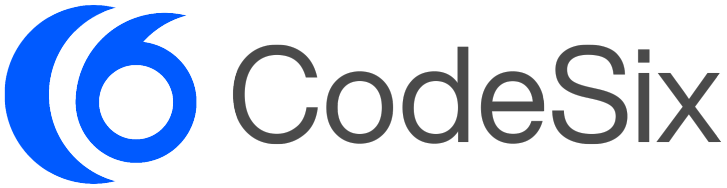


Specifica Architetturale

Progetto Ingegneria Del Software

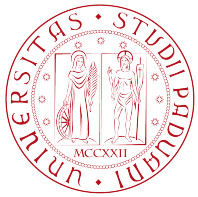
Versione: 1.0.0

Albertin Enrico
Davide Spada
Bettin Michele

Marcatti Pietro
Marco Andrea Limongelli
Matteo Raccanello

Dipartimento di Matematica
Università degli Studi di Padova

April 26, 2022



Registro delle Modifiche

Versione	Modifica	Ruolo	Esecutore	Data
1.0.0	Approvazione del documento	Responsabile	Marcatti Pietro	19/04/22
0.1.0	Verifica complessiva	Verificatore	Raccanello Matteo	16/04/22
0.0.5	Stesura sezione Soddisfacimento dei requisiti[3]	Analista, Verificatore	Spada Davide,Enrico Albertin	15/04/22
0.0.4	Stesura sezioni Architettura del prodotto [2.3, 2.4, 2.5]	Analista, Verificatore	Limongelli Marco Andrea, Marcatti Pietro	12/04/22
0.0.3	Stesura sezioni Architettura del prodotto [2.1 e 2.2]	Analista, Verificatore	Spada Davide, Enrico Albertin	10/04/22
0.0.2	Stesura sezione Introduzione [1]	Analista, Verificatore	Spada Davide, Raccanello Matteo	07/04/22
0.0.1	Stesura iniziale dello scheletro del documento e verifica	Analista, Verificatore	Spada Davide, Raccanello Matteo	04/04/22

Contents

1 Introduzione **4**
1.1 Scopo del prodotto . . . 4
1.2 Scopo del prodotto . . . 4
1.3 Glossario . . . 4
1.4 Riferimenti . . . 4
1.4.1 Riferimenti normativi . . . 4
1.4.2 Riferimenti informativi . . . 4

2 Architettura del prodotto **5**
2.1 Design patterns . . . 5
2.2 Descrizione generale . . . 5
2.2.1 Model-View-ViewModel . . . 5
2.2.2 Strategy . . . 5
2.2.3 Observer . . . 5
2.3 Diagramma dei package . . . 6
2.4 Diagrammi delle classi . . . 7
2.5 Diagrammi di sequenza . . . 10

3 Soddisfacimento dei requisiti **13**
3.1 Tabella di soddisfacimento dei requisiti . . . 13
3.2 Valutazione di soddisfacimento dei requisiti funzionali . . . 15

1 Introduzione

1.1 Scopo del prodotto

Lo scopo del presente documento è quello di descrivere e motivare le scelte architetturali prese dal gruppo CodeSix per quanto riguarda la fase di progettazione e codifica del prodotto software. Sono quindi riportati i diagrammi dei package, delle classi e di sequenza per descrivere in maniera formale e sintetica l'architettura e le funzionalità implementate nel prodotto. E' infine presente la sezione dedicata ai requisiti che il gruppo ha soddisfatto, in modo tale da dare una panoramica sullo stato di avanzamento del prodotto.

1.2 Scopo del prodotto

La sicurezza informatica, specialmente in applicazioni dove è necessaria l'autenticazione, è un aspetto fondamentale. Pertanto è necessario attuare dei metodi per migliorarla. Il capitolato C5 ha come obiettivo la creazione di un'applicazione per visualizzare graficamente i dati relativi ai login per accedere a un sistema. Lo scopo del prodotto sarà quello di fornire all'utente diverse tipologie di grafici e algoritmi di riduzione dimensionale, in modo che possa rilevare eventuali cluster ed eventualmente capire quali login sono sospetti grazie all'osservazione di tali grafici.

