roundabout

Roundabout - Etherless

Studio di fattibilità

Versione	0.0.4
Approvazione	
Redazione	Egon Galvani
	Antonio Zlatkovski
Verifica	
Stato	Non approvato
Uso	Interno
Destinato a	Roundabout
	Prof. Tullio Vardanega
	Prof. Riccardo Cardin

Descrizione
Studio di fattibilità dei capitolati proposti.

team.roundabout.13@gmail.com

Diario delle modifche

Versione	Data	Nominativo	Ruolo	Descrizione
0.0.4	2020-03-13	Egon Galvani	*Analista*	Stesura dello studio di fattibilità del capitolato C3.
0.0.3	2020-03-12	Egon Galvani	*Analista*	Stesura dello studio di fattibilità del capitolato C1.
0.0.2	2020-03-12	Egon Galvani	*Analista*	Stesura dello studio di fattibilità del capitolato C2.
0.0.1	2020-03-12	Egon Galvani	*Analista*	Creazione documento LaTeX e struttura generale.

Indice

1 Introduzione **3**
1.1 Scopo del documento: . . . 3
1.2 Glossario . . . 3
1.3 Riferimenti . . . 3
1.3.1 Riferimenti normativi . . . 3
1.3.2 Riferimenti informativi . . . 3

2 Capitolato scelto C2 - *Etherless* **4**
2.1 Informazioni generali . . . 4
2.2 Descrizione . . . 4
2.3 Obiettivo finale . . . 4
2.4 Tecnologie coinvolte . . . 4
2.5 Aspetti positivi . . . 5
2.6 Criticità . . . 5
2.7 Esito . . . 5

3 Valutazione capitolati rimanenti **6**
3.1 Capitolato C1 - Autonomous Highlights Platform . . . 6
3.1.1 Informazioni generali . . . 6
3.1.2 Descrizione . . . 6
3.1.3 Obiettivo finale . . . 6
3.1.4 Tecnologie coinvolte . . . 6
3.1.5 Aspetti positivi . . . 7
3.1.6 Criticità . . . 7
3.1.7 Esito . . . 7
3.2 Capitolato C3 - NaturalAPI . . . 8
3.2.1 Informazioni generali . . . 8
3.2.2 Descrizione . . . 8
3.2.3 Obiettivo finale . . . 8
3.2.4 Tecnologie coinvolte . . . 8
3.2.5 Aspetti positivi . . . 8
3.2.6 Criticità . . . 9
3.2.7 Esito . . . 9

1 Introduzione

1.1 Scopo del documento:

Elencare le motivazioni, i punti positivi e negativi considerati, che hanno portato alla scelta del capitolato C2 denominato *Etherless*, e alla consecutiva esclusione degli altri capitolati disponibili.

1.2 Glossario

I termini tecnici utilizzati in questo documento sono ulteriormente approfonditi nel Glossario denominato "Glossario Interno 1.0.0", disponibile in allegato al presente documento.

1.3 Riferimenti

1.3.1 Riferimenti normativi

- **Norme di Progetto**: *Norme di Progetto v1.0.0.*

1.3.2 Riferimenti informativi

- **Capitolato d'appalto C1: Autonomous Highlights Platform**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C1.pdf;
- **Capitolato d'appalto C2: Etherless**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C2.pdf;
- **Capitolato d'appalto C3: NaturalAPI**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C3.pdf;
- **Capitolato d'appalto C4: Predire in Grafana**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C4.pdf;
- **Capitolato d'appalto C5: Stalker**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C5.pdf;
- **Capitolato d'appalto C6: ThiReMa - Things Relationship Management**
  https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C6.pdf.

2 Capitolato scelto C2 - *Etherless*

2.1 Informazioni generali

- **Nome:** *Etherless*;
- **Proponente:** *Red Babel*;
- **Committente:** Prof. Tullio Vardanega e Prof. Riccardo Cardin.

2.2 Descrizione

Etherless è una piattaforma cloud che permette agli sviluppatori di fare il deploy di funzioni javascript nel cloud e far pagare gli utenti finali per la relativa esecuzione tramite l'utilizzo di smart-contract.

