**PRESENTAZIONE PROGETTO - SOFTWARE ENGINEERING**

**L' applicazione sviluppata dal team indifferente è una semplice applicazione Client-Server.**

**La nostra User-Interface è stata sviluppata principalmente in HTML5 e come si può vedere è semplice da capire e molto facile da utilizzare.**

**Le funzionalità principali che abbiamo individuato sono Creazione, Caricamento, Cancellazione di un albero e Calcolo della somma degli attributi.**

**La priorità è stata data alla creazione e al caricamento, infatti sono i due pulsanti presenti nella prima schermata, dove il pulsante "Crea" ci porta a generare un nuovo albero ed il pulsante "Lista" ci porta alla lista degli alberi già presenti.**

**Si sono utilizzate delle tecnologie di programmazione chiamate JSP rappresentanti contenuti dinamici. Esse si basano su un insieme di speciali tag, all'interno di una pagina HTML, con cui possono essere invocate funzioni predefinite sotto forma di codice JAVA e/o JAVASCRIPT.**

**Nel contesto della piattaforma JAVA, questa tecnologia è correlata con quella delle servlet, che appunto abbiamo utilizzato per lo scambio di informazioni.**

**Nella nostra applicazione sono state scartate alcune scelte come l'utilizzo di una struttura dati per la creazione di un albero.**

**Si sono fatte delle scelte per il passaggio da un linguaggio di programmazione compilato PHP ad un vero e proprio linguaggio di programmazione JAVA, poiché il PHP riscontrava problemi nella gestione di 100 utenti in contemporanea; Si è scelto JAVA semplicemente perché tutto il team aveva delle buone conoscenze sulle sue funzionalità.**

**Dopo un primo studio si era pensato di utilizzare come struttura dati un array, una struttura efficiente ma occupava molta RAM; Si è trovata una soluzione, ovvero quella di non utilizzare una struttura dati, ma di memorizzare l'albero a run-time.**

**Il calcolo della somma viene fatto sul server.**

**Qui si è costruito un algoritmo (ricorsivo) in grado di calcolare a run-time il percorso/path.**

**(Quindi per calcolare il padre abbiamo utilizzato una semplice regola che trova l'ultimo nodo di un certo livello, % splitsize - 1)**

**La concorrenza è stata gestita in fase di creazione e in fase di cancellazione.**

**Se un utente crea un nuovo albero, crea una collections con il nome dell'albero, se un secondo utente cerca di effettuare una creazione con il medesimo nome, allora fallisce.**

**In fase di cancellazione si è permesso a più utenti di effettuare la somma in contemporanea, mentre l'operazione di cancellazione è possibile solo se nessuno sta operando su quell'albero; Per questo tipo di problematiche, il team, ha introdotto l'utilizzo dei semafori per garantire la mutua esclusione.**

**E' stato anche deciso di non creare una classe arco ma di inserire tali proprietà nella classe nodo, che infatti, contiene sia gli attributi dei nodi che quelli degli archi.**

**Per quanto riguarda il DataBase si sono fatte delle scelte molto importanti.**

**Il database utilizzato è MongoDB, un database NOSQL famoso per la sua scalabilità orizzontale, in quanto avevamo bisogno di un database in grado di memorizzare grandi quantità di dati.**

**E' stato scartato il database MYSQL perché non si riuscivano a gestire la quantità di dati e si doveva ricorrere alla paginazione sulle tabelle.**