1.3 Glossario

Al fine di minimizzare le ambiguità il team ha prodotto ***Glossario 2.0.0*** che raccoglie termini di particolare importanza o ai quali è assegnato un significato particolare individuati nel testo da '*' ad apice.

1.4 Riferimenti

1.4.1 Riferimenti normativi

- **Capitolato di appalto C5 - Login Warrior**
- *Analisi dei requisiti 3.0.0*

1.4.2 Riferimenti informativi

- **Model-View patterns - Materiale didattico del corso di Ingegneria del Software**
- **Diagrammi dei package - Materiale didattico del corso di Ingegneria del Software**
- **Diagrammi di sequenza - Materiale didattico del corso di Ingegneria del Software**
- **Diagrammi delle classi - Materiale didattico del corso di Ingegneria del Software**

2 Architettura del prodotto

2.1 Design patterns

Sono di seguito presentati i design patterns implementati nel prodotto software.

- Model-View-ViewModel*
- Strategy*
- Observer*

2.2 Descrizione generale

2.2.1 Model-View-ViewModel

Il pattern architetturale al quale il gruppo CodeSix ha scelto di attenersi è il *Model-View-ViewModel**. Questa particolare tipologia di pattern permette di scrivere codice facilmente manutenibile e riusabile grazie al forte disaccoppiamento tra la logica di presentazione (UI) e di business. Inoltre questo pattern è risultato essere il migliore per essere utilizzato in React, libreria per la creazione di interfacce utente single-page che renderizza le componenti in base al loro stato interno ed unicamente ai cambiamenti che avvengono su queste.

Per permettere il passaggio dei dati dal Model alle varie componenti grafiche è stato utilizzato il Context React, al quale viene passato un'istanza del ViewModel (che nel nostro caso corrisponde al RootStore). Questo permette di accedere al valore del ViewModel in qualsiasi parte della View evitando di passare i dati di componente in componente. Ciò avverrebbe altrimenti attraverso le props, cioè gli argomenti della vista, che verrebbero passati fino a livelli potenzialmente profondi della gerarchia. Per evitare questo viene creata un'istanza del ViewModel passata ad un Context.Provider, ossia il contenitore della View. All'interno di questo ogni componente può utilizzare un hook per accedere al Context React ed utilizzare il valore più recente del ViewModel. Il Contex React evita quindi di passare i dati per molti componenti rischiando, nel caso peggiore, di doverli utilizzare nell'ultimo livello della gerarchia.

2.2.2 Strategy

Per quanto riguarda il processo di riduzione dimensionale viene utilizzato il design pattern strategy*. La scelta di implementare questa tipologia di design pattern deriva dall'osservazione secondo la quale l'unica principale differenza era determinata dalla tipologia di algoritmo applicato ai fini della riduzione. Esso permette quindi di definire una famiglia di algoritmi e isolarli all'interno di un oggetto che ha permesso di renderli interscambiabili dinamicamente ed evitare duplicazione di codice permettendo quindi l'integrazione di ulteriori algoritmi di riduzione dimensionale derivando nuove classi concrete dall'interfaccia esposta. Nel nostro caso abbiamo:

- **DimReductor.js**: è la classe concreta che invoca la ConcreteStrategy richiesta del client;
- **ReductionAlgorithm.js**: è l'interfaccia comune a tutti gli algoritmi ed utilizzata da DimReductor.js per impostarne i parametri ed invocarli;
- **tSNEAlgorithm.js, UMAPAlgorithm.js**: rappresentano gli algoritmi concreti di riduzione dimensionale che implementano ReductionAlgorithm.js.

2.2.3 Observer

Inoltre, per poter fare in modo che una componente della View si renderizzi non solo al cambiamento del suo stato interno ma anche al cambiamento dei dati nel Model, abbiamo utilizzato la libreria *Mobx**. Questa

permette di implementare l'*observer pattern** la quale fornisce la possibilità di segnare delle classi (o attributi di esse) come ”observable” e di costruire dei componenti della View come ”observer”. Questi ultimi vengono automaticamente ri-renderizzati al cambiamento di un qualsiasi attributo observable.

