FONDI STRUTTURALI E DI INVESTIMENTO EUROPEI 2014/2020

fondo sociale europeo

ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Report Finale "Laboratorio Integrato"

Gruppo 6, Cloud Fiesta

De lazzari, Dellera, Oglietti,
Murta, Cafasso, Carrieri

28 gennaio 2022

in collaborazione con:

UNIONE EUROPEA

REGIONE PIEMONTE

per una crescita intelligente, sostenibile ed inclusiva

www.regione.piemonte.it/europa2020

INIZIATIVA CO-FINANZIATA CON FSE

Indice

Capitolo 1

Introduzione

Riccardo Oglietti

1.1 Il Progetto

Il qui presente report ha lo scopo di illustrare lo svolgimento nella sua interezza lo svolgimento del progetto a opera del gruppo "Cloud Fiesta", il progetto e' stato commissionato dai docenti Blanchietti Andrea e Zimuel Enrico nell'ambito del corso *"Laboratorio Integrato"*.

Lo scopo del progetto e' quello di realizzare una piattaforma di *e-commerce* per conto di un azienda che si occupa di commercio al dettaglio, il sistema deve essere *scalabile* in maniera da poter limitare i costi a quanto strettamente necessrio e potersi mantenere aderente con le esigenze di crescita aziendale. Inoltre, e' essenziale che la piattaforma possa avere degli *standard di sicurezza elevati*, come ben sappiamo, durante i recenti anni si e' verificata un impennata dei crimini legati alla *Cybersecurity*, con un particolare aumento durante la corrente pandemia da COVID-19, come illustrato dall'Interpol. E' quindi fondamentale che un'applicazione che gestisce flussi di denaro sia quindi estremamente solida dal punto di vista della sicurezza informatica. Una seconda sezione del progetto, prevede che ogni gruppo si occupi di eseguire dei *penetration test* sul gruppo dall'*ID* successivo. Questo per simulare l'ingaggio di un azienda esterna allo scopo di testare la sicurezza di un prodotto prima di rilasciarlo effettivamente sul mercato, uno step di decisiva importanza che permettera' ai componenti di ogni gruppo di sperimentare le proprie conoscenze di sicurezza informatica all'interno di una situazione altamente realistica.

Vista la complessita' del progetto, e' stato scelto di realizzarlo tramite Team multidisciplinari, con componenti appartenenti ad due corsi afferenti agli indirizzi di *Cloud Specialist* e *ICT Security Specialist*. All'interno di questi due corsi sono presenti le competenze tecniche atte a svolgere il progetto commissionato, coprendo sia l'area di sicurezza e di architettura della rete interna, che quella di utilizzo delle piattaforme cloud che permettono di assicurare la scalabilita' necessaria all'azienda.

1.2 Il Team

Gli stutenti di entrambi i corsi sono stati divisi in sei differenti gruppi, composti da un totale di otto persone, il nostro gruppo, denominato "*Cloud Fiesta*" e' composto dai seguenti studenti:

- **Cafasso Giovanni**
- **Carrieri Riccardo**
- **De Lazzari Riccardo**
- **Dellera Lorenzo**
- **Murta Alessio**
- **Oglietti Riccardo**

Suddivisi rispettivamente all'interno dei due corsi come da tabella:

Cloud Specialist	ICT Security Specialist
Cafasso Giovanni	De Lazzari Riccardo
Carrieri Riccardo	Dellera Lorenzo
Murta Alessio	Oglietti Riccardo
Zuccarella	

Come consigliato dai docenti, abbiamo assegnato alcuni *ruoli* in grado di aiutarci con l'organizzazione delle mansioni e in genere della gestione del progetto, in particolare abbiamo individuato il ruolo di *Team Leader* e di e di *Co-Team Leader*, essi sono stati rispettivamente assegnati a *Oglietti Riccardo* e *Murta Alessio*. Abbiamo optato per assegnare queste due cariche ripartendole tra i due differenti corsi che compongono il gruppo in maniera da manternere un buon livello di equita' e rappresentanza per entrambe le anime del team. Da notare infine come lo studente "Zuccarella Andrea" si e' presentato a un singolo incontro durante la stesura dell'intero progetto, e non ha effettivamente contribuito in alcuna maniera allo svolgimento dello stesso.

Capitolo 2

Strumenti tecnico-organizzativi

Oglietti Riccardo

2.1 GANTT e cronoprogramma

Innanzitutto parlando di strumenti tecnico-organizzativi non possiamo che iniziare descrivendo il "*GANTT*. Strumento principe per l'organizzazione delle tempistiche, si tratta di una tabella a doppia entrata che permette di assegnare alcuni *task* ritenuti fondamentali a un membro e un momento nel quale realizzarlo.

