Eventi avversi vaccinazioni

Manuale tecnico

Università degli Studi dell’Insubria  
Sviluppato da:   
Antonicelli Sandy, matricola: 744947, VA  
Caffi Nicolò, matricola: 745391, VA  
Margherini Giorgio, matricola: 744148, VA

# Indice

[**Indice**](#_ctdnaom98uou) **2**

[**Introduzione**](#_ip451wimlrtr) **5**

[**Struttura del progetto**](#_4ez6p2skkdy1) **5**

[**Database**](#_js33aop191a3) **5**

[Schema concettuale](#_6wy3d7daqerv) 6

[vincoli di integrità](#_pm3vuy8hh36i) 6

[scelte progettuali](#_axe4d4lmezx6) 6

[Schema concettuale ristrutturato](#_x5w342tny3x4) 7

[Traduzione](#_gmjey4bc1wh1) 7

[Implementazione](#_vvqg07jxz4ij) 8

[Creazione database](#_w638hf5s0zpz) 8

[Creazione Tabelle](#_3y50efhdp0eb) 8

[Inserimenti](#_3nlbk2ix2jhb) 10

[nuovo centro vaccinale](#_tgx9kkt1ck7c) 10

[nuovo vaccinato](#_21911zbs2u51) 10

[nuovo utente](#_byzl5p26cnm3) 10

[nuovo evento avverso](#_ii3v4sudm58w) 10

[nuova aggregazione di eventi avversi](#_u9f15tr26b1p) 10

[nuova aggregazione di eventi](#_q1pvkgkhy80g) 10

[Query](#_yq2wyqr9gfok) 10

[Ricerca per nome di un centro vaccinale](#_7vri91jet9xb) 11

[Ricerca per comune e tipologia di un centro vaccinale](#_mnyrd1wdk2he) 11

[Selezione centro vaccinale](#_hp3beq6vh1ap) 11

[Selezione vaccinato](#_2dt7dhg4m715) 11

[Selezione cittadino registrato](#_dia7rd94f6k5) 11

[Selezione eventi avversi rispetto a un vaccinato](#_jqlaz3te5f2f) 11

[Selezione evento avverso](#_ttovb9szvatf) 12

[Selezione aggregazione di eventi relativi ad un centro vaccinale](#_5j2uo1gczr82) 12

[Aggiornamenti](#_vav6qjekdph) 12

[Update aggregazione eventi](#_ao1nszb7j3jo) 12

[Update centri vaccinali](#_dwt7xtbvn5g) 12

[**Applicazioni**](#_vlxclvxc4zqa) **13**

[Data model](#_o6cmn6okzj2q) 13

[Comunicazione](#_9gi84zxpzq6z) 14

[**Centri Vaccinali**](#_7vjen13o5lxs) **14**

[Packages](#_nnykqbdn0qfb) 14

[centrivaccinali](#_t2vgciz3yiqz) 15

[database](#_5r3ch81056cs) 15

[datamodel](#_xojai46sh2a6) 15

[interfaces](#_qv8u6qkh5j6r) 15

[**Cittadini**](#_j252phrkhs58) **15**

[Packages](#_2synr4bwaqhp) 16

[cittadini](#_208p2mnn9klr) 16

[gui](#_b8gpppfoagg9) 16

[interfaces](#_e62r69nuc0u6) 19

[datamodel](#_rvxj3ew7rwy8) 19

[**Bibliografia**](#_ssoknhi6wer) **19**

# 

# 

# Introduzione

Il progetto “Eventi Avversi Vaccinazioni” è stato sviluppato nell’ambito del laboratorio interdisciplinare B del corso di laurea in informatica dell’Università degli Studi dell’Insubria.

# Struttura del progetto

Il progetto si compone di due applicazioni separate: “Centri Vaccinali” e “Cittadini”. La prima funge da server e si occupa della comunicazione con il database e la gestione dei dati relativi ai centri vaccinali; la seconda permette agli utenti comuni di accedere ai servizi forniti.