2.3 Diagramma dei package

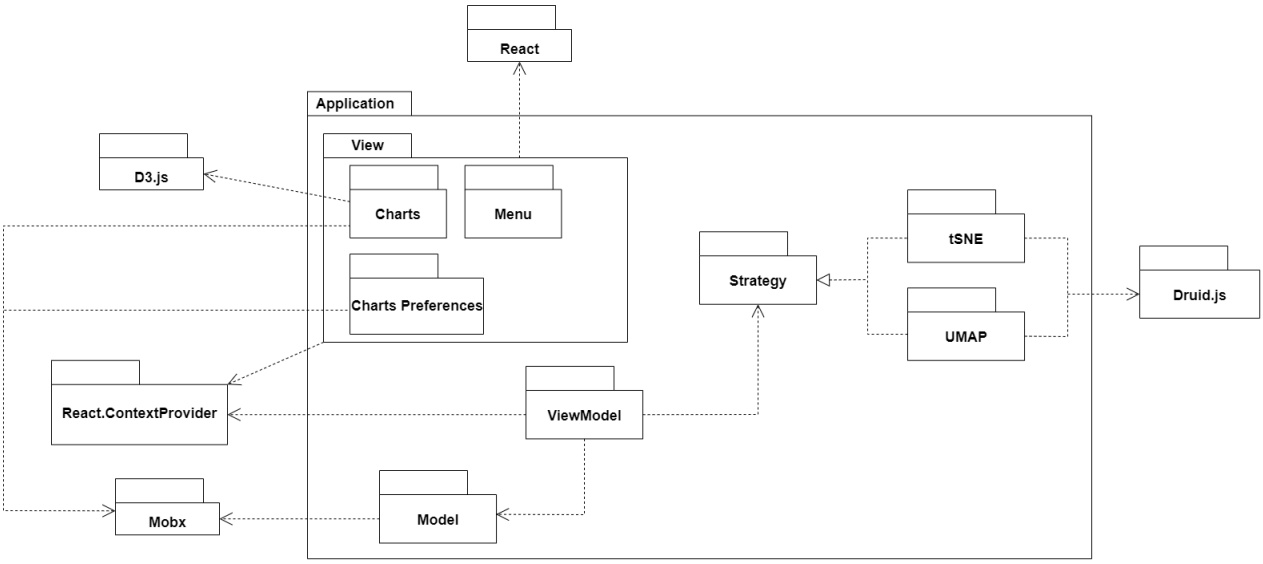


Figure 1: Diagramma dei package

2.4 Diagrammi delle classi

Sono di seguito riportati degli zoom del diagramma delle classi in particolare su RootStore, View e Menu.

