4OURSQUARED

4oursquared.unipd@gmail.com

Lumos Minima

Imola Informatica

Motivazione delle Scelte

Informazioni	
Redattori	Ceccato Francesco Soldà Matteo
Versione	0.0.0
Uso	esterno

Descrizione

Questo documento riporta i confronti tra i vari linguaggi presi in considerazione per lo sviluppo del progetto e la motivazione della scelta di uno rispetto agli altri.

Versione	Data	Redattore	Verificatore	Descrizione
0.0.0	09/05/2023	Soldà Matteo	Ceccato Francesco	Prima stesura.

Contents

1 Introduzione

Il seguente documento ha come scopo principale quello di confrontare i vari linguaggi, database e framework con lo scopo di fornire un prodotto quanto più efficace, efficiente e aderente alle richieste del proponente.

1.1 Sitografia

- https://www.quora.com/Why-are-there-so-many-backend-systems-written-in-dynamically-typed-languages-while-the-benefit-of-static-typing-is-obvious;
- https://mobisoftinfotech.com/resources/blog/typescript-vs-javascript/;
- https://stackoverflow.com/questions/49640121/mixing-javascript-and-typescript-in-node-js;
- https://www.monocubed.com/blog/top-python-frameworks/;
- https://www.theserverside.com/tip/The-differences-between-Java-and-TypeScript-devs-must-know;
- https://medium.com/rewrite-tech/embedding-typescript-in-java-a343576031be;
- https://blog.geekandjob.com/framework-javascript/.

2 Confronti

2.1 Database

Come database, non sapendo al momento come saranno strutturati i dati che necessitano di essere salvati in maniera persistente, abbiamo preso in considerazione MariaDB per i database SQL e MongoDB per i database NoSQL.
Queste due scelte sono state fatte per due motivi principali:

- Ricerca di un database computazionalmente leggero e che fosse aderente agli standard;
- Ricerca di un database la cui community è attiva e la documentazione chiara e puntuale, così da poter risolvere quanto prima i problemi che si potessero presentare data la poca esperienza accumulata con i database nel tempo.

2.2 Backend

Per quanto riguarda il backend, la scelta sarà vincolata dalla preferenza verso un linguaggio staticamente o dinamicamente tipizzato.
Qualora si scegliesse la tipizzazione statica, le scelte ricadono su:

- Java;
- C#;
- TypeScript.

Mentre, scegliendo la tipizzazione dinamica, le scelte ricadrebbero su:

- Python;
- JavaScript.

2.2.1 C#

Pro:

- Facile da eseguire su dispositivi Windows (che rappresenta la stragrande maggioranza degli utenti);
- Essendo un linguaggio compilato e non interpretato, rende difficile in caso di attacchi la lettura del codice sorgente in chiaro.

Contro:

- Deve essere ricompilato per ogni minima modifica;
- Hostabile solo su server Windows.

2.2.2 Java

Pro:

- Linguaggio facile da imparare;
- Linguaggio di alto livello (assenza di puntatori, presenza di garbage collector);
- Linguaggio OOP;

- *Platform Indipendent*;
- Supporto al *multithreading*$_G$.

Contro:

- Rispetto ad altri linguaggi può risultare più lento o povero in fatto di performance;
- Mancanza di feature per il backup dei dati;
- Rispetto ad altri linguaggi, richiede uno spazio di memoria significativo;
- Rispetto ad altri linguaggi, usa molto più codice per svolgere una operazione basilare.

2.2.3 JavaScript

Pro:

- Semplice da imparare;
- Interoperabile;
- Versatile in quanto linguaggio *fullstack*$_G$.

Contro:

- Linguaggio interpretato;
- Tipizzazione dinamica;
- Inconsistenza tra browser.

2.2.4 Python

Pro:

- Facile da imparare;
- Flessibile ed estensibile;
- Presenza di moltissime librerie;
- Estremamente scalabile;
- Portabile;
- Vasta integrazione con l'IoT_G.

Contro:

- Forte utilizzo di memoria;
- Tipizzazione dinamica;
- Multithreading complesso;
- Garbace collection che porta ad un potenziale spreco di memoria.

2.2.5 Typescript

Pro:

- Linguaggio compilato;
- Tipizzazione statica.

Contro:

- Inconsistenza tra browser;
- Tipizzazione alcune volte complessa;
- Necessità di ricompilare per ogni minima modifica.

2.3 Backend Framework

2.3.1 C#

- ASP.NET.

2.3.2 Java

- Java Spring.

2.3.3 JavaScript

- Next;
- Node.

2.3.4 Python

- Django;
- Flask.

2.3.5 Typescript

- Nest;
- Feathers;
- Loopback.

2.4 Frontend

Escludendo *HTML5*, *XHTML* e *CSS* che saranno di sicuro usati per la parte di frontend della webapp, i linguaggi tra cui scegliere risultano essere solamente

- TypeScript;
- JavaScript.

2.4.1 JavaScript

2.4.2 TypeScript

2.5 Frontend Framework

2.5.1 JavaScript

- Angular;
- React;

- Vue;
- Backbone;
- Preact;
- Express.

2.5.2 TypeScript

- Angular;
- Ember.

Per i pro e i contro dei framework, sia frontend che backend, si rimanda il lettore a questo link.

3 Scelte e Motivazioni

[WIP]