

**Progetto Centri Vaccinali**

Laboratorio Interdisciplinare A

**Portale Cittadini**

**Centro Vaccinale**

**Manuale Tecnico**

Preparato da

**Daniel Satriano Mat. 745232**

**Claudio Menegotto Mat. 745394**

**Cristian De Nicola Mat. 744954**

**Francesco Cavallini Mat. 746933**

**Sommario**

[**INTRODUZIONE** 3](#_Toc96888163)

[Librerie esterne utilizzate 3](#_Toc96888164)

[Struttura generale delle classi 4](#_Toc96888165)

[Struttura del DataBase 5](#_Toc96888166)

[**CORE CLASSES** 6](#_Toc96888167)

[**Classi enumerative** 8](#_Toc96888168)

# **INTRODUZIONE**

1. **Centro Operatori** è un progetto sviluppato nell’ambito del progetto di Laboratorio A per il corso di laurea in Informatica dell’Università degli Studi dell’Insubria.

Il progetto è sviluppato in Java 12, usa un’interfaccia grafica costruita con OpenJFx 12 ed è stato sviluppato e testato sul sistema operativo Windows 10 e Windows 11.

## **Librerie esterne utilizzate**

1.1 L’applicazione fa uso di 3 librerie:

* **OpenJFx 12:** La libreria OpenJFx 12 contiene tutti gli elementi per lo sviluppo dell’interfaccia grafica. L’utilizzo di questa libreria si è reso necessario poiché JavaFx non è più incluso nel Java Development Kit di Oracle dalla release di Java 9.
* **Jfoenix 9.0.10:** La libreria Jfoenix è una open source di java che permette di implementare (graficamente) “Google Material Design”. Usato assieme a scene builder.
* **Gson 2.8.6:** La libreria Gson è una libreria sviluppata da Google che permette la serializzazione e la de serializzazione di oggetti a JSON o il contrario.

## **Struttura generale delle classi**

1.2 Il progetto è strutturato fondamentalmente in 2 rami: le core classes + classi enumerative e le classi adibite alla gestione dell’interfaccia grafica [Figura 1]

Graphical user interface

Description automatically generated with medium confidence

* **Classi “core”:**
  + CentroVaccinale
  + EventoAvverso
  + Indirizzo
  + Storico
  + UtenteVaccinato
  + ApiRequest
* **Gestori file**
  + JsonReadWrite
* **Classi enumerative:**
  + Evento
  + FilePaths
  + Qualificatore
  + Severita
  + Tipologia
  + Vaccini
* **Controllers:**
  + MainWindow
* **Rispettivi FXML:**
  + MainWindow

Figura 1

## **Struttura del DataBase**

* 1. Il funzionamento dell’applicazione dipende dalla presenza di un pool di dati provenienti da diversi file testuali (salvati in formato “.dati” contenuti nella cartella database). Possiamo distinguere i vari file per diversi scopi:
* **CentroVaccinali**: file contenente un pool di centri vaccinali (registrati tramite l’applicazione apposita), l’applicazione non esegue alcuna scrittura sul file ma ne è fondamentale la lettura per la visualizzazione e ricerca di tutti i centri vaccinali. Entrambe le funzionalità sono presenti nella home.
* **Cittadini\_Registrati**: File contenente le credenziali di ogni utente, le quali informazioni anagrafiche vengono invece salvati all’interno del file “Vaccinati\_NomeCentroVaccinale”. L’applicazione può accedere sia in lettura che in scrittura al file per i motivi seguentemente elencati:
  + **Lettura**: la lettura è necessaria per effettuare il login dell’utente al centro vaccinale al quale si è iscritto.
  + **Scrittura**: la scrittura è necessaria per permettere a nuovi utenti di iscriversi presso nuovi centri vaccinali.
* **Vaccinati\_NomeCentroVaccinale**: File contenente un elenco di Utenti (e le loro informazioni anagrafiche), gli eventi avversi (caso in cui ne abbiano registrato uno). L’applicazione può accedere sia in lettura che in scrittura al file per i motivi seguentemente elencati:
  + **Lettura**: nel momento in cui l’utente decide di aprire la tab relativa al centro vaccinale l’applicazione legge ogni utente e ne mostra l’evento avverso sottoscritto. È inoltre necessario per la verifica dei dati anagrafici dell’utente già inserito nel sistema dall’applicazione complementare ma non ancora
  + **Scrittura**: è possibile per l’applicazione aggiungere un evento avverso una volta che l’utente ha effettuato il login. Quando questo avviene il file viene aggiornato con l’inserimento dell’evento avverso.

# **CORE CLASSES**

* 1. Verranno presentate ora le core classes nel dettaglio, non verrà affrontata invece una discussione dettagliata delle classi adibite alla gestione dell’interfaccia grafica poiché di interesse marginale e non essendo l’interfaccia grafica espressamente richiesta nei requisiti del progetto.

(N.B -> la maggior parte della classi presenta un override del toString e un metodo costruttore che non andremo a citare)

**CentroVaccinale:**

* 1. La classe **Centro Vaccinale** identifica i centri vaccinali e le loro funzioni.

Più in particolare presenta al suo interno quattro tag per l’identificazione:

1. LinkedList<Short> **IDVaccinazioni** : è una lista di ID delle vaccinazioni effettuate nel centro, in modo da poterne tenere traccia.
2. String **Nome** : indica il nome del centro vaccinale.
3. [Indirizzo](#Indirizzo) **Indirizzo** : va ad indicare l’indirizzo stradale del centro.
4. [Tipologia](#Tipologia) **Tipologia** : indica la tipologia del centro.

