Sirius

Sequenziatore

Definizione di prodotto

Versione 1.0.0

Ingegneria Del Software AA 2013-2014

Informazioni documento

Titolo documento:	Definizione Di Prodotto
Data creazione:	2014-01-29
Versione attuale:	1.0.0
Utilizzo:	Interno
Nome file:	*DefinizioneDiProdotto_v1.0.0.pdf*
Redazione:	Quaglio Davide
Approvazione:	Santangelo Davide
Distribuito da:	Sirius
Destinato a:	Prof. Vardanega Tullio
	Prof. Cardin Riccardo
	Zucchetti S.p.A.

Sommario

Tale documento andrà a trattare in modo approfondito le componenti e la struttura del prodotto il *Sequenziatore*trattate nel documento *SpecificaTecnica_v1.0.0.pdf*

Diario delle modifiche

Versione	Data	Autore	Ruolo	Descrizione

Indice

1 Introduzione

1.1 Scopo del documento

In questo documento si prefigge come obiettivo la definizione in modo approfondito della struttura, delle componenti e delle relazioni tra queste ultime del prodotto il *Sequenziatore*approfondendo quanto riportato nel documento di Specifica Tecnica

1.2 Glossario

Al fine di facilitare la comprensione del seguente documento, ed in generale di ogni documento che verrà fornito da parte del team *Sirius*, è stato creato appositamente un glossario (*Glossario_v2.0.0.pdf*) contenente la definizione dei termini più complessi o di quelli che necessitano un approfondimento. Questi vocaboli sono contrassegnati in ogni documento dal pedice G ($_G$).

2 Standard di progetto

3 Specifica della componente view

4 Specifica della componente presenter

Questa componente racchiude tutti i metodi che gestiscono le varie operazioni offerte dall' applicazione il *Sequenziatore*e viene suddivisa in due parti: Client e Server

4.1 Client

4.2 Server

Questa componente è incaricata di gestire la comunicazione con il client e di elaborarne le richieste restituendo i dati richiesti e quando necessario interroga la componente model per ottenere i dati dal database. Tale componente è composta dalle classi:

- com.sirius.sequenziatore.server.presenter.common.SignUpController
- com.sirius.sequenziatore.server.presenter.common.LoginController
- com.sirius.sequenziatore.server.presenter.common.StepInfoController
- com.sirius.sequenziatore.server.presenter.common.ProcessInfoController
- com.sirius.sequenziatore.server.presenter.processowner.StepController
- com.sirius.sequenziatore.server.presenter.processowner.ProcessController
- com.sirius.sequenziatore.server.presenter.processowner.ApproveStepController

- com.sirius.sequenziatore.server.presenter.user.AccountController
- com.sirius.sequenziatore.server.presenter.user.UserStepController
- com.sirius.sequenziatore.server.presenter.user.UserProcessController
- com.sirius.sequenziatore.server.presenter.user.ReportController

Nella prossime sessioni verranno trattate in dettaglio le seguenti classi dividendo l' esposizione per *package*, si evidenzia come la voce mappatura base sia l' estensione della mappatura su cui si programma il sistema che sarà *localhost:8080/sequenziatore/* , quindi tutte le mappature base saranno da considerarsi come aggiunte a seguito di /sequenziatore/ e successivamente le varie varianti dei metodi. Tutte le classi *controller* del *presenter* dovranno essere marcate come *@Controller* per essere riconosciute in modo corretto da Spring.

4.2.1 Package com.sirius.sequenziatore.server.presenter.common

IMMAGINE DEL PACKAGE All' interno di questa sezione verranno trattate tutte le classi contenute nel package *common*.

4.2.1.1 Classe SignUpController

Descrizione Questa classe dovrà gestire tutte le richieste di registrazione al sistema, sarà incaricata di inserire i dati nel database e di avvertire il client della riuscita della registrazione.

Mappatura base: *signup*

Attributi:

Metodi:
- +RegisterUser(Utente toBeRegistered) , questo metodo gestirà un metodo **POST** e restituirà un errore qual' ora ci siano stati problemi nella registrazione;

4.2.1.2 LoginController

Descrizione Questa classe gestirà le richieste di *log in*, dovrà controllare se l' utente esiste nel sistema e se le credenziali d' accesso siano corrette;

Mappatura base: *login*

Attributi:

Metodi:
- +CheckLogin(Utente toBeLogged) , questo metodo gestirà un metodo di tipo **POST** , controllerà le credenziali di accesso e dovrà restituire un errore qual' ora ci siano stati problemi nella login;

4.2.1.3 StepInfoController

Descrizione Questa classe restituirà lo scheletro, quindi la composizione del passo richiesto;

Mappatura base: $\backslash step \backslash \{id\}$

Attributi:

Metodi: •

4.2.1.4 ProcessInfoController

Descrizione

Mappatura base: \

Attributi:

Metodi: •

5 Specifica della componente model