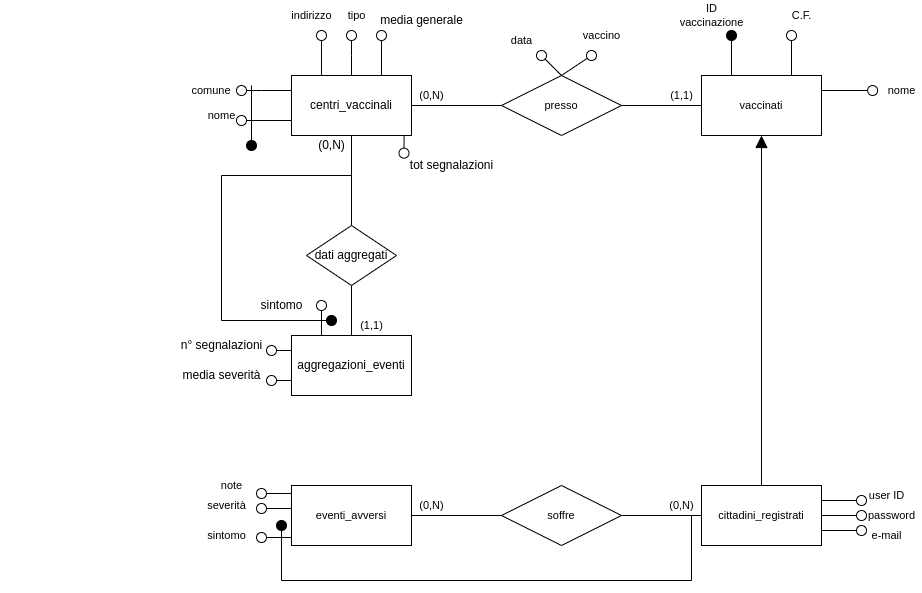
L’applicazione “Centri Vaccinali” è formata da due parti, una sezione (server) che fornisce un'interfaccia a “Cittadini” per comunicare con il database e una sezione che permette ai gestori dei centri vaccinali di inserire nuovi centri e di registrare i cittadini vaccinati.

L’applicazione “Cittadini” fornisce l’interfaccia grafica che permette ai cittadini di cercare e visualizzare informazioni sui centri vaccinali ed, eventualmente, segnalare eventi avversi presso il centro in cui hanno ricevuto la vaccinazione.

# Database

Il database per il progetto è stato implementato con postgreSQL 14 e comunica con l’applicazione grazie a JDBC.

## Schema concettuale



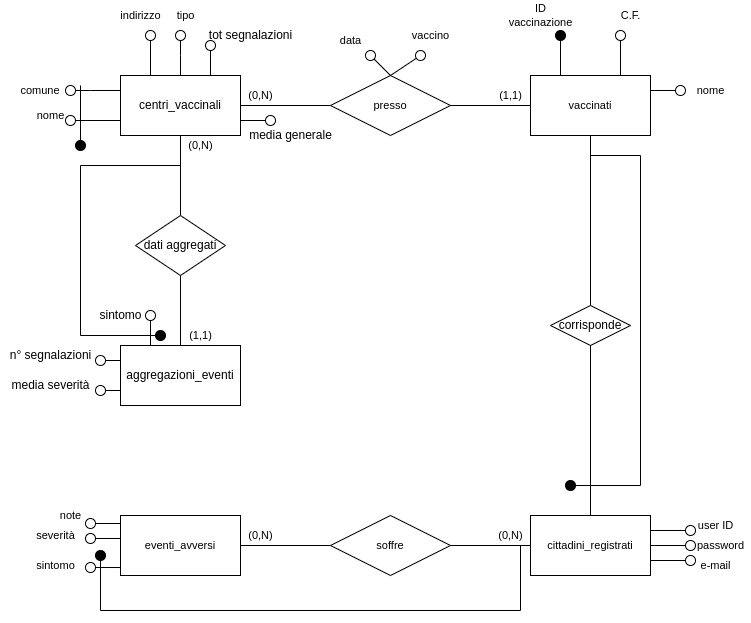
### vincoli di integrità

* ID vaccinazione: su 16 bit (da 0 a 65535);
* Codice Fiscale: segue il formato del codice fiscale;
* user ID: univoco;
* e-mail: segue il formato di un indirizzo e-mail;

### scelte progettuali

* Data la natura privata dei dati relativi agli eventi avversi, l’entità aggregazioni\_eventi serve a consultare i dati senza poter risalire alle persone a cui si riferiscono.

