Progetto di Laboratorio di Programmazione
Social Network

Santolo Barretta - N86003666
Giovanni D'Ecclesiis - N86003669

31/05/2021

Indice

1 Breve descrizione del progetto

Si richiede di programmare un social network minimalista che implementi le classiche funzioni di base di un social, tra cui: **sign up**, **log in**, **delete account**. Se l'utente ha effettuato il login correttamente, può scegliere di eseguire le sueguenti operazioni: **visualizzare gli username di tutti gli utenti**, **cercare se esiste un utente**, **seguire un utente**, **fare un post**, **vedere tutti i post di un altro utente**, **vedere l'ultimo post di ogni utente seguito**, **fare il log out**. Ogni utente ha la possibilità di visualizzare solo dieci post nel proprio account. All'avvio del programma il social network viene inizializzato con cinque account predefiniti, ognuno dei quali contenente già dieci post.

2 Dettagli implementativi

2.1 L'approccio alla soluzione

Per arrivare alla soluzione è stato utilizzato un approccio gruaduale, scomponendo il lavoro da svolgere in più fasi: quella di pianificazione ed analisi del sistema, progettazione, codifica, test ed infine verifica e collaudo del programma realizzato. In primo luogo, una volta letta la traccia del programma da implementare, c'è stata una fase iniziale di pianificazione ed analisi del sistema, nella quale è stata analizzata in modo dettagliato la realtà di riferimento (un social network minimalista). Sono state specificate tutte le funzionalità software da implementare in modo dettagliato, attraverso un attento processo di astrazione che ha permesso di concentrarsi sugli aspetti più rilevanti e realizzabili della realtà di riferimento, trascurando tutti i dettagli inutili.

Successivamente, nella fase di progettazione, sulla base dei requisiti individuati e dettagliati attraverso l'attività di analisi, è stata individuata e definita - a nostro avviso - la strategia più efficace ed efficiente che avrebbe portato alla soluzione e al completamento del progetto assegnato.

E' stato inizialmente progettato un algoritmo di funzionamento generale del programma con le sue varie funzionalità attraverso l'uso di uno pseudocodice. L'approccio utilizzato è stato di tipo **top-down**. Il progetto da implementare è stato scomposto in più moduli (funzioni) da risolvere e successivamente implementare, che unite fra loro permettono la realizzazione di tutte le funzionalità richieste per il social network. Sono state poi scelte le strutture dati da utilizzare (che verranno disccusse in seguito) e la struttura generale del programma.

Per la realizzazione del social network è stato scelto di implementare un sistema di menù di scelta utente, nel quale sono presenti tutte le varie opzioni disponibili. Ciò è stato possibile utilizzando il costrutto di selezione multipla **switch-case** del linguaggio di programmazione C. All'avvio del programma all'utente viene mostrato un menù di scelta iniziale nel quale è possibile scegliere fra:

- **SIGNUP**: funzione che effettua la registrazione dell'utente con la conseguente creazione di un nuovo account nel social network attraverso l'inserimento di un nome utente (username) e una password. Se si sceglie un username già registrato si riceverà un messaggio di errore. Consiste nell'inserimento in coda del nuovo nodo utente nella linked list implementata.
- **LOGIN**: funzione che effettua il login dell'utente dati in ingresso l'username e la password. Effettua quindi una ricerca dell'account dell'utente nel social network: se l'account è esistente avviene il login, altrimenti viene mostrato un messaggio di errore. Consiste in una ricerca sequenziale del nodo utente nella linked list implementata.
- **EXIT**: opzione che permette di uscire e terminare il programma.

Per entrare nel proprio account l'utente deve effettuare il login. Una volta effettuato il login all'utente verrà mostrato un ulteriore menù di scelta nel quale sono implementate tutte le ulteriori funzionalità che offre il social network. L'utente può scegliere le seguenti opzioni:

- **VISUALIZZA USERNAME DI TUTTI GLI UTENTI**: funzione che visualizza l'username di tutti gli utenti registrati sul social network.

Consiste in una visualizzazione del campo 'username' dei nodi utente della linked list implementata.

- **CERCA UTENTE**: funzione che dato un username permette di ricercare un utente sul social network. Se l'utente viene trovato viene mostrato un messaggio in cui si dice che l'utente esiste. In caso contrario viene mostrato un messaggio in cui si avvisa che l'utente non è registrato sul social. Consiste in una ricerca sequenziale del nodo utente nella linked list implementata.

