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 architetturali 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

Elenco delle tabelle

Elenco delle figure

1 Introduzione

1.1 Scopo del documento

Lo scopo del documento è quello di elencare e motivare le scelte architetturali 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 architetturale 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 invitati 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 molto semplice ed intuitiva. Non vi è alcuna classe e si basa su una programmazione procedurale. Nel *detect.py*, ovvero lo script principale del modulo, vengono richiamate le funzioni, in maniera sequenziale, per scaricare e manipolare i dati delle webcam. La scelta dell'utilizzo di un paradigma procedurale risiede nel fatto che la creazione di oggetti e il loro utilizzo risultavano, nell'insieme, più complicati mentre chiamando delle semplici funzioni esterne il programma risultava più leggibile ed efficiente. Gli unici oggetti presenti in *detect.py* sono quelli di tipo *data*, necessari per la giusta esecuzione dello script. Di seguito vengono riportati i diagrammi relativi all'*attività* e di *package*.

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 generare delle predizioni ed archiviarle nel database. Non necessita classi interne in quanto svolge esclusivamente operazioni funzionali. Per generare le predizioni il modulo svolge delle operazioni sequenziali:

1. prelevo i dati all'interno del database;
2. controllo dei dati per eliminare le informazioni che possono inquinare le predizioni;
3. creo un set di dati da predire;
4. definisco il modello della predizione;

5. alleno il modello con i dati prelevati e controllati;
6. utilizzo il modello allenato per ricavare le previsioni dal set di dati creato nel punto 2;
7. archivio il risultato nel database.

