di password.
8. Dovrà essere presente un account di default comune e accessibile a tutti gli utenti.

2) Studio del Problema:

I punti critici nella progettazione del suddetto software sono i seguenti:

1. La gestione dei file di memorizzazione dei database e delle credenziali di accesso degli utenti registrati al servizio.
2. L'interfaccia grafica che permetterà di accedere con il proprio account e gestire il database relativo all'utente.
3. La gestione dell'account di default.

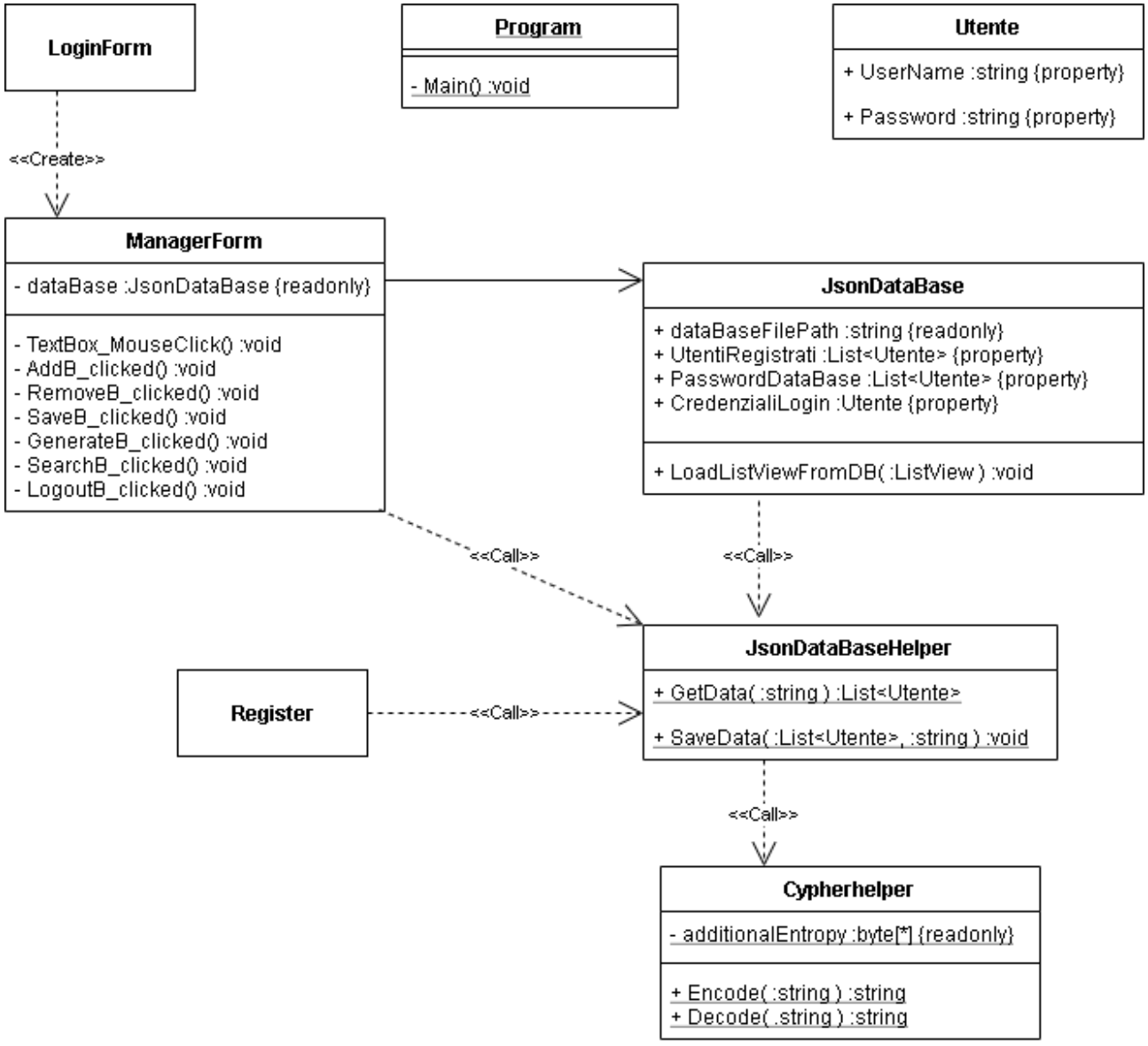
Da cui sono derivate le seguenti scelte di progetto:

