Documento di progettazione

Rev 2.0 (28/02/2017)

Progetto Crimson

Ingegneria del Software (A.A. 2016/17)

Gruppo MALG
Cavallo Mattia (845900@stud.unive.it)
Ceolin Andrea (846559@stud.unive.it)
Naccari Laura (846196@stud.unive.it)
Patti Giulio (847665@stud.unive.it)

Sommario

1. Introduzione

Il gruppo MALG ha avanzato come proposta una “app” che permetterà all’utente finale di ottenere informazioni circa la sicurezza di determinate sostanze additive presenti negli alimenti e nei prodotti cosmetici, in particolare quelli acquistati in (o provenienti da) paesi extra-UE, facendo riferimento ai data-set pubblicati dall’Unione Europea relativi all’approvazione o all’applicazione di un divieto sull’utilizzo di questi ultimi.

Dal momento che la scelta del nome del progetto avverrà presumibilmente durante una fase successiva a quella dello sviluppo, ma comunque precedente a quella della pubblicazione, tutte le repository, la documentazione interna e il codice faranno uso del codename *Crimson* per riferirsi al progetto stesso.

Il fine del presente documento è quello di descrivere dettagliatamente le modalità di realizzazione dell’applicazione proposta. A tal proposito, verranno analizzati:

- Un modello rappresentante le componenti dell’infrastruttura e la loro (inter)connessione,
- L’organizzazione e la stratagia per il caricamento del dataset,
- Un mockup dell’interfaccia utente dell’app,
- La progettazione architetturale (diagramma delle classi per ogni singola componente),
- Modello relazionale del database interno.

Si tiene a far presente che, dal momento che lo sviluppo del progetto segue il modello evolutivo (le fasi di progettazione e di implementazione non avvengono in tempi distinti in maniera rigida), il presente documento potrà essere oggetto di numerose variazioni ed ampliamenti.

2. Glossario

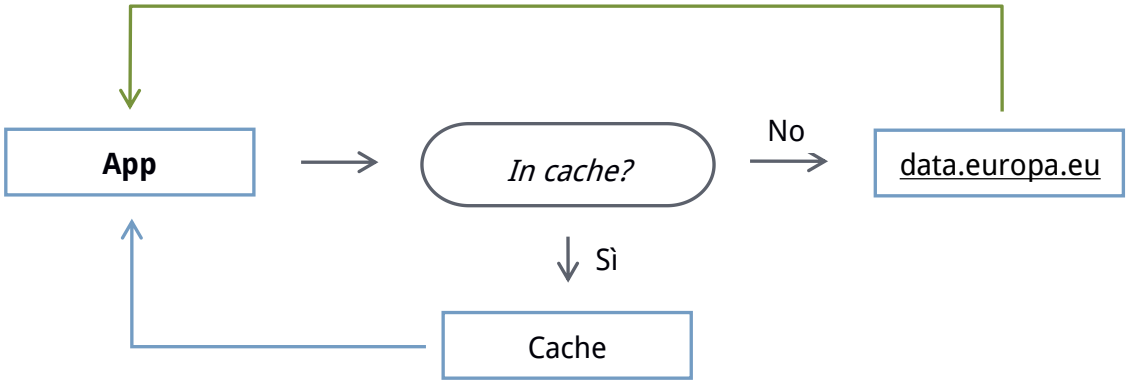
Nella stesura della presente documentazione, si fa uso dei seguenti termini e abbreviazioni:

- **Ambiente di sviluppo**: raccolta di strumenti destinati agli sviluppatori nell’ambito della produzione di progetti software.
- **Android**: sistema operativo per dispositivi mobili, in particolare smartphone e tablet.
- **App**: neologismo che sta per “applicazione (software)”. Solitamente, con questo termine, siriferisce ad applicazioni eseguibili da dispositivi mobili.
- **Cache**: memoria “nascosta” al programmatore che farà uso del componente che la implementa, finalizzata ad aumentare le prestazioni e migliorare l’usabilità evitando lo scaricamento e/o il ricalcolo della medesima informazione molteplici volte.
- **Code-name**: denominazione interna riferita ad un prodotto in fase prototipale.
- **Database**: archivio organizzato di dati, nel quale si definiscono relazioni, e si operano interrogazioni e manipolazioni.
- **Dataset**: insieme di dati statici (o aggiornati periodicamente in toto), organizzati in maniera analoga ad un *database*.
- **Diagrammi delle classi**: consentono di descrivere tipi di entità, con le loro caratteristiche e le eventuali relazioni fra questi tipi.
- **Diagrammi di sequenza**: descrivere uno scenario.
- **Diagrammi degli stati**: descrivono il comportamento dei sistemi, il quale viene analizzato e rappresentato tramite una serie di eventi che potrebbero accadere per ciascun stato.
- **Dispositivo**: meccanismo, congegno o elemento che, da solo o inserito in un meccanismo più complesso serve per compiere una determinata funzione.

- **Funzione**: particolare costrutto sintattico, che permette di raggruppare una sequenza di istruzioni in un unico blocco di istruzioni espletando così una determinata e in generale più complessa operazione, azione o elaborazione sui dati del programma stesso in modo tale che a partire da determinati argomenti restituisca determinati output.
- **Google Play Store**: negozio virtuale online di applicazioni, brani musicali, pellicole cinematografiche, libri e riviste sviluppato da Google Inc. principalmente per offrire servizi ai dispositivi mobili Android.
- **HTTP(S)**: protocollo standard per il trasferimento di ipertesti.
- **ID**: abbreviazione, tipica della lingua inglese, per “identificatore”.
- **Interfaccia utente**: elementi del programma con cui l’utente può interagire.
- **Java**: particolare linguaggio di programmazione con *paradigma* oggettivo.
- **JSON**: *linguaggio di markup* derivato dal linguaggio JavaScript.
- **Libreria**: insieme di funzioni o strutture dati predefinite e predisposte per essere collegate ad un programma software attraverso opportuno collegamento.
- **Linguaggi di Markup**: Linguaggio utilizzato per la rappresentazione di dati strutturati.
- **Metadati**: informazioni che descrivono un ulteriore insieme di informazioni (veri e propri dati)
- **Output**: indica il risultato di una elaborazione ed in senso più ampio il risultato o l'insieme dei risultati prodotti.
- **Overview**: visione d’insieme, riepilogo.
- **Paradigma**: insieme di strumenti concettuali forniti da un linguaggio di programmazione.
- **Pattern**: tecniche di programmazione tipiche per un determinato *paradigma*, consolidate e largamente diffuse.
- **Programma**: è un insieme di istruzioni che produce soluzioni per una data classe di problemi automatizzati.
- **Protocollo**: insieme di regole che governano ogni attività di scambio di dati fra due entità si hanno protocolli per il trasferimento dei file, per l'accesso alla rete di ogni livello.
- **Repository**: “deposito” per codice sorgente e documentazione progettuale, che mantiene inoltre traccia delle modifiche apportate al contenuto, al fine di permetterne il ripristino in qualunque momento.
- **Scenario**: determinata sequenza di azioni in cui tutte le scelte sono state già effettuate.
- **SDK**: kit per lo sviluppo di software.
- **Software**: termine generico che definisce programmi e procedure utilizzati per far eseguire al dispositivo un determinato compito.
- **SQLite:** piccolo DBMS ottimizzato per piccoli database locali.
- **Store**: sito web che offre cataloghi commerciali e merci da acquistare online.
- **UML (Unified Modeling Language)**: Linguaggio universale di modellazione e specifica.
- **Upload**: procedura con cui vengono inviati dati o file dal dispositivo locale ad uno remoto.
- **URL**: indirizzo che identifica univocamente una risorsa (pagine html, file, immagini e quant'altro) su Internet.
- **XML**: Particolare *linguaggio di markup*, con enfasi su dati organizzati in gerarchie.