2.3.1 Diagrammi dei package

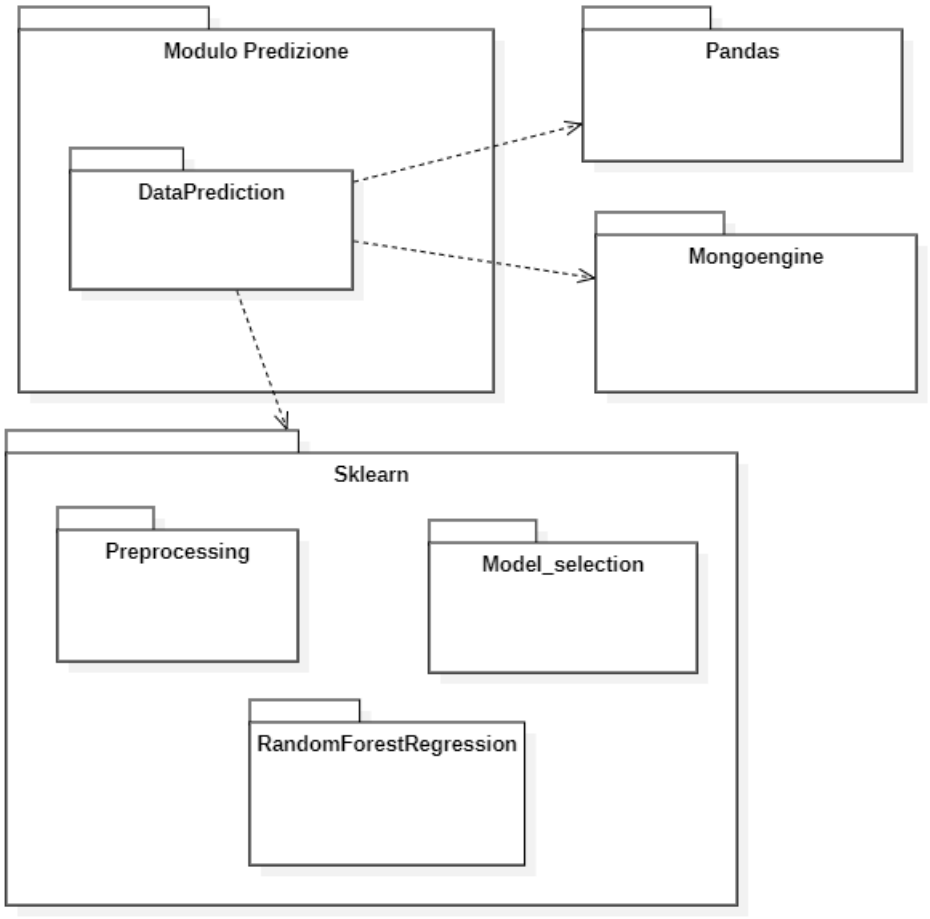

Figura 2.1: Diagramma dei package del modulo della predizione

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. Lo scambio di dati tra fron-end e back-end avviene attraverso il pattern architetturale REST. Si è deciso questo pattern per avere un oggetto di scambio tra le due parti unico strutturalmente e quindi non è essere vincolato dalla struttura presente nel database. Questo permette di aumentare la portabilità dell'applicazione web potendo applicare il back-end, possibilmente, a diversi front-end. Le richieste che il front-end effettua al back-end sono HTTP Request GET, quindi sono sempre richieste di visione di informazioni per poi essere utilizzate per aggiornare la parte grafica visibile dal client.

2.4.1 Diagrammi dei package

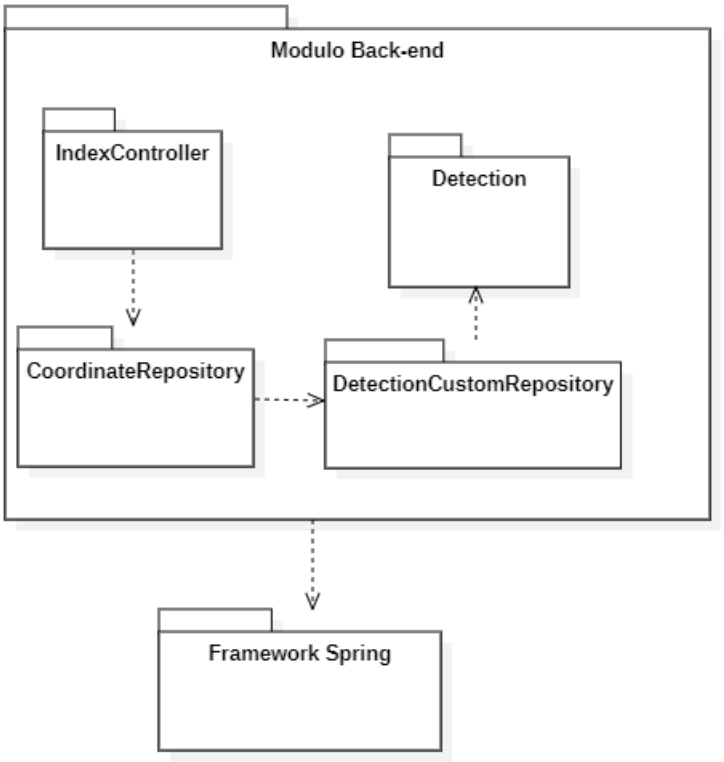

Figura 2.2: Diagramma dei package del modulo back-end

2.4.2 Diagrammi delle attività

In questo diagramma di attività viene illustrato la sequenza di operazioni dell'aggiornamento della heat-map a seguito di un cambiamento dei parametri della web-app da parte dell'utente.