1. I file contenenti i vari database sono gestiti in maniera analoga all'unico file di memorizzazione delle credenziali degli utenti registrati. Questa scelta è dovuta al fatto che le informazioni da registrare in entrambi i casi sono dello stesso tipo (nome utente e password). Si è scelto inoltre di gestire la creazione e la modifica dei vari file in maniera trasparente all'utente. In particolare l'aggiornamento dei vari database viene effettuato in due fasi: nella prima si caricano le eventuali informazione del database salvate in precedenza; nella seconda quando viene salvato il database il corrispondente file di memorizzazione viene sovrascritto.
2. L'interfaccia grafica è costituita da due schermate distinte che si alternano durante l'esecuzione del software. In particolare la prima di esse è utilizzata per permettere all'utente di registrarsi al servizio e fare login con il proprio account. Questa schermata dà dunque l'accesso alla seconda che permette di gestire il database relativo all'utente che ha effettuato il login. Quest'ultima inoltre sopperisce alle altre funzionalità richieste dalla specifica. Tramite un pulsante è poi possibile tornare alla prima schermata.
3. L'account di default e relativo database saranno gestiti tramite una specifica modalità di accesso presente nella prima schermata e disponibile per tutti gli utenti. In questo modo si accederà al suddetto database con le credenziali di default (nome utente: admin, password: admin).

3) Scelte architetturali:

Si è scelto per chiarezza e modularità di suddividere il diagramma delle classi in due parti che corrispondono alle due schermate sopradette che compongono il software (ManagerForm la seconda e LoginForm la prima).

ManagerForm UML:

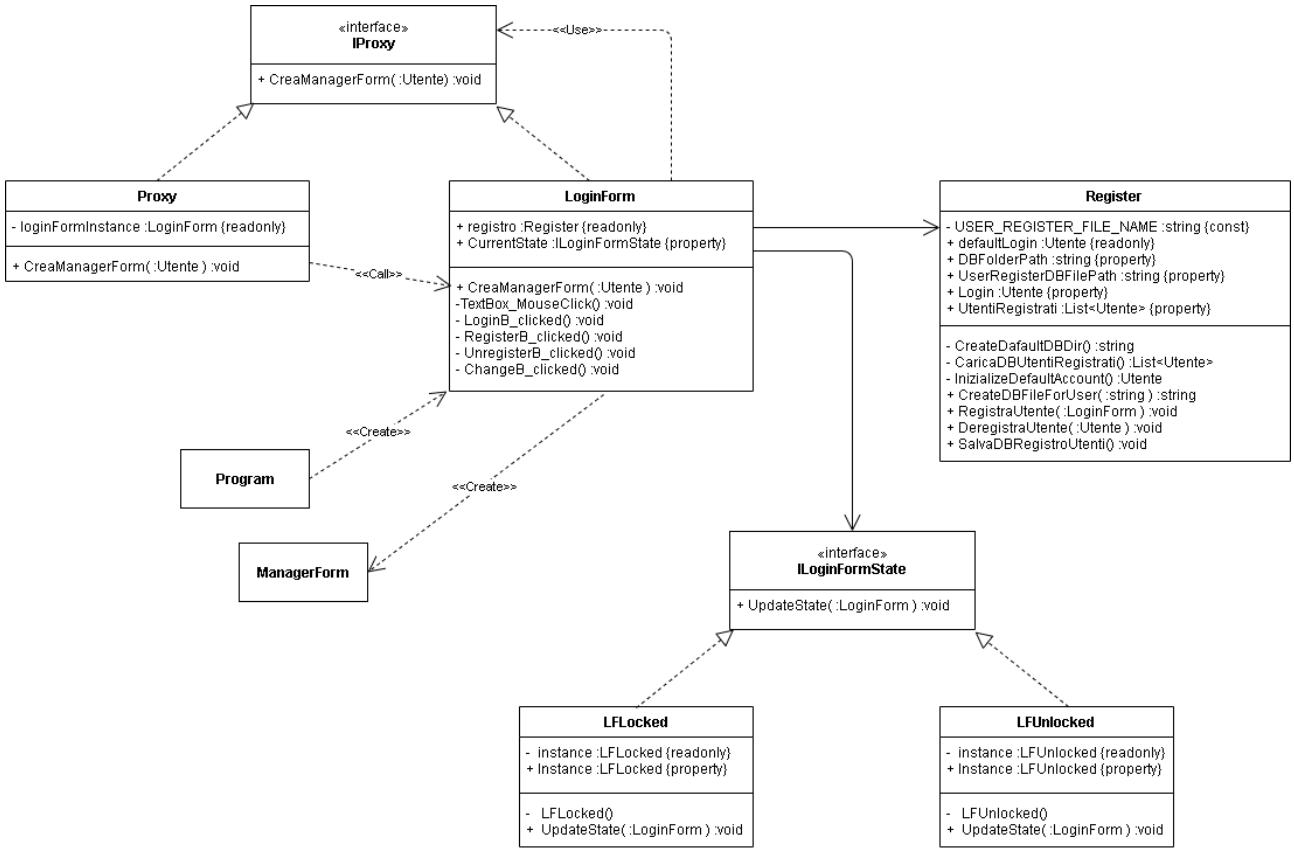


Nel digramma sopra riportato vengono sottolineate le seguenti relazioni:

- Un oggetto di tipo LoginForm è adibito alla creazione di una istanza della classe MangerForm.
- ManagerForm possiede metodi per la gestione della form stessa e un attributo di tipo JsonDataBase di ausilio alla form.

- Le classi ManagerForm, Register, e JsonDataBase hanno metodi che richiamano quelli della classe JsonDataBaseHelper.
- A sua volta la classe JsonDataBaseHelper richiama nei suoi metodi quelli della classe CypherHelper.

LoginForm UML:



Nel digramma sopra riportato vengono sottolineate le seguenti relazioni:

- La classe LoginForm possiede un attributo di tipo Register, con una serie di campi, di ausilio alla form.
- Inoltre possiede un attributo di tipo ILoginFormState, interfaccia che memorizza lo stato della form (State Pattern).
- Uno dei metodi della LoginForm (LoginB_clicked()) utilizza un oggetto di tipo IProxy, interfaccia implementata da un oggetto di tipo Proxy e dalla form stessa. Inoltre il metodo CreaManagerForm(…) di un eventuale oggetto di tipo Proxy richiama il metodo omonimo della LoginForm dopo avere effettuato dei controlli. Queste tre classi permettono la creazione di un oggetto di tipo ManagerForm (da parte della LoginForm) tramite il pattern Protection Proxy.
- Infine la classe statica (vedi ManagerForm UML) Program, il punto di partenza dell'applicazione, crea un oggetto di tipo LoginForm.

Design Pattern utilizzati:

1. Creazionale per oggetti: Singleton
   Il pattern Singleton è stato implementato nelle classi LFLocked e LFUnlocked in quanto solo una istanza di ciascuna classe, che rappresenta uno stato della LoginForm, deve essere istanziabile a tempo di esecuzione. Entrambe le classi presentano un costruttore privato, un attributo privato statico che memorizza un'istanza specifica della classe stessa e una proprietà pubblica statica che restituisce sempre tale unica istanza.

