**RELAZIONE DI SISTEMI E RETI – PROGETTO DI FINE ANNO**

**Michele Bellardi - 4IB**

**Descrizione:**Applicazione Client – Server per la consultazione da remoto di informazioni relative alle stazioni ferroviarie presenti in un file CSV.

**Obbiettivo:**Realizzare un’applicazione Client-Server in linguaggio Java, che consenta agli utenti di consultare in remoto le informazioni di interesse relative a Stazioni ferroviarie presenti in un file formato CSV.  
Il nome del file: [Mappa-delle-stazioni-ferroviarie-in-Italia.csv](http://www.datiopen.it/it/opendata/Mappa_delle_stazioni_ferroviarie_in_Italia?t=Tabella).

Il client deve vedere stampato un menù contenente varie opzioni di ricerca che dovranno includere:

* Ricerca delle stazioni per *comune*;
* Ricerca delle stazioni per *nome*;
* Ricerca delle stazioni per *anno di inserimento*;
* Ricerca delle stazioni per *coordinate*;
* Ricerca delle stazioni per *identificatore in OpenStreetMap*;
* Ricerca delle stazioni per *riga*.

**Descrizione dell’esercizio:**

1. Progettare e implementare un server che:

* Legga e interpreti un file CSV, organizzando i dati in una struttura interna (ad esempio, una lista di oggetti o una mappa).
* Offra un'interfaccia di comunicazione remota tramite socket per ricevere richieste e inviare risposte al client.
* Gestisca in modo robusto eventuali errori, come richieste non valide o malformate.

1. ﻿﻿﻿Progettare e implementare un'applicazione client che:

* Permetta all'utente di inviare richieste specifiche al server (ad esempio, richiedere una riga o colonna specifica del file CSV, effettuare ricerche o visualizzare tutto il contenuto).
* Mostri in modo chiaro e leggibile i dati ricevuti dal server.

1. Il protocollo di comunicazione tra client e server deve seguire un formato semplice e ben definito e documentato nella relazione allegata al progetto. Ad esempio:

* Richiesta dal client: 'GET\_ROW n' (per ottenere la riga n del file).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid row\*).
* Richiesta dal client: 'GET\_MUNICIPALITY c' (per ottenere le stazioni del comune indicato).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid municipality\*).
* Richiesta dal client: 'GET\_NAME n' (per ottenere le stazioni con il nome indicato).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid row\*).
* Richiesta dal client: 'GET\_YEAR y' (per ottenere le stazioni inserite nell’anno indicato).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid year\*).
* Richiesta dal client: 'GET\_COORDINATES cs' (per ottenere le stazioni nelle coordinate indicate).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid coordinates\*).
* Richiesta dal client: ‘GET\_INDICATOR i' (per ottenere le stazioni con l’indicatore indicato).  
  Risposta dal server: i dati corrispondenti, oppure un messaggio di errore (es. 'ERROR: Invalid identificator\*).

**Struttura del progetto:**

Il progetto prevede un’applicazione Client – Server che comunicano tramite il protocollo TCP, il Server può gestire più connessioni alla volta e il Client dispone di interfaccia grafica.

**Funzionamento:**

Il Server dispone del file CSV contenente le stazioni ferroviarie e le loro informazioni, legge il file e inserisce tutte le informazioni all’interno di un ArrayList.

Client e server comunicano scambiando stringhe.

**Specifiche:**

Il Server riceve dal Client una stringa contenente il comando da eseguire, una volta eseguito il comando viene inviata una stringa dal Server al Client contenente le informazioni richieste e verrà stampata sull’interfaccia grafica del Client.

È possibile anche utilizzare una connessione da terminale come client e inserire a mano i comandi.