## Schema concettuale ristrutturato



Lo schema viene ristrutturato eliminando la gerarchia di generalizzazione tra vaccinati e cittadini\_registrati, trasformandola in un’associazione e ponendo ID vaccinazione come chiave esterna di cittadini\_registrati.

## Traduzione

vaccinati (ID vaccinazione, codice fiscale, nome, nomecentri\_vaccinali, comunecentri\_vaccinali, data vaccinazione, tipo vaccino)

cittadini\_registrati (ID vaccinazionevaccinati, user ID, password, e-mail)

eventi\_avversi (sintomo, ID vaccinazionecittadini\_registrati, severità, note, nomecentri\_vaccinali, comunecentri\_vaccinali)

centri\_vaccinali (nome, comune, indirizzo, tipo, totale segnalazioni, media generale)

aggregazioni\_eventi (sintomo, nomecentri\_vaccinali, comunecentri\_vaccinali, numero segnalazioni, media severità)

## Implementazione

### Creazione database

CREATE DATABASE centrivaccinali

### Creazione Tabelle

CREATE TABLE CentriVaccinali(  
 nome VARCHAR(50) NOT NULL,  
 comune VARCHAR(35) NOT NULL,  
 indirizzo VARCHAR(50) NOT NULL,  
 tipo VARCHAR(20) NOT NULL,  
 totale\_segnalazioni NUMERIC,  
 media\_generale NUMERIC,  
 PRIMARY KEY(nome,comune)  
);

CREATE TABLE Vaccinati(  
 ID\_vaccinazione INTEGER PRIMARY KEY CHECK (ID\_vaccinazione>=0 AND ID\_vaccinazione<=65535),  
codice\_fiscale CHAR(16) UNIQUE NOT NULL CHECK(codice\_fiscale ~ '[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z]'),  
 nome\_cognome VARCHAR(40),  
 nome VARCHAR(50) NOT NULL,  
 comune VARCHAR(35) NOT NULL,  
 data\_vaccinazione DATE DEFAULT CURRENT\_DATE,  
 tipo\_vaccino VARCHAR(20),  
 FOREIGN KEY(nome,comune) REFERENCES CentriVaccinali  
);

CREATE TABLE Cittadini\_Registrati(  
 ID\_vaccinazione INTEGER PRIMARY KEY REFERENCES Vaccinati,  
 user\_ID VARCHAR(30) NOT NULL UNIQUE,  
 password VARCHAR(30) NOT NULL,  
 email VARCHAR(50) NOT NULL CHECK(email ~\* '^[-\w.]+@[A-Z\_.]+?[A-Z]{2,4}$')  
);

CREATE TABLE eventi\_avversi(  
 sintomo VARCHAR(30) NOT NULL,  
 id\_vaccinazione INTEGER NOT NULL REFERENCES Cittadini\_Registrati,  
 severita INTEGER CHECK(severita>=1 AND severita <=5),  
 note VARCHAR(256),  
 nome VARCHAR(50) NOT NULL,  
 comune VARCHAR(35) NOT NULL,  
 PRIMARY KEY(sintomo,id\_vaccinazione),  
 FOREIGN KEY(nome,comune) REFERENCES CentriVaccinali  
);

CREATE TABLE aggregazioni\_eventi(  
 sintomo VARCHAR(30) NOT NULL,  
 nome VARCHAR(50) NOT NULL,  
 comune VARCHAR(35) NOT NULL,  
 numero\_segnalazioni INTEGER,  
 media\_severita NUMERIC,  
 PRIMARY KEY(sintomo,nome,comune),  
 FOREIGN KEY(nome,comune) REFERENCES CentriVaccinali  
);