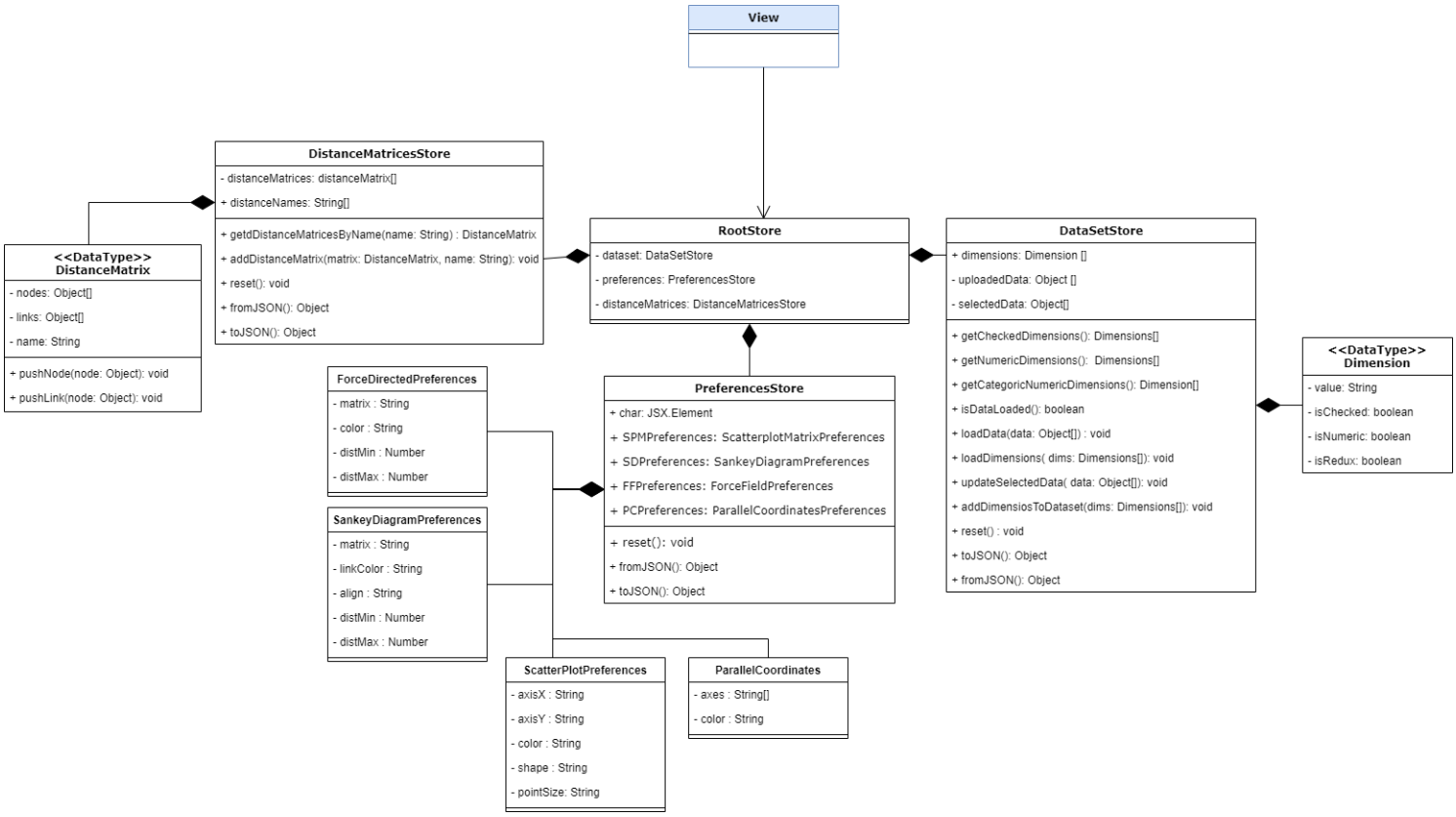


Figure 2: Diagramma delle classi - Zoom su RootStore

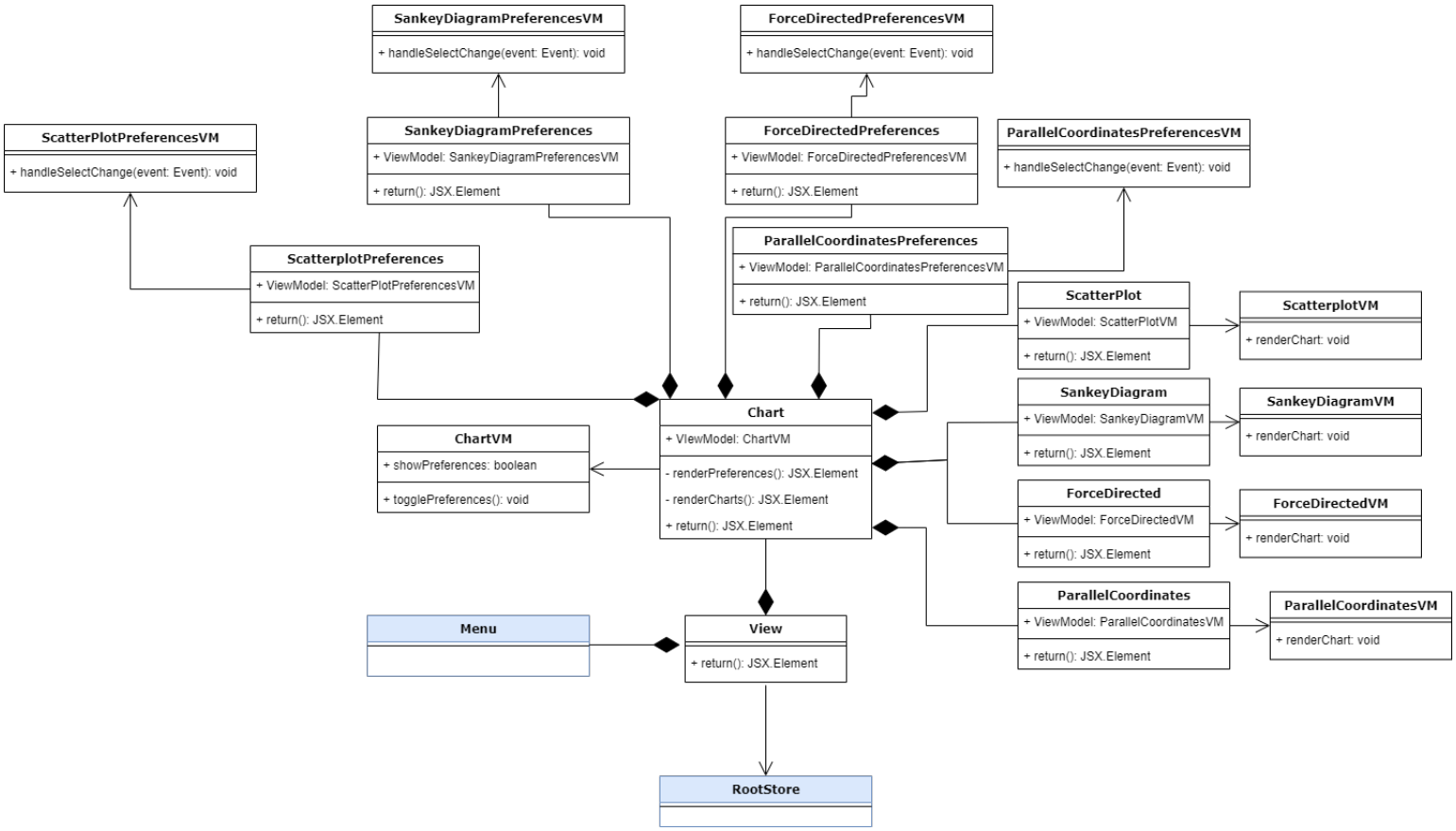


Figure 3: Diagramma delle classi - Zoom sulla View