- **SEGUI UTENTE**: funzione che dato un username permette di seguire un altro utente sul social. Se l'utente prova a seguire se stesso verrà mostrato un messaggio di errore; lo stesso se l'utente tenta di seguire un utente che non esiste. Se invece l'utente da seguire esiste viene mostrato un messaggio in cui si dice che l'utente è stato seguito correttamente. Consiste nell'aggiunta della stringa relativa all'username dell'utente da seguire in coda nell'array dei seguiti del nodo utente della linked list implementata.

- **VISUALIZZA TUTTI I POST DI UN UTENTE**: funzione che dato l'username di un utente, permette di visualizzare i suoi post. Se l'utente non ha scritto alcun post verrà mostrato un avviso, nel caso contrario saranno mostrati tutti i post partendo da quello più recente fino a quello più vecchio. Consiste in una visualizzazione dell'array di stringhe 'post' del nodo utente ricercato della linked list implementata.

- **VISUALIZZA ULTIMO POST DEGLI UTENTI SEGUITI**: funzione che permette di visualizzare l'ultimo post di ogni utente seguito dall'utente loggato. Ciò è possibile solo se l'utente ha seguito almeno un utente, in caso contrario sarà mostrato un messaggio di errore poichè l'utente non segue altri utenti. Inoltre se qualche utente seguito non ha creato alcun post, allora si mostrerà un messaggio in cui si dice che l'utente non ha alcun post da mostrare. Se l'utente seguito invece ha creato già dei post, allora la funzione mostra il post più recente, quello in ultima posizione nell'array dei post.

- **CREA POST**: funzione che permette di creare un post all'utente loggato. Inizialmente si fa inserire il post all'utente in una stringa di appoggio, si effettuano diversi controlli sulla validità del post scritto, dopodiché si aggiunge la stringa corrispondente al post in coda nell'array dei post del nodo utente della linked list implementata. Inoltre la funzione tiene conto del caso in cui l'utente ha raggiunto il massimo numero di post (10); in questo caso per creare un nuovo post bisogna cancellare il post più vecchio dell'utente: per farlo si applica uno shift dell'array verso sinistra di una posizione, cancellando quindi il primo post più vecchio dopodiché si aggiunge in ultima posizione dell'array il nuovo post appena creato dall'utente. Infine se l'utente non ha scritto ancora nessun post si aggiunge il nuovo post creato in prima posizione nell'array dei post del nodo utente corrente.

- **VISUALIZZA IL TUO ACCOUNT**: funzione aggiuntiva che permette di visualizzare l'account dell'utente loggato, nella quale si mostra la lista

degli utenti seguiti, i suoi post creati (richiamando la funzione visualizzapostutente) e gli ultimi post degli utenti seguiti (richiamando la funzione visualizzaultimopostseguiti).

- **DELETE ACCOUNT**: funzione che permette di cancellare l'account dell'utente loggato. Prima di cancellare l'account all'utente viene chiesto se è sicuro di farlo. In caso affermativo si procede alla cancellazione dell'account e alla disconnessione, riportando l'utente alla schermata principale del programma in cui potrà effettuare una nuova registrazione. In caso contrario l'account non viene cancellato e l'utente potrà scegliere nuovamente un'opzione. Consiste nella cancellazione di un nodo utente dalla linked list implementata.

- **LOG OUT**: disconnette l'utente loggato dall'account e lo riporta alla schermata iniziale, nella quale può nuovamente effettuare un signup o un login.

Tutte le funzioni descritte, una volta implementate, sono state opportunamente richiamate nei corrispondenti **case** degli **switch** inseriti.