Ecco una lista riassuntiva dei processi e degli *step* fondamentali che abbiamo individuato al fine della realizzazione ottimale del progetto, divisi in base al corso di afferenza dei destinatari:

- 1. Parsing file CSV
  2. Definizione struttura di rete
  3. Deploy infrastruttura
  4. Test di sicurezza
  5. Modfica struttura in base alle falle trovate
  6. Deploy struttura finale
  7. Stesura report
- 1. Brainstorming
  2. Test locali nopCommerce
  3. Revisione manuale file CSV
  4. Selezione architettura Cloud
  5. Installazione locale nopCommerce/ DB su due macchine
  6. Containerazziazione su distro linux
  7. Upload su Cloud Provider
  8. Calcolo dei prezzi dell'Hosting di tutto il progetto (macchine virtuali, storage, call)
  9. Stesura report economico

2.2 Strumenti di comunicazione

Durante il primo incontro uno dei principali punti che e' stato chiarito e' quello della *comunicazione*. E' infatti essenziale che in un gruppo di lavoro sia possibile gestire la comunicazione in maniera piu' efficente e inclusiva possibile, senza quindi escludere membri o affidarsi a piattaforme troppo lente o non organizzate.

La nostra scelta e' quindi ricaduta sulla piattaforma di messaggistica istantanea *Telegram*, grazie alla puntualita' delle opzioni di gestione di una *chat* di gruppo e' possibile *fissare* messaggi, creare sondaggi e inviare file di grandi dimensioni. Grazie a recenti aggiornamenti e' inoltre possibile effettuare videochiamate e condividere eventualmente il proprio desktop, una feature essenziale nel campo del lavoro collaborativo.

2.3 Organizzazione codice

Data la forte componente di scrittura software presente all'interno del progetto, abbiamo optato per l'utlizzo di una piattaforma di sviluppo collaborativo, in maniera da organizzare la stesura del codice nella maniera piu' semplice ed esaustiva possibile. In particolare ci siamo affidati al software *GIT* a opera di *Linus Torvalds*, creando un organizzazione sulla popolare piattaforma di proprieta' *Microsoft*, *GitHub*.

Sulla piattaforma ci siamo quindi premurati di creare immediatamente tre *repository* atti a contenere il lavoro da noi prodotto, in particolare essi sono:

1. Random_Script
2. Report
3. Report_Economy

Il repository numero 1, *Random_Script* e' atto al contenimento di una serie di programmi di piccola entita', come il *parser* che si e' occupato di scaricare le immagini dei prodotti da aggiungere successivamente al database dello store, o i *Dockerfile* che serviranno per *Deployare* l'infrastruttura in ambiente di produzione.

Per quanto riguarda *Report*, ossia il numero 2, si tratta dello spazio atto alla creazione del report finale, esso e' stato redatto tramite l'utilizzo del linguaggio LaTeX, argomento che sara' affrontato in dettaglio in seguito.

Infine, il *repository* 3, nominato come *Report_Economy*, e' atto ad accogliere i documenti e gli appunti che permetteranno la stesura di un preventivo attendibile dell'implementazione, come richiesto dai requisiti del progetto.

2.4 Strumenti di scrittura

Come accennato durante la precedente sezione, lo strumento principe che e' stato impiegato per la redazione della relazione di progetto e' stato li linguaggio LaTeX. Si tratta di un linguaggio in grado di produrre un testo correttamente formattato a partire da semicodice, in questo modo viene automatizzata la gestione di alcune importanti caratteristiche del documento finale, come per esempio le immagini, spesso punto di debolezza dei comuni software di videoscrittura.

Capitolo 3

Componenti e architettura

3.1 In generale

3.1.1 Microsoft Azure

Per descrivere l'architettura da noi ideata riteniamo importante descrivere il *provider* al quale abbiamo deciso di appoggiarci. La scelta e' ricaduta sullo strumento "*Microsoft Azure*", si tratta di un servizio di *Cloud Computing* offerto da *Microsoft*, in particolare esso offre servizi di *Platform as a Service*, *Software as a Service* e *Infrastructure as a Service*. In particolare e' stato scelto in quanto aderente alle nostre necessita' di gestione di macchine remote da parte di un gruppo organizzato, inoltre , *Microsoft* offre un bonus gratuito di credito da spendere sulla piattaforma per ogni persona che vi si registri come studente. Cio' in concomitanza con i prezzi in linea con il mercato, ci ha permesso di sperimentare senza rischiare di dilapidare denaro.

3.1.2 nopCommerce

Durante la nostra ricerca di una soluzione che ci permettesse di creare uno *store online* ci siamo imbattuti in *nopCommerce*, una tecnologia a opera di *nopSolutions*. Si tratta di una soluzione *libre* e *open source* che permette la creazione di *store online* mantanendo una discreta semplicita' di utilizzo, nonche una facile integrazione in ogni sistema grazie alla possibilita' di essere installato all'interno di *contatiner Docker*.