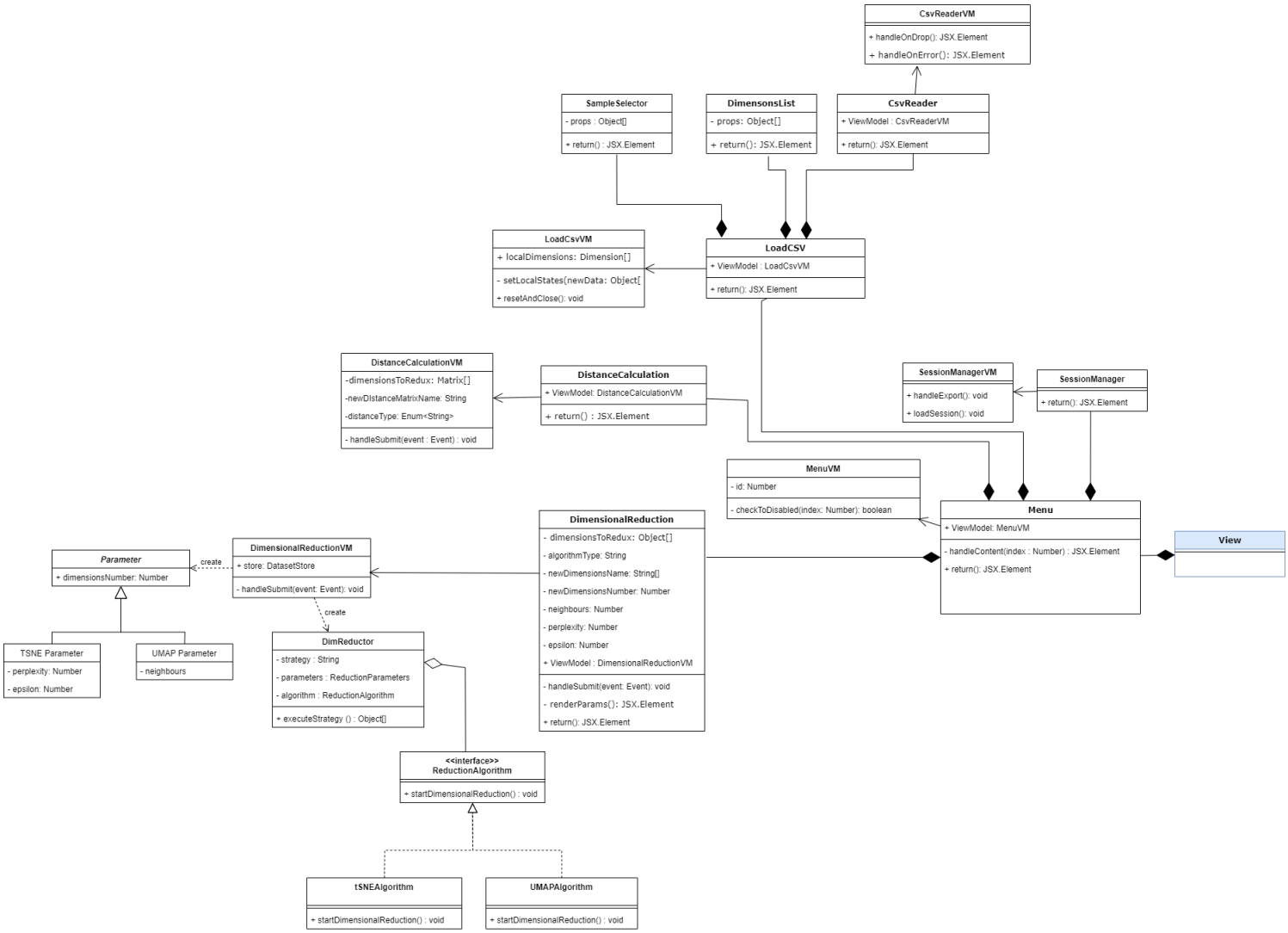


Figure 4: Diagramma delle classi - Zoom sul Menu

2.5 Diagrammi di sequenza

Sono di seguito presentati i diagrammi di sequenza