## Inserimenti

### nuovo centro vaccinale

INSERT INTO centri\_vaccinali (nome, comune, indirizzo, tipo)  
VALUES (?,?,?,?);

### nuovo vaccinato

INSERT INTO vaccinati (ID\_vaccinazione, codice\_fiscale, nome\_cognome, nome, comune, data\_vaccinazione, tipo\_vaccino)  
VALUES (?,?,?,?,?,?);

### nuovo utente

INSERT INTO cittadini\_registrati (ID\_vaccinazione, user\_ID, password, email)  
VALUES (?,?,?,?);

### nuovo evento avverso

INSERT INTO eventi\_avversi  
VALUES (?,?,?,?,?,?);

### nuova aggregazione di eventi avversi

INSERT INTO aggregazioni\_eventi(sintomo, nome, comune)  
SELECT DISTINCT sintomo, nome, comune  
FROM eventi\_avversi

### nuova aggregazione di eventi

Eseguito quando viene segnalato un evento avverso per un certo centro vaccinale con un sintomo che non era ancora stato inserito

INSERT INTO aggregazioni\_eventi(sintomo,nome,comune)  
SELECT DISTINCT sintomo,nome,comune FROM eventi\_avversi;

## Query

### Ricerca per nome di un centro vaccinale

SELECT nome, comune  
FROM centri\_vaccinali  
WHERE nome LIKE('%?%')

### Ricerca per comune e tipologia di un centro vaccinale

SELECT nome, comune  
FROM centri\_vaccinali  
WHERE comune=? AND tipo=?  
ORDER BY comune

### Selezione centro vaccinale

SELECT \*  
FROM centri\_vaccinali  
WHERE nome=? AND comune=?  
ORDER BY comune

### Selezione vaccinato

SELECT \*  
FROM vaccinati  
WHERE id\_vaccinazione=?

### Selezione cittadino registrato

SELECT \*  
FROM cittadini\_registrati  
WHERE user\_id=?

### Selezione eventi avversi rispetto a un vaccinato

SELECT \*  
FROM eventi\_avversi  
WHERE id\_vaccinazione=?

### Selezione evento avverso

SELECT \*  
FROM eventi\_avversi  
WHERE user\_id=? AND sintomo=?

### Selezione aggregazione di eventi relativi ad un centro vaccinale

SELECT \*  
FROM aggregazioni\_eventi  
WHERE nome=? AND comune=?  
ORDER BY numero\_segnalazioni DESC, media\_severita DESC

## Aggiornamenti

### Update aggregazione eventi

Calcola il numero di segnalazioni e la media delle severità per ogni sintomo rispetto al relativo centro vaccinale.

UPDATE aggregazioni\_eventi ae  
SET numero\_segnalazioni=(SELECT COUNT(\*)  
 FROM centri\_vaccinali cv, eventi\_avversi ea  
 WHERE ea.nome=cv.nome AND ea.comune=cv.comune AND ae.sintomo=ea.sintomo),  
media\_severita=(SELECT AVG(severita)  
 FROM eventi\_avversi ea, centri\_vaccinali cv  
 WHERE ea.nome=cv.nome AND ea.comune=cv.comune AND ae.sintomo=ea.sintomo)

### Update centri vaccinali

Calcola il numero totale di segnalazioni e la media delle severità per ogni centro vaccinale.