Nato nel 2008, sviluppo e supporto non si sono mai interrotti, l'ampia *community* che lo mantiene e lo sviluppa permette inoltre una facile risoluzione di eventuali problemi grazie all'ampia documentazione prodotta nel corso degli anni. Questo prodotto abbraccia le moderne tecnologie in ambito di sviluppo web e di sicurezza grazie al massiccio utilizzo di *ASP.NET Core 5* e all'utilizzo come database predefinito di *MySQL* fino alle ultime versioni stabili.

3.2 Architettura di rete

L'architettura di rete scelta e' basata sull'utilizzo di una singola macchina virtuale in *cloud* sulla sopracitata piattaforma *MS Azure*.

La macchina virtuale monta un sistema operativo *GNU/Linux Ubuntu Server 20.04 LTS*, e ha le seguenti caratteristiche:

Informazione	Metrica
Tipologia	Standard_D2s_v3
CPU	2
RAM	8 GB
Disco	30 GB SSD Premium con ridondanza locale
Sicurezza	Standard

Al suo interno sono poi presenti alcuni elementi aggiuntivi, che compongono la vera e propria infrastruttura.

Innanzitutto e' presente *Docker*, si tratta del piu' diffuso orchestratore di *container* diffuso sul mercato. I *container* sono "entita'" che al loro interno contengono un ambiente minimale con tutte le componenti necessarie a un applicativo per svolgere il suo funzionamento, comprese tutte le dipendenze di ognuna delle sue parti. Questa entita' si interfaccia poi con il *Docker Engine*, un software che si occupa di tradurre le richieste del container in chiamate al sistema operativo sottostante, in questo caso una distribuzione di *Ubuntu GNU/Linux*.

Da sottolineare poi la fondamentale presenza di *SSH*. Si tratta dell'implementazione dell'omonimo protocollo di connessione remota per sistemi operativi *UNIX like*. Esso permette di effettuare una connessione remota con una macchina tramite una coppia di credenziali oppure una chiave *RSA*, criptando il traffico in maniera da mantenere la riservatezza della comunicazione.

Infine, gli ultimi due applicativi che e' opportuno segnalare come parti fondamentali della topografia di rete sono *Uncomplicated FireWall* e *Fail2Ban*. Il primo, come intuibile dal nome, e' un *Firewall* atto a limitare le connessioni non autorizzate verso la macchina per il quale e' configurato. In particolare, questo firewall e' in realta' un *frontend* semplificato per *IP Tables*, probabilmente il piu' utilizzato firewall in ambiente *GNU/Linux*. Anche *Fail2Ban* si occupa di una funzione simile, in quanto la sua funzione prinicpale all'interno dell' architettura proposta e' quella di regolamentare e limitare l'accesso alle connessioni *SSH* per i soggetti non autorizzati.

3.3 Organizzazione container

Come accennato durante il precedente paragrafo, la nostra architettura e' organizzata tramite l'utilizzo di *container*. Questa metodologia di *deploy* e' stata scelta perche offre numerosi vantaggi rispetto alla piu' "classica" architettura basata sull'utlizzo di macchine virtuali dedicate. In particolare, una macchina e' in grado di gestire diversi *container* diversi, ottimizzando al meglio le risorse *hardware* a disposizione. Inoltre, il parziale isolamento di un *container* rispetto alla macchina *host* rende l'infrastruttura piu' sicura, in quanto compromettere un singolo applicativo non mette a rischio il resto dell'infrastruttura o degli altri servizi che condividono le medesime risorse. Infine e' importante sottolineare che, grazie ad alcuni moderni *software* di orchestrazione, come ad esempio *Docker swarm*, e' possibile gestire in maniera estremamente puntuale l'eventuale ridondanza dei servizi e la loro posizione fisica all'interno del *cluster* di

macchine virtuali. In questo modo e' possibile rendere un infrastruttura estrememanete resiliente, e' inoltre possibile effettuare operazioni di manutenzione con un impatto minimo sull'utente finale.

Per implementare la nostra infrastruttura, abbiamo deciso di utilizzare i seguenti *container*:

- nopCommerce
- MySQL

Il primo denominato come *nopCommerce* contiene l'effettiva struttura dello *store online*, compreso di tutte le componenti atte a pubblicare le pagine *web* sulla porta `8080`, tra cui i *CSS* e il *backend* scritto in *ASP.NET*.

Il secondo *contaier* invece, contiene un database *MySQL* che si occupera' di contenere tutte le informazioni riguardanti i prodotti, ecco alcuni dei campi presenti all'interno del *database*

1. ProductId
2. ProductType
3. Name
4. FullDescription
5. Vendor

I *container* sono configurati in maneira da utilizzare due porte per la comunicazione, esse sono la porta `80` e la porta `3306`. Rispettivamente usate per permettere al *container* contenente *nopCommerce* di essere esposto verso internet, e sempre al medesimo di effettuare le comunicazioni con il database *MySQL*.

Capitolo 4

Processo di implementazione

4.1 Implementazione locale

4.2 Implementazione remota