

*Polo territoriale di Como*

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL’INFORMAZIONE

INGEGNERIA INFORMATICA

**Corso di Ingegneria del Software**

****

**AuthOK**

**PROGETTO DEL CORSO DI INGEGNERIA DEL SOFTWARE**

**Parte II – Design**

Ferrario Stefano

Gumus Tayfun

Isella Paolo

Martinese Federico

Indice

[Introduzione 3](#_Toc501524912)

[Authorizer 4](#_Toc501524913)

[Gestore token 4](#_Toc501524914)

[Token 4](#_Toc501524915)

[Jsonrpc 5](#_Toc501524916)

[Class 5](#_Toc501524917)

[Zeromq 6](#_Toc501524918)

[Class 6](#_Toc501524919)

[User 7](#_Toc501524920)

[Class 7](#_Toc501524921)

**Introduzione**

L’implementazione del codice Java di AuthOK è la parte finale del processo di realizzazione del software. Essa ha il compito di dar vita al programma reale che dovrà rispettare tutti i modelli precedentemente realizzati, a partire dal Goal Diagram fino all’ultimo dei diagrammi UML. Uno alla volta, tutti i metodi sono stati implementati e integrati tra di loro al fine di realizzare la logica applicativa del programma. L’implementazione prevede la creazione delle diverse classi suddivise nei package “Authorizer”, “User”, “Jsonrpc” e “Zeromq”.

**Authorizer**

## **Gestore autorizzazioni**

La classe Gestore Autorizzazioni implementa tutti i metodi necessari per le operazioni che vengono eseguite sulle autorizzazioni. Il costruttore di questa classe utilizza il design pattern singleton per garantirne l’univocità. Durante l’istanziazione di un nuovo gestore viene creata una nuova HashMap autorizzazioni che conterrà le nuove autorizzazioni create per consentirne il salvataggio e le successive verifiche su di essi. I principali metodi implementati in questa classe sono:

* genera\_chiave\_unica
* creaAutorizzazione
* revocaAutorizzazione
* verificaEisitenzaAutorizzazione
* verificaValiditàAutorizzazione

## **Autorizzazione**

La classe Autorizzazione ha al suo interno il costruttore necessario per la creazione di un nuovo token e i metodi per recuperarne le sue informazioni. Ogni autorizzazione è costituita da un nome utente, un livello di accesso e una scadenza.

## **Gestore token**

La classe Gestore Token implementa tutti i metodi necessari per le operazioni che vengono eseguite sui token. Il costruttore di questa classe utilizza il design pattern singleton per garantirne l’univocità. Durante l’istanziazione di un nuovo gestore viene creata una nuova HashMap che conterrà i token creati per consentirne il salvataggio e le successive verifiche su di essi. I metodi implementati in questa classe sono:

* creaToken
* verificaToken
* cancellaTokenScaduti
* cancellaTokenChiave

## **Token**

La classe Token ha al suo interno il costruttore necessario per la creazione di un nuovo token e i metodi per recuperarne le sue informazioni.

**Jsonrpc**

## **Class**

testo

**Zeromq**

## **Class**

testo

**User**

## **Class**

testo

**Identificativo gruppo**: 10

**Progetto**: Tema A

**Componenti**:

845386 Ferrario Stefano [stefano6.ferrario@mail.polimi.it](mailto:stefano6.ferrario@mail.polimi.it)

843994 Gumus Tayfun [tayfun.gumus@mail.polimi.it](mailto:tayfun.gumus@mail.polimi.it)

854412 Isella Paolo [paolo.isella@mail.polimi.it](mailto:paolo.isella@mail.polimi.it)

845326 Martinese Federico [federico.martinese@mail.polimi.it](mailto:federico.martinese@mail.polimi.it)