3. Overview dell'infrastruttura

Al fine di garantire la fruizione del servizio, è necessario un interfacciamento agli open-data analogo a quella rappresentato in figura:



Si farà uso del seguente dataset:

- https://data.europa.eu/euodp/en/data/dataset/1gXgb0Yj73R4ttDChQ5Wyg

4. Struttura del dataset

Il dataset in questione si presenta come un archivio compresso `.zip`, il quale verrà scompattato in seguito al download e mantenuto localmente fino al successivo aggiornamento. L'archivio conterrà diversi file codificati in JSON (la versione XML degli stessi verrà invece scartata), i quali saranno letti e trasportati all'interno di un database SQLite, grazie ad un componente realizzato appositamente.

4.1. Procedura di download

Il download avverrà secondo la seguente modalità:

1. Si apre una richiesta HTTPS verso l'URL di download del dataset;

2. Si confronta il valore dell'intestazione HTTP "`Last-Modified`" con la versione in cache. Se il valore è identico, non sarà necessario proseguire.

3. Altrimenti, si procede scaricando il corpo della richiesta in un percorso temporaneo.

4. Si scompatta l'archivio nella locazione temporanea, si leggono i vari file e si inizia a popolare con essi il database SQLite (che verrà svuotato preventivamente).

5. Si eliminano sia l'archivio sia i file estratti precedentemente, per liberare spazio sul dispositivo dell'utente. Se l'operazione è andata a buon fine, si memorizza all'interno del database il valore dell'intestazione "`Last-Modified`".

4.2. File di interesse

Segue l'elenco dei file di interesse:

- `additives.rdf.additive.json`: Nomi e informazioni sugli ingredienti
- `additives.rdf.application.json`: Investigazioni sugli ingredienti da parte dell'EU
- `Additives.rdf.document.json`: Eventuale documentazione di riferimento

Tutti gli altri file vanno scartati a completamento dell'estrazione dell'archivio.

4.3. Organizzazione dell'informazione da estrarre

L'informazione presente nel dataset è organizzata come illustrato di seguito.

Tenere presente che, al momento della stesura del presente documento, tutte le occorrenze di `$URL` dovranno essere sostituite in fase implementativa dalla stringa:

http://ec.europa.eu/semantic_webgate/dataset/additives/resource

La quale, stando al file `readme.txt` fornito con il dataset, può essere soggetta a variazioni.

4.3.1. Nomi ed informazioni sugli ingredienti

L'elenco completo degli ingredienti (additivi alimentari) e relative informazioni è contenuto all'interno del file `additives.rdf.additive.json` (vedere struttura in appendice).

4.3.2. Investigazioni da parte dell'UE

L'elenco completo delle approvazioni e disapprovazioni relative ai singoli ingredienti è contenuto all'interno del file `additives.rdf.application.json` (vedere struttura in appendice).

4.3.3. Documentazione fornita in supporto alle investigazioni UE

L'elenco completo delle approvazioni e disapprovazioni relative ai singoli ingredienti è contenuto all'interno del file `additives.rdf.application.json`(vedere struttura in appendice).

4.4. Rappresentazione dell'informazione interna all'app

Le informazioni estratte secondo le modalità illustrate al paragrafo 4.3 precedente saranno riorganizzate all'interno del database dell'applicazione, in un unica tabella "*Additives*", come illustrato al paragrafo 5.4.