2.3 Obiettivo finale

La struttura richiesta per Etherless si compone di 3 moduli principali:

- **Etherless-cli:** è il modulo con cui gli sviluppatori interagiscono con Etherless. Supporta diversi comandi, tramite i quali lo sviluppatore può:
  - configurare il proprio account;
  - eseguire il deploy di funzioni javascript;
  - visualizzare la lista delle funzioni di cui ha già svolto il deploy;
  - richiedere l'esecuzione di una funzione e visualizzarne il risultato;
  - visualizzare i log relativi all'esecuzione di una specifica funzione;
- **Etherless-smart:** un insieme di Ethereum smart-contract che gestiscono la comunicazione tra *Etherless-cli* e *Etherless-server* e i pagamenti in ETH necessari per l'esecuzione delle funzioni;
- **Etherless-server:** si occupa di ascoltare gli eventi emessi da *Etherless-smart* e di far eseguire le funzioni così richieste. I risultati ottenuti vengono trasmessi tramite un ulteriore evento nella blockchain e mostrati all'utente attraverso *Etherless-cli*;

2.4 Tecnologie coinvolte

- **Ethereum:** piattaforma che permette ai suoi utenti di scrivere applicazioni decentralizzate (dette DAPP) che usano la tecnologia blockchain;
- **Ethereum Virtual Machine (EVM):** macchina virtuale distribuita sulla rete Ethereum;
- **Solidity:** linguaggio usato per la codifica degli smart-contract;
- **Serverless Framework:** framework per la realizzazione di applicazioni che vengono eseguite in architetture serverless (come AWS Lambda, Google Cloud Functions, Microsoft Azure Functions);
- **Typescript 3.6:** Linguaggio che supporta JavaScript ES6, introducendo alcune funzionalità, tra cui la tipizzazione;

- **ESLint:** tool di analisi statica del codice e syntax checking;
- **AWS Lambda:** consente di eseguire codice in risposta a determinati eventi, senza dover effettuare il provisioning né gestire server.

2.5 Aspetti positivi

- La blockchain e le tecnologie serverless sono argomenti che hanno suscitato un notevole interesse negli ultimi anni; per questo la relativa conoscenza potrebbe essere positiva da un punto di vista curricolare;
- la conoscenza di TypeScript e JavaScript è molto richiesta, oltre ad essere la base di svariate librerie e framework;
- il tema principale del progetto è stato accolto con molto interesse dal gruppo, che si è dimostrato disponibile ad approfondire l'argomento.

2.6 Criticità

- L'azienda ha sede all'estero, quindi la comunicazione con i referenti sarà meno agevole rispetto ad aziende che si trovano nel territorio nazionale;
- il capitolato richiede l'utilizzo di tecnologie recenti, la cui documentazione potrebbe non essere pienamente esaustiva e il cui apprendimento richiederà una mole di studio non indifferente.

2.7 Esito

Il progetto è stato subito accolto caldamente dai componenti del gruppo, le sfide tecnologiche sono stimolanti e gli argomenti trattati sono di interesse. Con questo progetto tutti i componenti del team avranno l'opportunità di studiare un campo dell'informatica che all'università è poco trattato, potendo aggiungere al proprio bagaglio curricolare delle voci particolarmente interessanti.

3 Valutazione capitolati rimanenti

3.1 Capitolato C1 - Autonomous Highlights Platform

3.1.1 Informazioni generali

- **Nome:** Autonomous Highlights Platform;
- **Proponente:** Zero12;
- **Committente:** Prof. Tullio Vardanega e Prof. Riccardo Cardin.

3.1.2 Descrizione

Creare una piattaforma web che riceve in input dei video di eventi sportivi come una partita di calcio, F1, Moto GP e crea in autonomia un video di massimo 5 minuti contenente i suoi momenti salienti.

3.1.3 Obiettivo finale

Viene richiesto che la piattaforma sia dotata di un modello di machine learning in grado di identificare ogni momento importante dell'evento. Il flusso di generazione di tale video deve essere così strutturato:

- caricamento del video;
- identificazione dei momenti salienti;
- estrazione delle corrispondenti parti di video;
- generazione del video di sintesi.