UPDATE centri\_vaccinali cv  
SET totale\_segnalazioni=(SELECT DISTINCT SUM(numero\_segnalazioni)  
 FROM aggregazioni\_eventi  
 WHERE nome=cv.nome AND comune=cv.comune),  
media\_generale=(SELECT DISTINCT SUM(media\_severita \* numero\_segnalazioni)/SUM(numero\_segnalazioni)  
 FROM aggregazioni\_eventi  
 WHERE nome=cv.nome AND comune=cv.comune)

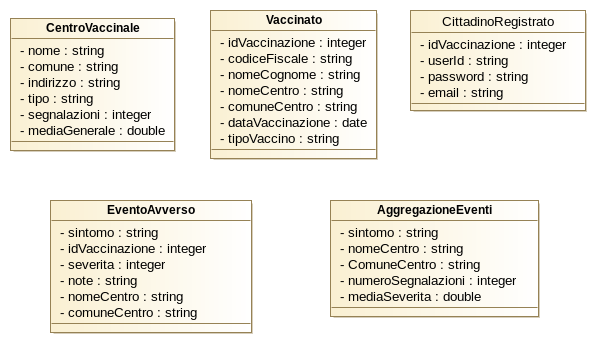
# Applicazioni

## Data model

Per semplificare il passaggio remoto e la gestione delle informazioni è stato creato il package “datamodel” contenente classi per la gestione dei dati in modo simile alla struttura che assumono nel database.

In particolare il package contiene le classi:

* AggregazioneEventi;
* CentroVaccinale;
* CittadinoRegistrato;
* EventoAvverso;
* Vaccinato;

le classi assumono la struttura presentata nel seguente class diagram.

## Comunicazione

La connessione tra le due applicazioni è implementata tramite java RMI.  
Il server è stato implementato come uno UnicastRemoteObject che viene esportato su un registry per permettere ai client di collegarsi ad esso.

# Centri Vaccinali

## Packages

L’applicazione Centri Vaccinali è costituita da i seguenti package:

* centrivaccinali;
* database;
* datamodel;
* interfaces.

### centrivaccinali

Il package centrivaccinali contiene la classe CentriVaccinali che funge da main per l’applicazione e implementa la classe UnicastRemoteObject per permettere la chiamata remota dei metodi. All’avvio vengono richieste le credenziali per collegarsi al database (user,password e host).

### database

Questo package contiene le classi Database e DatabaseProxy.  
La classe Database serve a collegarsi al database e, grazie al design pattern singleton, si assicura che solo una connessione al database possa essere aperta alla volta, così da poterla controllare in modo centralizzato.  
La Classe DatabaseProxy si occupa di richiedere al database tutte le operazioni necessarie al funzionamento dell’applicazione, in particolare le funzioni di inserimento, selezione, ricerca e aggiornamento dei dati.

### datamodel

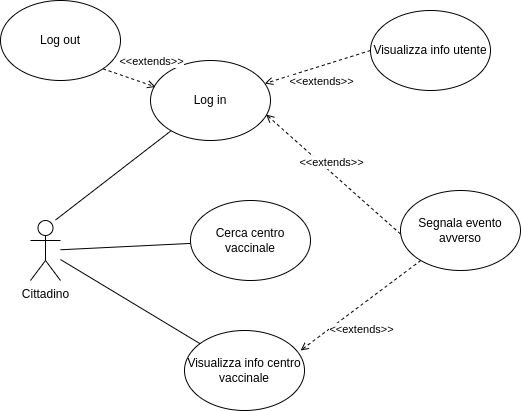
vedi Data model

### interfaces

Contiene l’interfaccia CentriVaccinaliInt, necessaria per la comunicazione con il client.

# Cittadini

L’applicazione Cittadini è stata sviluppata a partire dal seguente use-case diagram.



## Packages

L’applicazione è formata dai seguenti package:

* cittadini;
* gui;
* interfaces;
* datamodel.

### cittadini

Il package cittadini contiene la classe Cittadini che contiene il metodo main e che comunica con il server.

