Documentazione progetto TPSIT

# Descrizione del progetto

Lo scopo principale è aiutare gli utenti che desiderano avere un acquario a scegliere le piante giuste per loro da usare nel loro aquascape.

Si vuole realizzare un progetto java che utilizzi un web-service locale con metodo di comunicazione mediante XML e interpretazione dei dati tramite la libreria JAXB.  
Lo scopo è quello di realizzare un web-service mediante **mysql** che dia accesso a tutte le piante per acquario con le relative informazioni (nome, link pagina, link immagine, tipo, origine, tasso crescita, altezza minima, altezza massima, luce, co2, link logo, difficoltà). L’app java ha il compito di avere la parte client che deve permettere la ricerca (per nome, filtrata di vario tipo) delle piante, accedendo al web service e facendosi inviare tramite XML i dati che poi verranno riportati come oggetti da java tramite JAXB, e poi verranno fatti visualizzare all’utente.

# Architettura del sistema

**Descrizione dell'architettura complessiva:** Il sistema è composto da due componenti principali: un web service RESTful in PHP e un client Java. Il web service ospita un database MySQL di informazioni sulle piante d'acquario.   
  
Il client Java ha un’interfaccia collegata al web service per permettere agli utenti di:

* Ricercare le piante per nome, tipo, origine, tasso di crescita, luminosità richiesta, necessità di CO2 e altri criteri.
* Visualizzare i dettagli di una specifica pianta, tra cui immagini, nome, continente di origine, dimensioni, valori di temperatura ideali.
* Salvare le proprie ricerche e creare una lista di piante preferite (Acquario).

**Componenti chiave del sistema:**

**Web service:**

* Sviluppato in PHP con Apache Web Server.
* Database MySQL che memorizza informazioni su:
  + centinaia di specie di piante d'acquario.
  + immagini.
  + link a siti web con informazioni più approfondite.
* API RESTful per la ricerca, la ricezione e la modifica dei dati sulle piante.

**Client Java:**

* Libreria JAXB per la mappatura XML-Java con creazione delle classi tramite xsd e libreria xjc.
* Interfaccia utente grafica intuitiva per:
  + Inserire i criteri di ricerca.
  + Visualizzare i risultati della ricerca in una tabella con filtri e ordinamento.
  + Visualizzare i dettagli di una specifica pianta con immagini e informazioni complete.
  + Salvare le ricerche e le piante preferite.

**Interazioni tra i componenti:**

1. L'utente interagisce con l'interfaccia del client Java.
2. Il client Java invia una richiesta al web service (PHP) tramite l'API RESTful locale.
3. Il web service (PHP) elabora la richiesta, si connette al database MySQL e recupera i dati.
4. Il web service (PHP) costruisce la risposta in formato XML seguendo l’XSD definito.
5. Il client Java utilizza JAXB per mappare i dati XML in oggetti Java.
6. Il client Java visualizza i dati all'utente.

**Diagramma dell'architettura:**

Immagine che contiene testo, diagramma, disegno, design

Descrizione generata automaticamenteDiagramma dell'architettura del sistema:

# 3. Descrizione del Web-service (PHP)

**Scopo:** Fornire un servizio RESTful per la richiesta alle informazioni sulle piante d'acquario.

**Funzionalità:**

* Gestione delle richieste HTTP GET per la ricerca e il recupero dei dati sulle piante.
* Connessione al database MySQL per l'interrogazione dei dati.
* Generazione dinamica di XML seguendo lo schema XSD (plants.xsd) per la risposta alle richieste.
* Restituzione di piante per nome, tipo, origine, tasso di crescita, luminosità richiesta, necessità di CO2 e altri criteri.
* Recupero dei dettagli di una specifica pianta, tra cui immagini, nome scientifico, famiglia, continente di origine, dimensioni, valori di pH e temperatura ideali, consigli per la cura e la propagazione.
* Filtraggio e ordinamento dei risultati della ricerca.
* Paginazione dei risultati della ricerca.