2. Strutturale per classi: Proxy
   Il Protection Proxy è utilizzato per delegare a un'altra classe (Proxy) il controllo delle credenziali di login prima di dare l'accesso alla ManagerForm tramite il metodo CreaManagerForm(…) della classe LoginForm. Questo pattern permette inoltre di saltare tale controllo a tempo di esecuzione per l'accesso tramite account di default. LoginForm e Proxy implementano l'interfaccia IProxy e dunque il metodo per creare la ManagerForm.

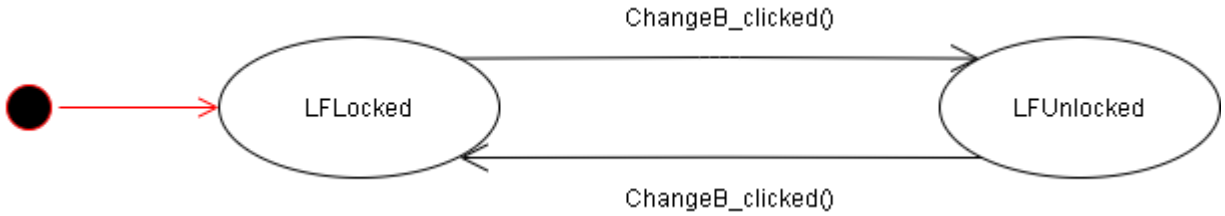
   Il metodo di LoginForm che utilizza l'intefaccia IProxy e permette di scegliere se evitare il controllo delle credenziali (e dunque l'utilizzo del Proxy) è il seguente:

```
private void LoginB_clicked(object sender, EventArgs e)
{
    IProxy loadDB;

    if (CurrentState is LFLocked)   //SI Proxy
    {
        //test credenziali
        loadDB = new Proxy(this);
    }
    else    //LFUnlocked => NO Proxy
    {
        loadDB = this;
    }

    loadDB.CreaManagerForm(registro.defaultLogin);  //NB implementazione in Proxy.cs
}
```

3. Comportamentale per oggetti: State
   Possiamo schematizzare la possibilità di accedere alternativamente ai database tramite credenziali o con account di default fornendo alla LoginForm uno stato interno. Possiamo dunque modellare la seguente macchina a stati finiti sulla quale progettare uno state pattern per gestire lo stato interno della form.