La classe inoltre presenta inoltre un metodo:

* + **getNome**() : che restituisce il nome del centro vaccinale.

**EventoAvverso:**

* 1. La classe **Evento Avverso**  identifica e classifica gli eventi avversi che sono capitati agli utenti a cui è stato sottoministrato il vaccino.

Presenta al suo interno quattro tag, i quali verranno adesso elencati:

1. [Evento](#Evento) **evento** : va ad identificare il tipo di evento che è capitato all’utente.
2. [Severita](#Severità) **severità** : indica, come descritto dal nome, la severità dell’evento subito dall’utente.
3. Short **IDVaccinazione** : serve per tenere traccia della vaccinazione somministrata all’utente che sta riportando l’evento avverso.
4. String **noteOpzionali** : come da nome, sono note opzionali che l’utente può aggiungere prima di pubblicare l’evento avverso.

**Indirizzo:**

* 1. La classe **indirizzo** serve come classe complementare alla classe [CentroVaccinale](#CentroVaccinale), in quanto contiene al suo interno tutte le informazioni riguardanti l’indirizzo stradale del centro. Al suo interno possiamo trovare sei tag che la caratterizzano:

1. [Qualificatore](#Qualificatore) **qualificatore** : va ad indicare il qualificatore dell’indirizzo stradale.
2. String **nome** : nome della strada a cui fa riferimento l’indirizzo.
3. Int **numeroCivico** : numero civico della strada ove situato il centro.
4. String **comune** : comune dove è situato il centro.
5. String **provincia** : provincia dove è situato il centro.
6. Int **cap** : codice di avviamento postale del comune.

**Storico:**

1. La classe **Storico** serve a implementare la funzione dello storico dei vaccini nell’applicazione, presenta al suo interno due tag:
2. StringProperty **informazioniSomministrazioni** :contiene al suo interno informazioni riguardanti il vaccino usato e il nome e cognome della persona a cui è stato somminstrato.
3. StringProperty **dataSomministrazione** : indica la data di somminstrazione del vaccino.

**UtenteVaccinato:**

1. La classe **Utente Vaccinato** serve a salvare le informazioni degli utenti vaccinati di tutti i centri registrati nell’applicativo. Al suo interno presenta 8 tag:
2. String **nomeCentroVaccinale** : nome del centro.
3. String **nome** : nome dell’utente vaccinato.
4. String **cognome** : cognome dell’utente vaccinato.
5. String **codiceFiscale**: codice fiscale dell’utente vaccinato.
6. String **dataSomministrazione** : data somministrazione del vaccino.
7. [Vaccini](#Vaccini) **vaccino** : tipo di vaccino effettuato.
8. Short **idVaccinazione** : id della vaccinazione.
9. [EventoAvverso](#EventoAvverso) **evento** : \*\*

La classe presenta tre metodi:

* + **getIdVaccinazione**() : restituisce l’idVaccinazione dell’utente.
  + **getInformation**() : restituisce tutte le informazioni dell’oggetto sotto forma di stringa in UPPER case.
  + **getDataSomministrazione**() : restituisce la data di somminstrazione-

**ApiRequest:**

1. La classe **Api request** è una classe utilitaria alla UI, in quanto ha il compito di andare a prelevare le informazioni che vengono mostrate nella home.

Per far questo adopera la libreria java.net, in particolare le sottoclassi della classe http.

E anche la libreria gson di google per la manipolazione del json.

Presenta al suo interno tre metodi:

* + **makeRequest**(String url) : prende in input una pagina web e ne restituisce le informazioni sotto forma di JsonArray andando a richiamere il metodo [convertJSONintoJSONobject](#convertJSONintoJSONobject). (N.B assicurarsi che la pagina web data in input printi a schermo un array di json per il corretto funzionamento del metodo.)
  + **convertJSONintoJSONobject**(String json) : metodo che presa in input una stringa contenente della sintassi json la trasforma in un oggetto JsonObject.
  + **infoGrabber**(JsonObject tmp, String nodo) : restituisce al codice chiamante una determinata sezione del json object definita dal tag *nodo*.

# **Classi enumerative**

Vengono presentate ora le classi enumerative usate nel progetto.

* **Evento:**

Classe enumerativa utilizzata nella gestione degli eventi avversi, in particolare possiamo trovare al suo interno i seguenti tag:

1. Mal di testa
2. febbre,
3. dolori\_muscolari\_e\_articolari
4. infodonopatia
5. tachicardia
6. crisi\_ipertensiva
7. altro

* **FilePaths:**

Classe enumerativa di supporto alla programmazione, in quanto possiede al suo interno i percorsi per i “database”. La classe presenta al suo interno i seguenti tag:

* 1. CentriVaccinali
  2. CittadiniRegistrati
  3. VaccinatiNomeCentro
* **Qualificatore:**

Classe enumarativa utilizzata per indicare i diversi tipi di qualificatori di un indirizzo. Possiamo trovare all’interno della classe i seguenti tag:

1. Via
2. Viale
3. Piazza
4. Corso.

* **Severita:**

Enum utilizzata per la gestione della severità. Al suo interno troviamo i seguenti tag:

1. Molto\_bassa\_1
2. bassa\_2
3. fastidiosa\_3
4. sopportabile\_4
5. insopportabile\_

* **Tipologia:**

Enum utilizzato per la definizione della tipologia di un centro vaccinale. Al suo interno troviamo i seguenti tag:

1. Aziendale
2. Ospedaliero
3. Hub.

* **Vaccini:**

Enum utilizzato per la definizione dei diversi vaccini disponibili in commercio. Al suo interno abbiamo i seguenti tag:

1. Pfizer
2. AstraZeneca
3. Moderna
4. JeJ