**Parametri di input:**

* **Ricerca per nome:**
  + nome: il nome della pianta da ricercare.
* **Ricerca con filtri avanzati:**
  + tipo: il tipo di pianta (es: erba, anubias, ecc.).
  + origine: l'origine della pianta (es: America, Asia, Africa, ecc.).
  + tasso\_crescita: il tasso di crescita della pianta (es: lento, medio, veloce).
  + luce: la quantità di luce richiesta dalla pianta (es: bassa, media, alta).
  + co2: la necessità di CO2 per la crescita della pianta (es: no, si).
  + ph\_minimo: il valore minimo di pH tollerato dalla pianta.
  + ph\_massimo: il valore massimo di pH tollerato dalla pianta.
  + temperatura\_minima: la temperatura minima tollerata dalla pianta.
  + temperatura\_massima: la temperatura massima tollerata dalla pianta.

# 4. Requisiti del Progetto

**Funzionalità:**

* Il sistema deve permettere agli utenti di:
  + Ricercare le piante per nome, tipo, origine, tasso di crescita, luminosità richiesta, necessità di CO2 e altri criteri.
  + Visualizzare i dettagli di una specifica pianta, tra cui immagini, nome scientifico, famiglia, continente di origine, dimensioni, valori di pH e temperatura ideali, consigli per la cura e la propagazione.
  + Salvare le proprie ricerche e creare una lista di piante preferite.
* Il sistema deve essere in grado di gestire un elevato numero di utenti e di richieste, Il sistema deve essere sicuro e protetto da accessi non autorizzati (implementazione futura).

Immagine che contiene cerchio, disegno, schizzo, diagramma

Descrizione generata automaticamenteDiagramma dei casi d’uso dell’applicazione client:

**Prestazioni:**

* Il sistema deve essere in grado di rispondere alle richieste in modo rapido e efficiente.
* Il tempo di caricamento delle pagine deve essere minimo.

# 5. Implementazione

**5.1 Linguaggi di programmazione utilizzati:**

* Java 8 🡪 eclipse v. 2023-09
* PHP 7.4
* SQL

**5.2 Framework o librerie utilizzate:**

* Libreria Swing per l'interfaccia utente Java
* Libreria jgoodies-forms per la creazione di form Java
* Libreria miglayout per il layout dell'interfaccia utente Java
* Libreria PDO per la connessione al database MySQL in PHP
* Libreria Slim per lo sviluppo di API RESTful in PHP

**5.3 Descrizione delle principali fasi di sviluppo:**

* Analisi dei requisiti: Definizione delle funzionalità dell'applicazione e delle caratteristiche desiderate.
* Progettazione: Creazione di diagrammi UML per illustrare le classi e le loro relazioni.
* Implementazione: Sviluppo del codice sorgente per le diverse classi e funzionalità in Java e PHP.
* Test: Esecuzione di test unitari e di integrazione per la verifica del corretto funzionamento.
* Debug: Correzione di eventuali errori e bug trovati durante i test.
* Deploy: Distribuzione dell'applicazione su un ambiente di produzione.

**5.4 Struttura del codice sorgente:**

Lato Java:

Il codice sorgente è organizzato in pacchetti Java:

piante: Contiene le classi che modellano le informazioni sulle piante d'acquario.

view: Contiene la classe Window che rappresenta l'interfaccia utente.

control: Contiene la classe Controller che gestisce l'interazione con l'utente.

Lato PHP:

Il codice sorgente è organizzato in file PHP:

//da fare

**5.5 Descrizione delle scelte architetturali o di design:**

* Separazione tra logica e interfaccia utente: Il codice è stato strutturato per separare la logica dell'applicazione dall'interfaccia utente, rendendo il codice più modulare e manutenibile.
* Utilizzo di un design MVC (Model-View-Controller): Il design MVC è stato adottato per migliorare la separazione delle preoccupazioni e la modularità del codice.
* Sviluppo di un'API RESTful: Un'API RESTful è stata sviluppata in PHP per permettere al client Java di comunicare con il database.
* Utilizzo di librerie per la sicurezza: Librerie come PDO e Slim sono state utilizzate per garantire la sicurezza del database e dell'API.