ELABORATO TELEMATICA

RIEPILOGO PROGETTO

DATA RELAZIONE	NOME PROGETTO	PREPARATO DA
23 febbraio 2019	Web crawler (spider)	Giuseppe La Gualano 265681

DESCRIZIONE (REQUISITI)

Spider

1) **Inizializzazione**

- Lo spider legge il file "ConfigDB.txt" (se non presente, lo crea) contenente i parametri di connessione al database che possono essere eventualmente modificati dall'utente al lancio del software o modificando il file.

- Vengono chiesti ad input utente uno o più URL e TAG (parole) iniziali.

È possibile lanciare più istanze di questo software con inizializzazioni anche differenti, lavorano parallelamente senza conflitti.

2) **Interazione ed indice Database**

- Lo **schema logico** del database è basato su un'unica tabella:
  SITI(URL, HTTP Version, Robots.txt, Mobile-Friendly, Rank)

  URL è chiave primaria, assicura che non vengano inseriti duplicati.
  HTTP Version indica la versione protocollo http (2.0 / 1.1 / 1.0)
  Robots.txt indica la presenza del percorso /robots.txt
  Mobile-Friendly indica la presenza del meta tag "viewport"
  Rank indica il valore* del sito

  *Il valore è un numero intero che va da 0 a 3 e viene attribuito in base alle colonne precedenti secondo questa pesatura:
  HTTP 2.0 = 1 punto, HTTP 1.1 = 0,5 punti, HTTP 1.0 = zero punti.
  Robots.txt presente = 1 punto
  Viewport presente = 1 punto

  Questo sistema ha lo scopo di mostrare all'utente, in ordine crescente o decrescente a scelta sul sito, un riassunto dei dati raccolti ed avere come primi risultati i siti più moderni ed aggiornati piuttosto che altri.

- **Inserimento in DB**

  Lo spider analizza il sorgente di ogni pagina visitata, oltre che effettuare **http get request** per la versione http, ma inserisce nel database il singolo URL filtrato **netloc** (per ottenere il Root Domain pulito).

  - Le pagine vengono rivisitate solo se si esegue un altro spider **parallelamente** o si riesegue il corrente.

  Lo spider organizza gli URL in una **lista** che viene ampliata ad ogni aggiunta di un nuovo URL trovato all'interno della pagina corrente soggetta allo **scraping**.

  - Per ogni iterazione viene effettuato un controllo sugli URL già presenti quindi non vengono inseriti in lista **duplicati**.

  Lo spider lavora fino ad **interruzione** forzata dall'utente (oppure fino a quando ha finito di scorrere tutta la lista il che è impossibile per il numero di URL che vengono aggiunti ad ogni iterazione)

  Vengono inseriti SOLO i siti che al loro interno contengono una o più parole tra quelle inserite in fase di inizializzazione. Le **parole** sono ricercate in ogni elemento del sito.

  - La scelta è voluta in quanto le **meta-keyword** non sono più presenti su molti siti in quanto i motori di ricerca più rinomati da alcuni anni ignorano questo tag per finalità **SEO** e controllano l'intero documento, basandosi specialmente su titoli **H1** ed **H2**, ma per correttezza il controllo è effettuato su ogni parte di testo del sito.

3) **Accesso e Gestione dati estratti**

  User-Agent: Mozilla/5.0

  HTTPConnection prova connessione con porta **443**, in caso di fallimento esclude la presenza di HTTP/2 e riesegue la connessione esplicitando HTTP/1.1 o 1.0.

  Redirect 301/302 per il caso **non-www/www** viene automaticamente gestito. Per i casi di redirect ad altri domini differenti, vengono analizzati in automatico in quanto la lettura html viene effettuata successivamente.

  Status Code (4xx o 5xx) vengono gestiti con minima latenza portando lo spider all'elemento successivo in lista url.
  In questo modo viene tenuta traccia di pagine non funzionanti così da non rianalizzarle fino ad esplicito inserimento utente.