5. Applicazione mobile

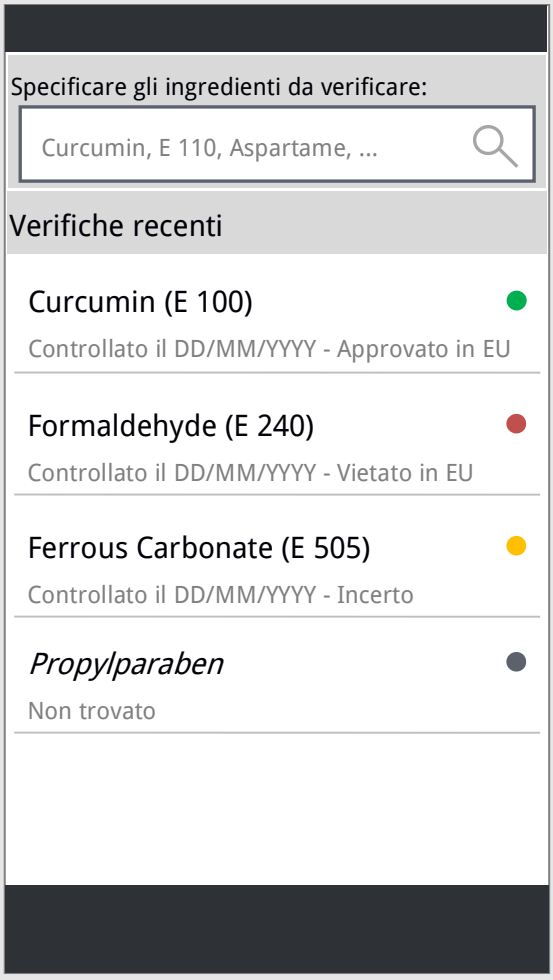
Sarà messa a disposizione degli utenti un'app, scaricabile dallo *store* del dispositivo mobile (smartphone, tablet, ecc.) da essi posseduto. Essa sarà, almeno inizialmente, disponibile in esclusiva su piattaforma Android, e dunque nella sezione applicazioni del Google Play Store.

5.1. Dipendenze esterne

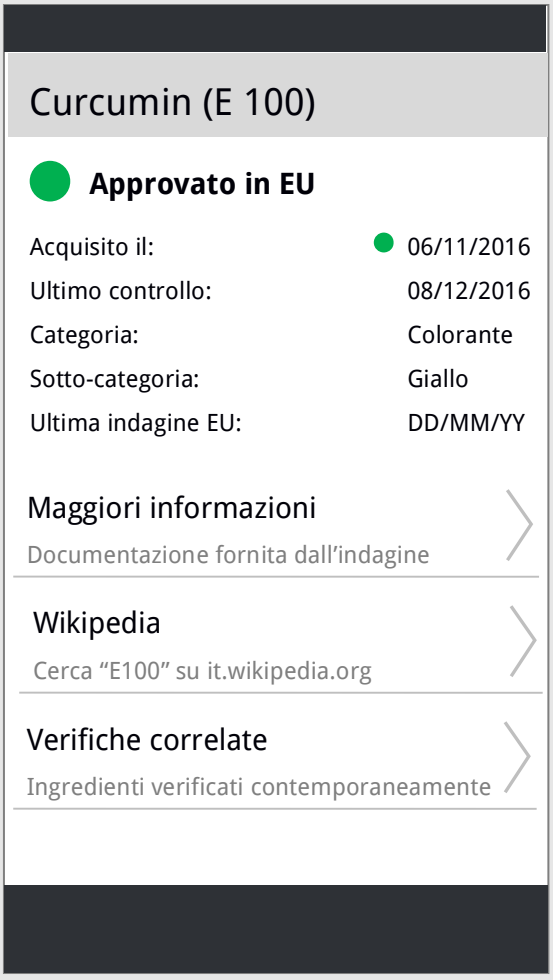
L'app sarà realizzata con l'**Android SDK** (https://developer.android.com/sdk/index.html) e l'IDE ufficiale **Android Studio**. Pertanto, al momento, non si prevede l'utilizzo di librerie o componenti di terze parti.

5.2. Concept per l'interfaccia utente

L'interfaccia utente dell'applicazione finale sarà basata sui seguenti modelli.