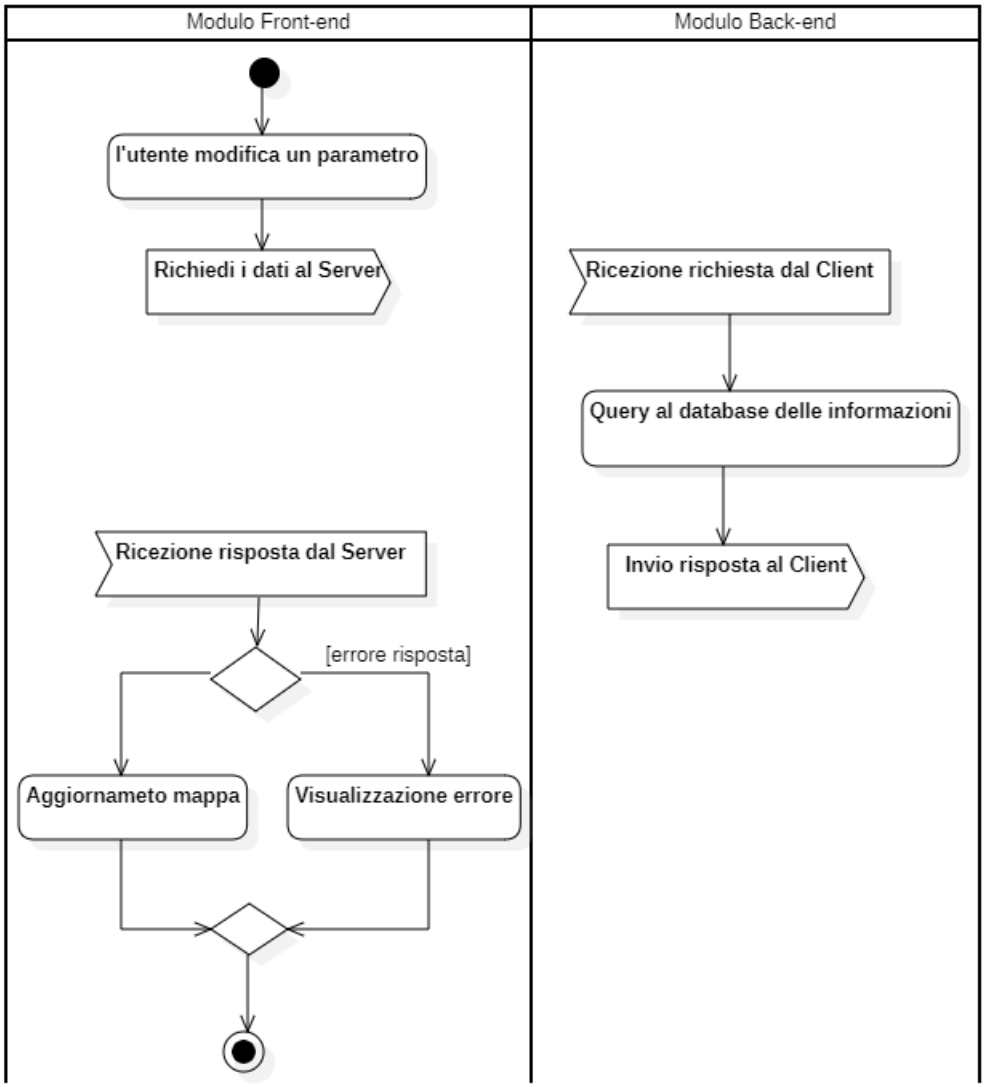

Figura 2.3: Diagramma di attività tra modulo front-end e modulo back-end

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*. Utilizzando la codifica descritta all'interno delle *Norme 3.0.0*

3.1 Tabella requisiti funzionali

Codice Requisito	Soddisfatto
RSFO1	Soddisfatto
RSFF2	Non soddisfatto
RSFO3	Soddisfatto
RSFO4	Soddisfatto
RSFO4.1	Soddisfatto
RSFO4.2	Soddisfatto
RSFO5	Soddisfatto
RSFD5.1	Non soddisfatto
RSFD6	Non soddisfatto
RSFO7	Soddisfatto
RSFO8	Non soddisfatto
RSFO9	Soddisfatto
RSFO10	Soddisfatto
RSFO11	Non soddisfatto
RSFF12	Non soddisfatto
RSFD13	Non soddisfatto
RSFD14	Non soddisfatto
RSFF15	Non soddisfatto
RSFF16	Non soddisfatto
RSFO17	Non soddisfatto