Prototipi delle funzioni implementate. *Per ulteriori dettagli sul funzionamento generale di ogni funzione si faccia riferimento alla documentazione interna del progetto (commenti).*

```
1  void signup(struct nodo* head,char *username,char *password);
2  int login(struct nodo* head, char *username, char *password);
3  void deleteaccount(struct nodo* head, char *username);
4  void visualizzausernameutenti(struct nodo* u);
5  int ricercautente(struct nodo* head, char *username);
6  void seguiutente(struct nodo *head, char *username, char *
       usernamelog);
7  void creapost(struct nodo* head, char *username);
8  void visualizzapostutente(struct nodo* head, char *username);
9  void visualizzaultimopostseguiti(struct nodo* head,char *username);
10 void visualizzaprofilo(struct nodo* head, char *username);
```

2.2 La scelta delle strutture dati

Per quanto riguarda la scelta delle strutture dati, si è giunti alla conclusione che un generico **utente** del social network può essere descritto attraverso:

- un **username**: una stringa (di massimo 20 caratteri nel nostro caso);
- una **password**: una stringa (di massimo 8 caratteri nel nostro caso);
- i suoi **post**: un array di stringhe (di massimo 10 stringhe in quanto l'utente può scrivere massimo dieci post);
- i suoi **seguiti**: un array di stringhe nel quale sono contenuti gli username degli utenti seguiti;

Dunque abbiamo diversi tipi di dato (la maggior parte di tipo **char**). Questo ci ha suggerito inevitabilmente di accorpare tutti i dati in una struttra dati di tipo **record** che ci permette di trattare informazioni di tipo diverso relative ad un stesso oggetto preso in esame. Come è noto, nel linguaggio di programmazione C

i record vengono implementati attraverso le strutture (**struct**), cioè un insieme di variabili di uno o più tipi, raggruppate da un nome in comune. Dunque è stata definita una **struct Utente** che contiene tutte le informazioni relative ad un singolo utente.

```
typedef struct
{
  char username[MAXU+1];
  char password[MAXP+1];
  char post[MAXPOST][MAXLENPOST];
  int numpost;
  char seguiti[MAXF][MAXU+1];
  int utentiseguiti;
}Utente;
```

Un social network, però, può definirsi tale se è composto da un insieme di utenti. Siccome non si è a conoscenza di quanti utenti si registreranno nel social network, si è pensato di implementare una struttura dati dinamica: la **linked list**. Essa è estremamente utile poiché ha una natura dinamica e potrà aumentare se un utente decide di registrarsi sul social network, con l'aggiunta di un nuovo nodo utente o diminuire se si cancella un account, cancellando il rispettivo nodo utente.

E' noto che un nodo della linked list può contenere dati di ogni tipo, comprese struct. Per questo la linked list implementata definisce un generico **nodo** utente mediante un'istanza della **struct Utente** definita in precedenza che permette di accedere alle informazioni dell'utente e un **puntatore** al nodo successivo della linked list, in quanto autoreferenziante.

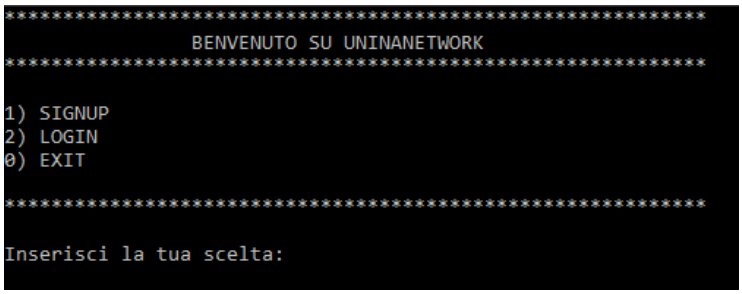
```
struct nodo
{
  Utente u;
  struct nodo *next;
};
```

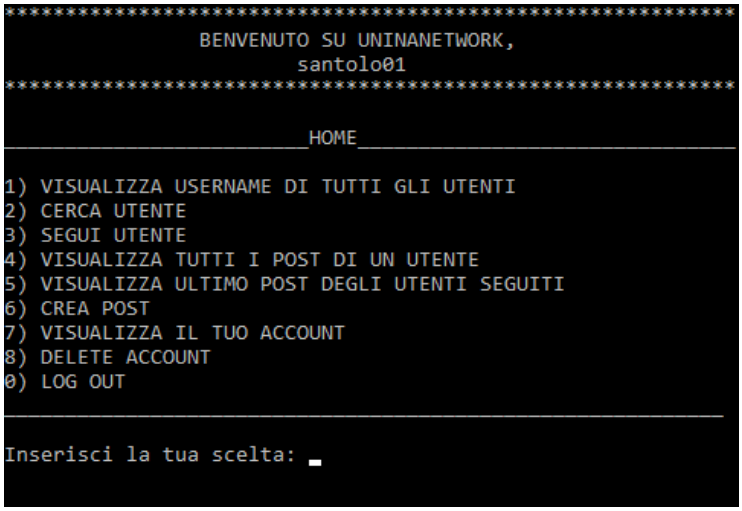
Come da traccia all'inizio sono stati dichiarati 5 utenti predefiniti, ossia cinque nodi iniziali della linked list collegati fra di loro, di cui il primo è l'**head** della linked list e l'ultimo punta a **NULL**. Ognuno dei cinque nodi utente iniziali, corrispondenti a cinque account utente, è stato settato con 10 post predefiniti, un username, una password e dei seguiti.