Schermata principale

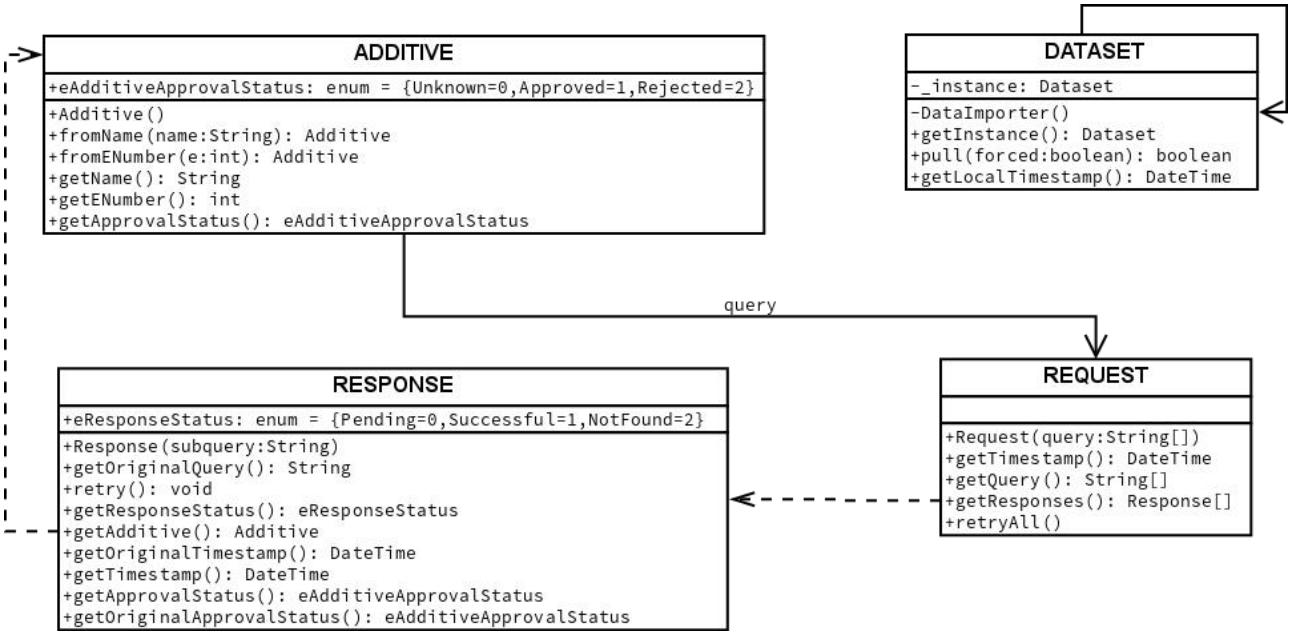


Dettagli

5.3. Organizzazione del codice

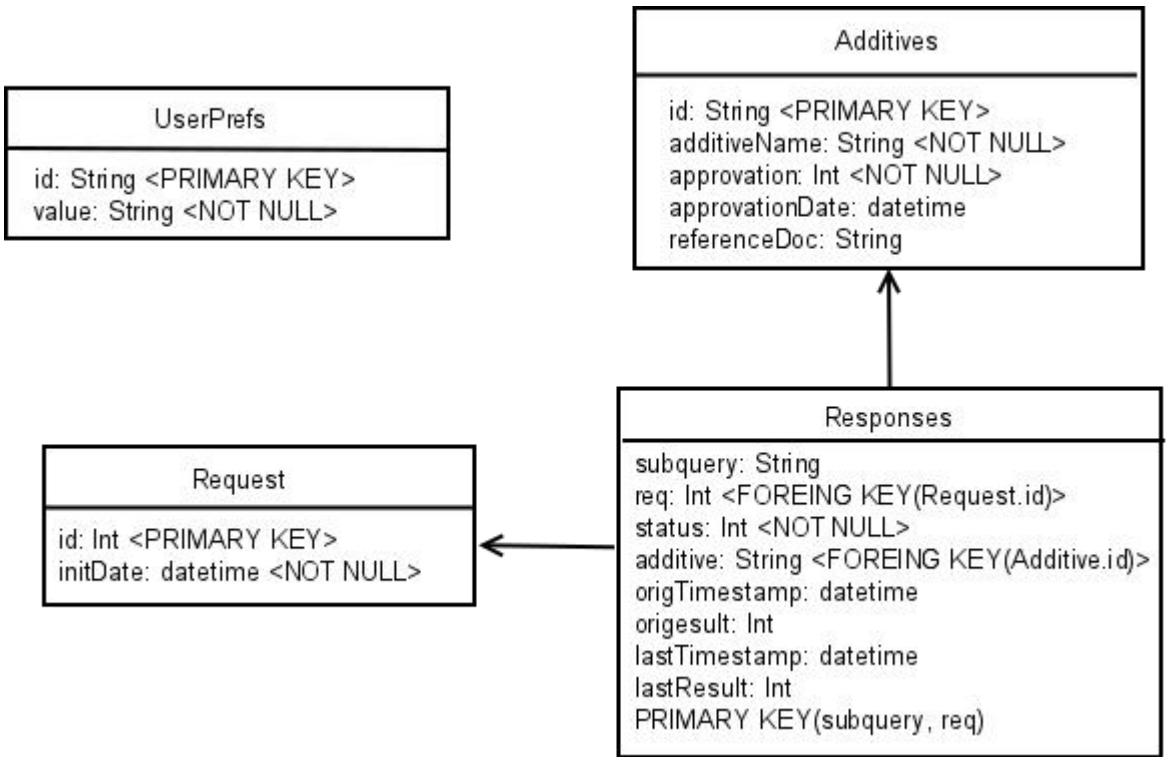
Il codice applicativo, così come previsto dal linee guida di sviluppo per Android, sarà realizzato in linguaggio Java seguendo il paradigma oggettivo e i relativi pattern.