RSFO18	Soddisfatto
RSFO18.1	Soddisfatto
RSFO19	Soddisfatto
RSFO20	Soddisfatto
RSFO21	Soddisfatto
RSFO22	Soddisfatto
RSFO22.1	Soddisfatto
RSFO22.2	Soddisfatto
RSFF23	Non soddisfatto
RSFO24	Soddisfatto
RSFO25	Non soddisfatto
RSFO26	Non soddisfatto
RSFO27	Soddisfatto
RSFO28	Soddisfatto
RSFD29	Non soddisfatto
RSFO30	Soddisfatto
RSFF31	Non soddisfatto
RSFO32	Soddisfatto
RSFO32.1	Soddisfatto
RSFO32.1.1	Soddisfatto
RSFO32.1.2	Soddisfatto
RSFO32.1.3	Soddisfatto
RSFO32.2	Soddisfatto
RSFD33	Non soddisfatto
RSFD33.1	Soddisfatto
RSFD33.2	Non soddisfatto
RSFD34	Non soddisfatto

RSFD35	Non soddisfatto
RSFD36	Non soddisfatto
RSFD36.1	Non soddisfatto
RSFD36.2	Non soddisfatto
RSFD37	Non soddisfatto
RSFD37.1	Non soddisfatto
RSFD38	Non soddisfatto
RSFD39	Non soddisfatto
RSFD40	Non soddisfatto
RSFD41	Soddisfatto

Tabella 3.1: Requisiti funzionali soddisfatti

3.2 Grafici requisiti funzionali

Nel grafico seguente vengono visualizzati tutti i requisiti funzionali soddisfatti.