3.1.4 Tecnologie coinvolte

- **Elastic Container Service o Elastic Kubernetes Service:** è un servizio di orchestrazione di contenitori altamente dimensionabile ad elevate prestazioni;
- **DynamoDB:** database NoSQL dalle alte performance ideale per la conservazione di tag o altre informazioni a supporto dell'applicativo;
- **AWS Transcode:** servizio gestito per la conversione ed elaborazione di diversi formati video;
- **Sage Maker:** servizio completamente gestito che copre l'intero flusso di lavoro dell'apprendimento automatico per etichettare e preparare i dati, scegliere un algoritmo, formare il modello, ottimizzarlo per la distribuzione, effettuare previsioni e intraprendere azioni;
- **AWS Rekognition video:** servizio di analisi video basato su apprendimento approfondito; è in grado di riconoscere i movimenti delle persone in un fotogramma e di riconoscere soggetti, volti, oggetti, celebrità e contenuti inappropriati.

3.1.5 Aspetti positivi

- Tutte le tecnologie coinvolte sono ben documentate;
- il proponente zero12 fornisce attività di formazione sulle principali tecnologie AWS e wire-frame dell'interfaccia della console web di analisi e controllo dello stato di elaborazione dei video per la generazione del video di highlights.

3.1.6 Criticità

- Il proponente non fornisce alcun data-set per effettuare il training del modello di machine learning;
- l'identificazione manuale di tutti i momenti principali di un evento sportivo è un'azione ripetitiva e costosa in termini di tempo.

3.1.7 Esito

Il capitolato non è risultato molto stimolante, in quanto lo sviluppo di alcune componenti sembra caratterizzato da attività ripetitive. Si è quindi deciso di non considerare questa proposta.

3.2 Capitolato C3 - NaturalAPI

3.2.1 Informazioni generali

- **Nome:** NaturalAPI;
- **Proponente:** teal.blue;
- **Committente:** Prof. Tullio Vardanega e Prof. Riccardo Cardin.

3.2.2 Descrizione

Creare un toolkit per restringere il divario presente tra le specifiche di progetto e le API, permettendo così ai programmatori di dedicarsi principalmente allo sviluppo di nuove funzionalità piuttosto che provare a replicare il modello aziendale.

3.2.3 Obiettivo finale

Il toolkit richiesto si dovrà comporre di tre parti:

- **NaturalAPI Discover:** si occupa di identificare ed estrarre da documenti di testo potenziali entità di business (oggetti/nomi), processi (azioni/verbi) e predicati. In questo modo viene creato il Business Domain Language (BDL);
- **NaturalAPI Design:** si occupa di creare un Business Application Language (BAL) a partire dai documenti in Gherkin;
- **NaturalAPI Develop:** converte il BAL in API scritte in uno dei linguaggi di programmazione e framework supportati; il tutto gestendo sia la creazione di nuove repository per il codice, sia la modifica di quelle già esistenti.

3.2.4 Tecnologie coinvolte

- **Natual language processing:** il toolkit utilizza tecniche di elaborazione del linguaggio naturale;
- **Gherkin:** lo standard principale per scrivere in linguaggio naturale specifiche di funzionalità;
- **Cucumber:** framework per eseguire test automatici per BDD (Behavior-driven development);
- **Swagger:** tool di generazione di codice che si basa su OpenAPI.

3.2.5 Aspetti positivi

- Non viene richiesta un'interfaccia grafica, lasciando quindi più tempo al team di dedicarsi allo sviluppo delle funzionalità richieste;
- il capitolato è molto dettagliato e descrive in maniera precisa il comportamento richiesto dal toolkit.

3.2.6 Criticità

- Il capitolato non ha suscitato molto interesse nel gruppo;
- gli argomenti trattati sono stati considerati complessi e di difficile comprensione.

3.2.7 Esito

Lo scopo del capitolato non è risultato molto stimolante. Si è quindi deciso di non considerare questa proposta.