Segue un modello delle componenti principali dell'infrastruttura:



5.4. Organizzazione del database

L'app farà uso di un piccolo database interno (*SQLite*, seguendo le linee guida di sviluppo Android), al fine di memorizzare una copia locale del dataset come illustrato al paragrafo 4.1, lo storico delle verifiche lanciate dall'utente, ed eventuali metadati.



6. Appendice (struttura files)

Struttura del file `additives.rdf.additive.json`:

```
{ items: [
  {
    @context: "$URL/Additive",
    @subject: "$URL/additive-262",
    $URL/additiveName: "Aluminium silicate (Kaolin)",
    $URL/eNumber: "E 559",
    $URL/id: "262",
    $URL/isGroup: "No"
  },
  ...
]; }
```

I campi di nostro interesse dunque sono:

- Il `@subject`, per riferirsi ai singoli ingredienti;
- L'`additiveName` e l'`eNumber`, per ricercare nel dataset gli ingredienti richiesti dall'utente.

Struttura del file `additives.rdf.application.json`:

```
{ items: [
  {
    @context: "$URL/Application",
    @subject: "$URL/application-124",
    $URL/applicationDate: "2010-06-16 00:00:00.0",
    $URL/applicationStatus: "application accepted",
    $URL/decision: "positive",
    $URL/decisionDate: "2011-11-25 00:00:00.0",
    $URL/hasAdditive: "$URL/additive-54",
    $URL/hasDocument: "$URL/document-2894",
    $URL/id: "124",
    $URL/refNumberAppProc: "2010/03",
  },
  ...
]; }
```

I campi di nostro interesse dunque sono:

- La `decision` e la `decisionDate`, per calcolare l'esito delle richieste dell'utente;
- `hasAdditive`, ove disponibile, per collegare investigazioni ed ingredienti;
- `hasDocument`, se ed ove disponibile, per collegarle con la documentazione.

Struttura del file `additives.rdf.application.json`:

```
{ items: [
  {
    @context: "$URL/Document",
    @subject: "$URL/document-2561"
    $URL/documentType: "information",
    $URL/documentUrl: "https://webgate.ec.europa.eu/sanco_…",
    $URL/documentReference: "DE - TEIL D LEBENSMITTELKATEGORIEN",
    $URL/fileName: "DE - TEIL D LEBENSMITTELKATEGORIEN …",
    $URL/id: "2561",
  },
  ...
]; }
```

I campi di nostro interesse dunque sono:

- Il `@subject`, per riferirsi ai singoli documenti;
- `documentUrl`, ove disponibile, per fornire un link agli utenti interessati;