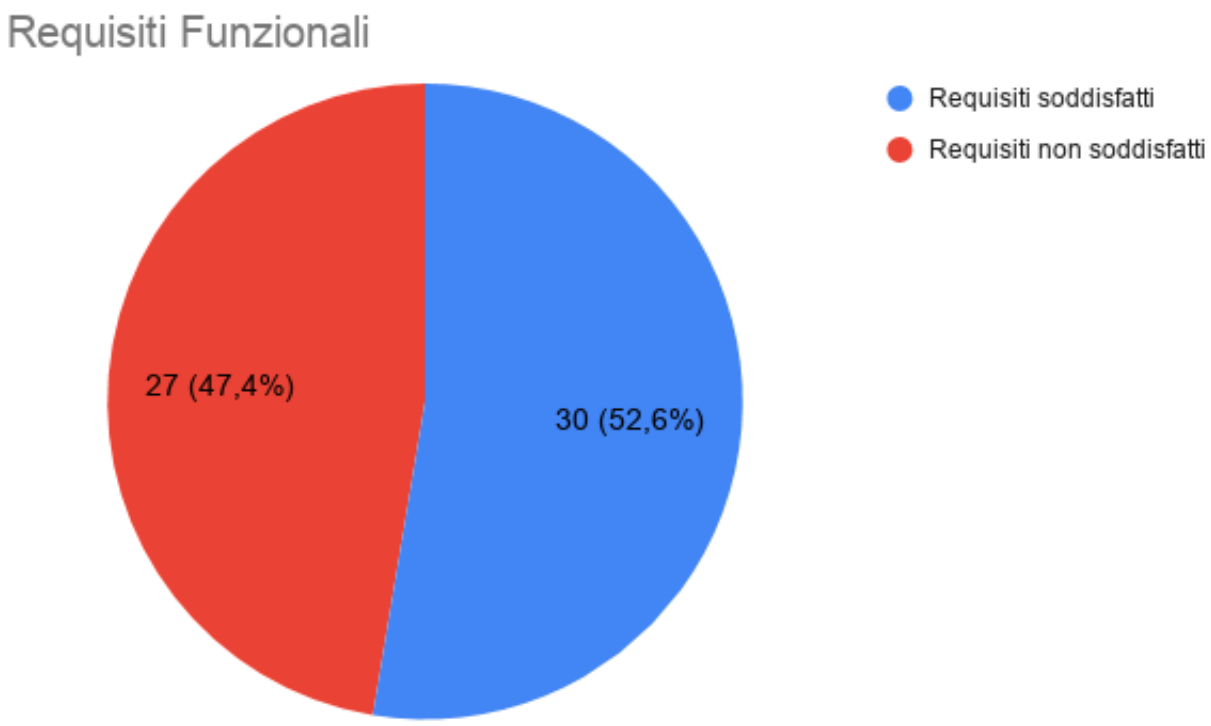

Figura 3.1: Diagramma a torta di tutti i requisiti funzionali soddisfatti

Nel grafico seguente vengono visualizzati tutti i requisiti funzionali obbligatori soddisfatti.

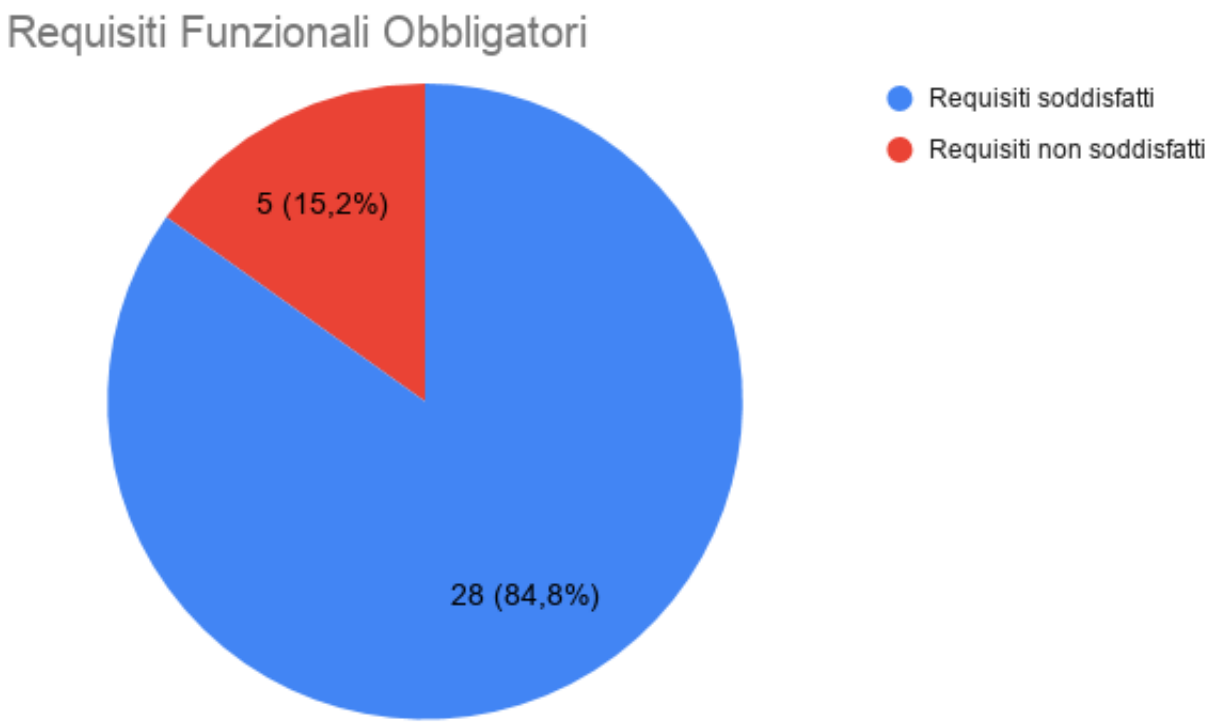

Figura 3.2: Diagramma a torta di tutti i requisiti funzionali obbligatori soddisfatti