### gui

Il package gui contiene tutte le classi che compongono l’interfaccia grafica, sviluppata con java Swing e AWT:

* CentrePanel;
* EventPanel.
* Gui;
* HomePanel;
* LoginPanel;
* RegisterPanel;
* SearchPanel;
* UserPanel;

La classe Gui estende JFrame e gestisce la comunicazione con la classe Cittadini. Ciascuna delle altre classi estende JPanel e si occupa della creazione di un pannello specifico dell’interfaccia fungendo anche da ActionListener.  
In tre di queste classi (CentrePanel, SearchPanel e UserPanel) sono mostrate informazioni all’interno di JTable; a questo scopo i dati interessati vengono inseriti in array bidimensionali di String attraverso un metodo con complessità in tempo . Riportiamo di seguito parte del codice della classe UserPanel che dimostra tale processo.

for (int i = 0; i < size; i++) {  
 String[] temp = segnalazioni.get(i).toArray();  
 data[i][0] = temp[0];  
 data[i][1] = temp[2];  
 data[i][2] = temp[3];  
}

Dove “segnalazioni” è l’ArrayList<EventoAvverso> contenente le informazioni ricevute dal database, “size” è la dimensione di quest’ultima e “data” è l’array bidimensionale di String in cui vengono memorizzati i dati da mostrare nella tabella.

Segue una descrizione dettagliata delle classi del package.

* CentrePanel:  
   Genera un JPanel che mostra le informazioni di un dato centro vaccinale, inclusi i dati aggregati relativi agli eventi avversi segnalati per il centro sotto forma di JTable.  
   Se l’utente è registrato presso questo centro vaccinale, un bottone per inserire un nuovo evento avverso diventa visibile.
* EventPanel:  
   Genera un JPanel che permette di inserire un evento avverso, specificando sintomo (selezionabile da un JComboBox), severità (da 1 a 5) ed eventuali note testuali (massimo 256 caratteri). Un cittadino può inserire un solo evento per sintomo; in caso di errori nell’inserimento, l’utente viene notificato da un messaggio popup.
* Gui:  
   Crea il JFrame all’interno del quale viene visualizzata l’interfaccia, crea e gestisce la barra del menu e gestisce il passaggio da un pannello all’altro. Inoltre funge da intermediario tra la classe Cittadini e le altre classi del package gui.
* HomePanel:  
   Genera il JPanel iniziale dell’applicazione.
* LoginPanel:  
   Genera un JPanel che permette agli utenti già registrati di accedere all’applicazione. Vengono richesti username e password per l’accesso. In caso di accesso fallito, l’utente viene notificato con un messaggio popup.
* RegisterPanel:  
   Genera un JPanel che permette ad un cittadino che abbia ricevuto la vaccinazione di registrarsi all’applicazione. Vengono richiesti: nome e cognome, codice fiscale, e-mail, username, password (da inserire due volte) e il codice identificativo della vaccinazione (fornito al momento della stessa). In caso l’utente sia già registrato, abbia inserito due password diverse o ci sia stato un errore nella registrazione, l’utente viene notificato con un messaggio popup appropriato.
* SearchPanel:  
   Genera un JPanel che permette di cercare un centro vaccinale. La ricerca può essere eseguita specificando il nome del centro vaccinale desiderato (o parte di esso) o, alternativamente, inserendo il comune e la tipologia del centro; la scelta della tipologia avviene tramite un JcomboBox. I risultati della ricerca vengono mostrati in una Jtable che specifica il nome e il comune dei centri vaccinali trovati. Selezionando uno dei centri vaccinali è possibile vederne le relative informazioni.
* UserPanel:  
   Genera un Jpanel che mostra le informazioni dell’utente che ha eseguito l’accesso, compresa una lista degli eventi avversi segnalati all’interno di una Jtable.

### interfaces

Contiene l’interfaccia CentriVaccinaliInt, necessaria per la comunicazione con il server.

### datamodel

vedi Data model.

# Bibliografia

* B. Catania, E. Ferrari, e G. Guerrini. Sistemi di Gestione Dati: Concetti e Architetture, Città Studi Edizioni, 2006;
* Documentazione java, online: [docs.oracle.com](http://docs.oracle.com/);
* Java Swing tutorial, online: [www.javatpoint.com/java-swing](http://www.javatpoint.com/java-swing);
* Robert Sedgewick. Algoritmi in Java terza edizione, Pearson, 2003.