



Specifica Tecnica

Gruppo SWEet BIT – Progetto SWEDesigner

Informazioni sul documento

Versione	1.0.0
Redazione	Santimaria Davide Massignan Fabio
Verifica	Massignan Fabio Bodian Malick
Approvazione	Pilò Salvatore
Uso	Esterno
Distribuzione	Prof. Tullio Vardanega Prof. Riccardo Cardin Gruppo SWEet BIT Zucchetti S.p.A.

Descrizione

Questo documento descrive la specifica tecnica e l'architettura del prodotto sviluppato dal gruppo SWEet BIT per la realizzazione del progetto SWEDesigner.

Versioni del documento

Versione	Data	Persone coinvolte	Descrizione
1.3.0	2017/03/22	Pilò Salvatore	Approvazione Documento
1.1.0	2017/02/27	Massignan Fabio	Verifica Documento
1.0.1	2017/02/24	Santimaria Davide	Stesura Riunione Esterna 2017/02/23
1.0.0	2017/05/02	Santimaria Davide	Creazione struttura documento

Indice

1	Introduzione	4
1.1	Scopo del documento	4
1.2	Scopo del prodotto	4
1.3	Glossario	4
1.4	Riferimenti	4
2	Tecnologie Utilizzate	6
2.1	Node.js	6
2.2	Express.js	6
2.3	Accettazione dei componenti	6
2.4	MongoDB	6
2.5	Mongoose	6
3	Descrizione architettura	7
3.1	Metodo e formalismo di specifica	7
3.2	Architettura generale	7
3.3	Accettazione dei componenti	7
3.4	Interfaccia REST-like	7
4	Back-end	8
4.1	Interfaccia REST	8
4.2	Descrizione packages e classi	8
4.3	Scenari	9
4.4	Descrizione librerie aggiuntive	9
5	Front-end	10
5.1	Descrizione packages e classi	10
6	Diagrammi delle attività	11
6.1	Applicazione SWEDesigner	11
7	Stime di fattibilità e di bisogno e di risorse	12
8	Design pattern	13
8.1	Design Pattern Architeturali	13
8.2	Design Pattern Creazionali	13
8.3	Design Pattern Strutturali	13
8.4	Design Pattern Comportamentali	13
9	Tracciamento	14
9.1	Tracciamento componenti - requisiti	14
9.2	Tracciamento requisiti - componenti	14

10 Appendici	15
A Descrizione Design Pattern	15
A.1 Design Pattern Architettureali	15
A.2 Design Pattern Creazionali	15
A.3 Design Pattern Strutturali	15
A.4 Design Pattern Comportamentali	15

1 Introduzione

1.1 Scopo del documento

Questo documento ha come scopo quello di definire la *progettazione ad alto livello_G* per il prodotto. Verrà presentata la struttura generale secondo la quale saranno organizzate le varie componenti software e i *Design Pattern_G* utilizzati nella creazione del prodotto SWEDesigner. Verrà dettagliato il tracciamento tra le componenti software individuate ed i requisiti.

1.2 Scopo del prodotto

Lo scopo del progetto è la realizzazione di una *Web App_G* che fornisca all'*Utente_G* un *UML_G Designer_G* con il quale riuscire a disegnare correttamente *Diagrammi_G* delle *Classi_G* e descrivere il comportamento dei *Metodi_G* interni alle stesse attraverso l'utilizzo di *Diagrammi_G* delle attività. La *Web App_G* permetterà all'*Utente_G* di generare *Codice_G Java_G* dall'insieme dei *diagrammi classi_G* e dei rispettivi *metodi_G*.

1.3 Glossario

Con lo scopo di evitare ambiguità di linguaggio e di massimizzare la comprensione dei documenti, il gruppo ha steso un documento interno che è il *Glossario v1.2.0*. In esso saranno definiti, in modo chiaro e conciso i termini che possono causare ambiguità o incomprensione del testo.

1.4 Riferimenti

1.4.1 Normativi

- **Capitolato d'Appalto C6: SWEDesigner**
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C6p.pdf>
- **Norme di Progetto:** *Norme di Progetto v1.2.0*.
- **Analisi dei Requisiti:** *Analisi dei Requisiti v1.2.0*.

1.4.2 Informativi

- Slide dell'insegnamento Ingegneria del Software modulo A:
<http://www.math.unipd.it/~tullio/IS-1/2016/>.

2 Tecnologie Utilizzate

2.1 Node.js

2.2 Express.js

2.3 Accettazione dei componenti

2.4 MongoDB

2.5 Mongoose

3 Descrizione architettura

3.1 Metodo e formalismo di specifica

3.2 Architettura generale

3.3 Accettazione dei componenti

3.4 Interfaccia REST-like

3.4.1 Back-end

3.4.2 Front-end

4 Back-end

4.1 Interfaccia REST

4.2 Descrizione packages e classi

4.2.1 Back-end

4.2.1.1 Informazioni sul package

4.2.2 Back-end::Lib

4.2.2.1 Informazioni sul package

4.2.3 Back-end::Lib::AuthModel

4.2.3.1 Informazioni sul package

4.2.3.2 Classi

4.2.4 Back-end::Lib::Whatever

4.2.4.1 Informazioni sul package

4.2.4.2 Classi

4.3 Scenari

4.3.1 Gestione generale delle richieste

4.3.2 Fallimento vincolo "utente autenticato"

4.3.3 Fallimento vincolo "utente non autenticato"

4.3.4 Richiesta POST /login

4.3.5 Richiesta DELETE /logout

4.4 Descrizione librerie aggiuntive

5 Front-end

5.1 Descrizione packages e classi

5.1.1 Front-end

5.1.1.1 Informazioni sul package

5.1.2 Front-end::Controllers

5.1.2.1 Informazioni sul package

5.1.2.2 Classi

5.1.3 Front-end::Services

5.1.3.1 Informazioni sul package

5.1.3.2 Classi

5.1.4 Front-end::Model

5.1.4.1 Informazioni sul package

5.1.4.2 Classi

6 Diagrammi delle attività

6.1 Applicazione SWEDwsigner

6.1.1 Attività principali

6.1.2 Registrazione

6.1.3 Recupero password

6.1.4 Login

6.1.5 Modifica profilo

6.1.6 Altro

7 Stime di fattibilità e di bisogno e di risorse

8 Design pattern

8.1 Design Pattern Architetture

8.1.1 MVVM

8.1.2 Dependency Injection

8.2 Design Pattern Creazionali

8.2.1 Factory ad esempio

8.3 Design Pattern Strutturali

8.3.1 Decorator

8.3.2 Faccade

8.4 Design Pattern Comportamentali

8.4.1 Observer

8.4.2 Command

9 Tracciamento

9.1 Tracciamento componenti - requisiti

9.2 Tracciamento requisiti - componenti

10 Appendici

A Descrizione Design Pattern

A.1 Design Pattern Architetture

A.1.1 MVVM

A.1.2 Dependency Injection

A.2 Design Pattern Creazionali

A.2.1 Factory ad esempio

A.3 Design Pattern Strutturali

A.3.1 Decorator

A.3.2 Facede

A.4 Design Pattern Comportamentali

A.4.1 Observer

A.4.2 Command