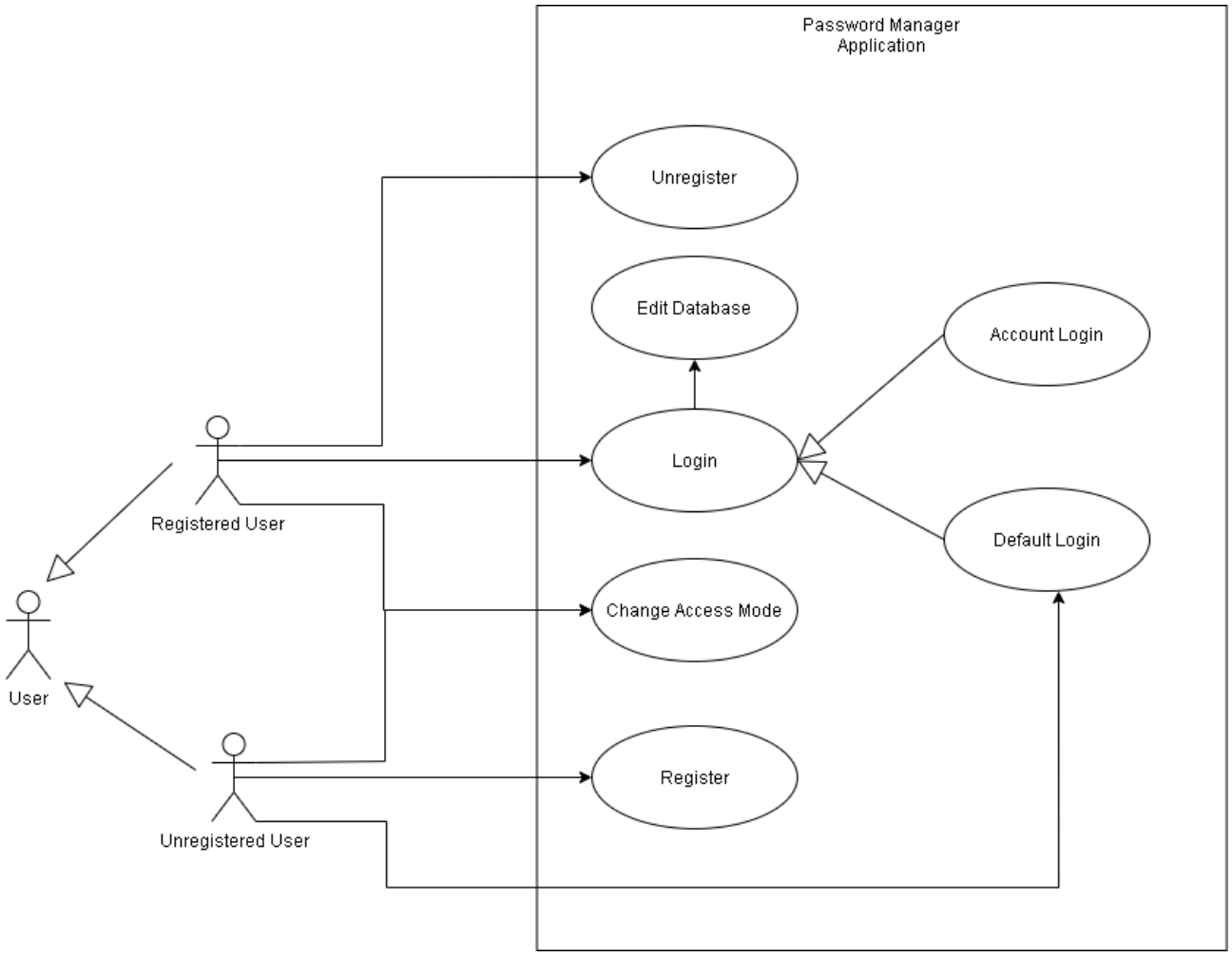
   Lo stato iniziale sarà LFLocked ovvero accesso con credenziali, e si potrà cambiare modalità (LFUnlocked), per accedere con account di default, tramite il bottone “Change” della LoginForm. Tale bottone infatti modificherà l’attributo CurrentState della form che corrisponde al suo stato interno e che gestisce la modalità di accesso tramite il metodo LoginB_clicked() sopra riportato. Tale pattern è dunque costituito dall’interfaccia ILoginFormState e i due possibili stati della form (LFLocked, LFUnlocked) che la implementano.

   Considerazioni finali:
   Lo state pattern può essere omesso nel caso in cui non si ritenga necessario aggiungere in futuro degli altri stati alla LoginForm, in questo caso però il software non sarebbe *Future-Proof*. Equivalentemente si può dunque utilizzare un attributo booleano nella LoginForm da trattare come stato interno (ad es. true equivale a LFLocked e false a LFUnlocked) in sostituzione all’utilizzo del pattern.

4) Documentazione sull'utilizzo:

- L'applicativo è stato progettato tramite il framework grafico Windows Forms e Json.NET (inclusione Newtonsoft.Json).
- Inoltre per la crittografia si utilizza implicitamente la DPAPI (Data Protection API) disponibile solo su sistemi Windows.
- Il software è stato realizzato per il framework .NET core 3.1.
- Non sono necessari particolari parametri per l'esecuzione o la compilazione della soluzione.

5) Use cases UML:



L’utente che utilizza il software può essere di due tipi: registrato oppure non registrato. L’utente non registrato può registrarsi, cambiare modalità di accesso e fare login esclusivamente con l’account di default. L’utente registrato invece può deregistrarsi, cambiare modalità di accesso e fare login con le proprie credenziali o tramite account di default. Una volta fatto login si può effettuare la modifica del database e tutte le altre operazioni analoghe permesse dall’applicativo. In generale il login può essere di due tipi: tramite credenziali o account di default (comune a tutti gli utenti).

Termine della relazione.