4) Esecuzione Spider

- Lo spider è munito di **GUI a console** (terminale), che alla sua esecuzione parte con l'inizializzazione dei parametri sopra citati (chiedendo opportunamente all'utente quelli mancanti) e mostra a video le iterazioni: nome del sito, versione http e se trovati o meno i tag (le parole inizializzate), il robots.txt ed il meta tag viewport.

- Il software si presenta in modalità **Stand Alone Executable** per sistemi operativi **UNIX-like**, ma può essere creato anche l'exe anche per sistemi Microsoft Windows.

In questo modo non è necessario installarlo ma basta solo aprire il file così da non richiedere la presenza di python (e librerie utilizzate) sul computer su cui si vuole eseguire lo spider.

5) Website

- **XAMPP** viene utilizzato per visualizzare da locale (quindi il sito non è pubblico per comodità di accesso e gestione del database) la pagina web creata appositamente per mostrare gli elementi del database.

- **Bootstrap** è stato utilizzato per la creazione della pagina web che è mobile-friendly, per cui può essere visualizzata da dispositivi mobili con design responsive.

  Il suo **CSS** e **JS** viene utilizzato per avere una pagina con un template moderno completo di tabella munita di ordinamenti (crescenti e decrescenti, basati sugli elementi già letti senza interrogare nuovamente il database), animazioni di menu laterali a scomparsa e impaginazione delle righe per evitare che la pagina sia troppo lunga e lenta.

SVOLGIMENTO (STRUMENTI DI SVILUPPO)		
STRUMENTO	DETTAGLIO	NOTA
Computer	MacBook Pro	Indifferente, vedere punto 4
Sistema Operativo	OS X El Capitan	Indifferente, vedere punto 4
Librerie	v. 2018.3.1	urllib.parse, urllib.request, requests, hyper, re, mysql.connector
XAMPP	v. 7.3.0	Apache + MariaDB + PHP

LINGUAGGI UTILIZZATI

NOME	VERSIONE	USO
Python	3.7.1	Spider
SQL (MySQL)	MariaDB, 10.1.37	Connessione Spider e piattaforma web
PHP	7.3.0	Visualizzazione dati su piattaforma web
HTML	5	“ “
CSS	3	Implementazione Boostrap 4.1.3 mobile-friendly
JavaScript	1.8.5	Piccole animazioni di transizione elementi in sito

Altro

Per lo sviluppo del software si era inizialmente partiti da **Scrapy** con l’intenzione di rendere eseguibile lo spider (mediante servizi es. **scrapinghub**) direttamente da **cloud** piuttosto che con software su macchina fisica.

Successivamente la prova con **Beautifulsoup** risultava essere con funzioni troppo lente e che facevano fatica a gestire molte eccezioni per vari errori durante l’esecuzione che deve risultare ininterrotta.

Si è preferito partire da più librerie leggere (citate in strumenti di sviluppo, es **hyper** per http get request) apposite ad ogni requisito del software presentato, molto versatili e compatibili tra loro.

Per la creazione dello Stand Alone Executable si è utilizzato **PyInstaller**, quindi se lanciato da macchina con windows, andrà a crearne il suo eseguibile.

Il sito web si è preferito non mettere online in quanto gli **hosting provider** gratuiti (es. altervista) bloccano l’accesso da remoto al database, mentre altri non gratuiti (es. netsons) generano più volte errori di time-out in connessione.

La creazione del file “**ConfigDB.txt**” è appositamente voluta in quanto è abitudine di piccoli software analoghi avere un file di configurazione della connessione al database e non mostrare in codice eventuali user e password. Questo comportamento non è ripetuto per l’inserimento degli url e dei tag che sono opportunamente chiesti all’utente come input.

La **finalità** di questo **spider di indicizzazione** non è tanto la raccolta di interi documenti ma è basata sulla **SERP** come se l’utente scrivesse su un motore di ricerca una parola e venissero mostrati i nomi dei siti opportunamente posizionati dal più completo e moderno a quello meno ma che in ogni caso sia pertinente alla ricerca.