



Allegato Tecnico

Jawa Druids

Versione	1.0.0
Data approvazione	??-??-????
Responsabile	Nome Cognome
Redattori	Nome Cognome Nome Cognome
Verificatori	Nome Cognome Nome Cognome Nome Cognome
Stato	Approvato
Lista distribuzione	Jawa Druids Prof. Tullio Vardanega Prof. Riccardo Cardin
Uso	Esterno

Sommario

Il presente documento contiene le scelte architettrurali che il gruppo *Jawa Druids* ha effettuato ai fini realizzativi del progetto. Contiene i design pattern e i diagrammi di attività, sequenza, classi e package.



Indice

1	Introduzione	2
1.1	Scopo del documento	2
1.2	Scopo del prodotto	2
1.3	Glossario	2
2	Architettura del prodotto	3
2.1	Descrizione generale	3
2.2	Architettura Acquisition	3
2.3	Architettura Prediction	3
2.4	Architettura Web-App	3
3	Requisiti soddisfatti	4
3.1	Tabella requisiti funzionali	4
3.2	Grafici requisiti funzionali	6



1 Introduzione

1.1 Scopo del documento

Lo scopo del documento è quello di elencare e motivare le scelte architetturelle fatte dal gruppo Jawa Druids, per quanto riguarda il progetto GDP: Gathering Detection Platform.

1.2 Scopo del prodotto

In seguito alla pandemia del virus COVID-19 è nata l'esigenza di limitare il più possibile i contatti fra le persone, specialmente evitando la formazione di assembramenti. Il progetto *GDP: Gathering Detection Platform* di *Sync Lab* ha pertanto l'obiettivo di **creare una piattaforma in grado di rappresentare graficamente le zone potenzialmente a rischio di assembramento, al fine di prevenirlo**. Il prodotto finale è rivolto specificatamente agli organi amministrativi delle singole città, cosicché possano gestire al meglio i punti sensibili di affollamento, come piazze o siti turistici. Lo scopo che il software intende raggiungere non è solo quello della rappresentazione grafica real-time ma anche quella di poter riuscire a prevedere assembramenti in intervalli futuri di tempo.

Al tal fine il gruppo *Jawa Druids* si prefigge di sviluppare un prototipo software in grado di acquisire, monitorare ed analizzare i molteplici dati provenienti dai diversi sistemi e dispositivi, a scopo di identificare i possibili eventi che concorrono all'insorgere di variazioni di flussi di utenti. Il gruppo prevede inoltre lo sviluppo di un'applicazione web da interporre fra i dati elaborati e l'utente, per favorirne la consultazione.

1.3 Glossario

All'interno della documentazione viene fornito un *Glossario*, con l'obiettivo di assistere il lettore specificando il significato e contesto d'utilizzo di alcuni termini strettamente tecnici o ambigui, segnalati con una *G* a pedice.



2 Architettura del prodotto

2.1 Descrizione generale

In fase di progettazione, il gruppo *Jawa Druids* ha deciso di suddividere la modellazione architeturale di *Gathering-Detection-Platform* in tre distinti moduli, tutti indipendenti tra loro. Il primo modulo si occupa solamente di leggere, tramite file $JSON_G$, tutte le webcam disponibili per poi effettuare il riconoscimento persone tramite i frame scaricati. Successivamente i dati estrapolati verranno inviati al database. Il secondo modulo, il $machine-learning_G$, si occupa di recuperare questi dati dal database per lavorarli producendo predizioni per le ore future. Infine il terzo modulo, la $web-app_G$ vera e propria, si occuperà di rappresentare graficamente i dati all'interno del database mediante una $heat-map_G$ e farli visualizzare all'utente.

2.2 Architettura Acquisition

L'architettura riguardante il modulo di acquisizione, ovvero il primo modulo del software, è basata sul fatto che è creata sul paradigma della codifica procedurale. Inoltre non presenta alcuna classe in quanto non crea oggetti, crea esclusivamente un array con i dati che estrapola dai frame e dalle informazioni del tempo.

2.3 Architettura Prediction

L'architettura del modulo del $machine-learning$ si può semplificare ad un modulo unico con all'interno i metodi necessari per prelevare dati dal database per poi reinviarli da lavorati. Non necessita classi interne in quanto svolge esclusivamente operazioni funzionali

2.4 Architettura Web-App

Per il modulo relativo al $front-end_G$, si è deciso di utilizzare il pattern *Model-View-Controller*(MVC). Questa scelta è dovuta al fatto che, essendo la $web-app$ sviluppata con spring, il pattern è quello che più si adatta alla tipologia sia di modellazione sia di scopo.



3 Requisiti soddisfatti

In questo capitolo vengono illustrati attraverso grafici a torta e tabelle i requisiti funzionali che sono stati implementati all'interno della demo sviluppata per la *Revisione di qualifica*.

3.1 Tabella requisiti funzionali

Codice Requi	Fonte
RSFO1	Capitolato _G V. esterno 17-12-2020
RSFF2	Capitolato _G
RSFO3	UC2
RSFO4	Capitolato _G UC8
RSFO4.1	Capitolato _G UC8.1 UC8.2
RSFO4.2	Capitolato _G UC8.3
RSFO5	Capitolato _G
RSFD5.1	Capitolato _G
RSFD6	Capitolato _G
RSFO7	Capitolato _G UC1
RSFO8	Interno
RSFO9	Interno UC1
RSFO10	Interno UC1



RSFO11	Interno UC1
RSFF12	Interno
RSFD13	Interno UC10
RSFD14	Interno UC10
RSFF15	Interno UC11.1
RSFF16	Interno UC11.2
RSFO17	Interno
RSFO18	V. esterno 2-02-2021
RSFO18.1	V. esterno 2-02-2021
RSFO19	Interno UC9
RSFO20	Interno UC4
RSFO21	Interno
RSFO22	Interno
RSFO22.1	Interno
RSFO22.2	Interno
RSFF23	Interno
RSFO24	UC5.1
RSFO25	Interno
RSFO26	UC5.1
RSFO27	UC5.2
RSFO28	UC5.3
RSFD29	Interno
RSFO30	V. esterno 02-02-2021



RSFF31	Interno UC12
RSFO32	UC3
RSFO32.1	UC3.1
RSFO32.1.1	UC3.1.1
RSFO32.1.2	UC3.1.2
RSFO32.1.3	UC3.1.2
RSFO32.2	UC3.2
RSFD33	UC6
RSFD33.1	UC6.1
RSFD33.2	UC6.2
RSFD34	UC7
RSFD35	UC13
RSFD36	UC14
RSFD36.1	UC14.1
RSFD36.2	UC14.2
RSFD37	UC15
RSFD37.1	UC16
RSFD38	UC15
RSFD39	UC15
RSFD40	UC17
RSFD41	UC18

Tabella 3.1: Requisiti funzionali soddisfatti

3.2 Grafici requisiti funzionali