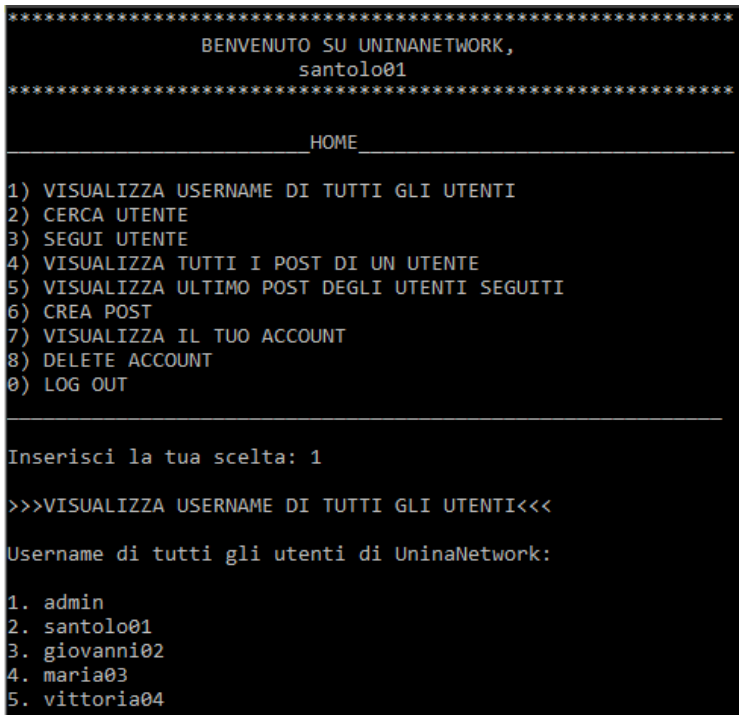
L'implementazione della linked list ha agevolato la creazione delle funzionalità del social network in quanto sono stati implementati degli algoritmi noti riguardanti le linked list quali la ricerca di un nodo nella lista, l'inserimento di un nodo in coda, la cancellazione di un nodo ecc.

2.3 Esempio di esecuzione e casi d'uso del programma

Fig.1: Schermata iniziale.
Fig.2: Schermata di un account utente loggato.
Fig.3: Un caso d'uso.


BENVENUTO SU UNINANETWORK
1) SIGNUP
2) LOGIN
0) EXIT
Inserisci la tua scelta:


BENVENUTO SU UNINANETWORK,
santolo01
HOME
1) VISUALIZZA USERNAME DI TUTTI GLI UTENTI
2) CERCA UTENTE
3) SEGUI UTENTE
4) VISUALIZZA TUTTI I POST DI UN UTENTE
5) VISUALIZZA ULTIMO POST DEGLI UTENTI SEGUITI
6) CREA POST
7) VISUALIZZA IL TUO ACCOUNT
8) DELETE ACCOUNT
0) LOG OUT
Inserisci la tua scelta:


BENVENUTO SU UNINANETWORK,
santolo01
HOME
1) VISUALIZZA USERNAME DI TUTTI GLI UTENTI
2) CERCA UTENTE
3) SEGUI UTENTE
4) VISUALIZZA TUTTI I POST DI UN UTENTE
5) VISUALIZZA ULTIMO POST DEGLI UTENTI SEGUITI
6) CREA POST
7) VISUALIZZA IL TUO ACCOUNT
8) DELETE ACCOUNT
0) LOG OUT
Inserisci la tua scelta: 1
>>>VISUALIZZA USERNAME DI TUTTI GLI UTENTI<<<
Username di tutti gli utenti di UninaNetwork:
1. admin
2. santolo01
3. giovanni02
4. maria03
5. vittoria04