per le tre operazioni salienti che caratterizzano il prodotto software.

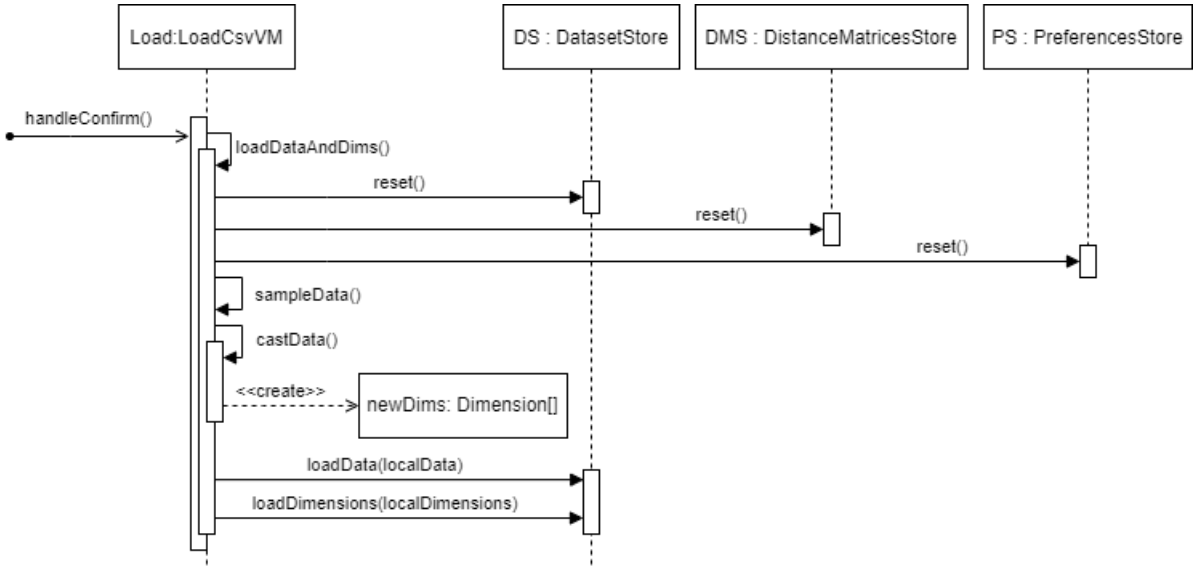


Figure 5: Diagramma di sequenza per il caricamento di un file .csv

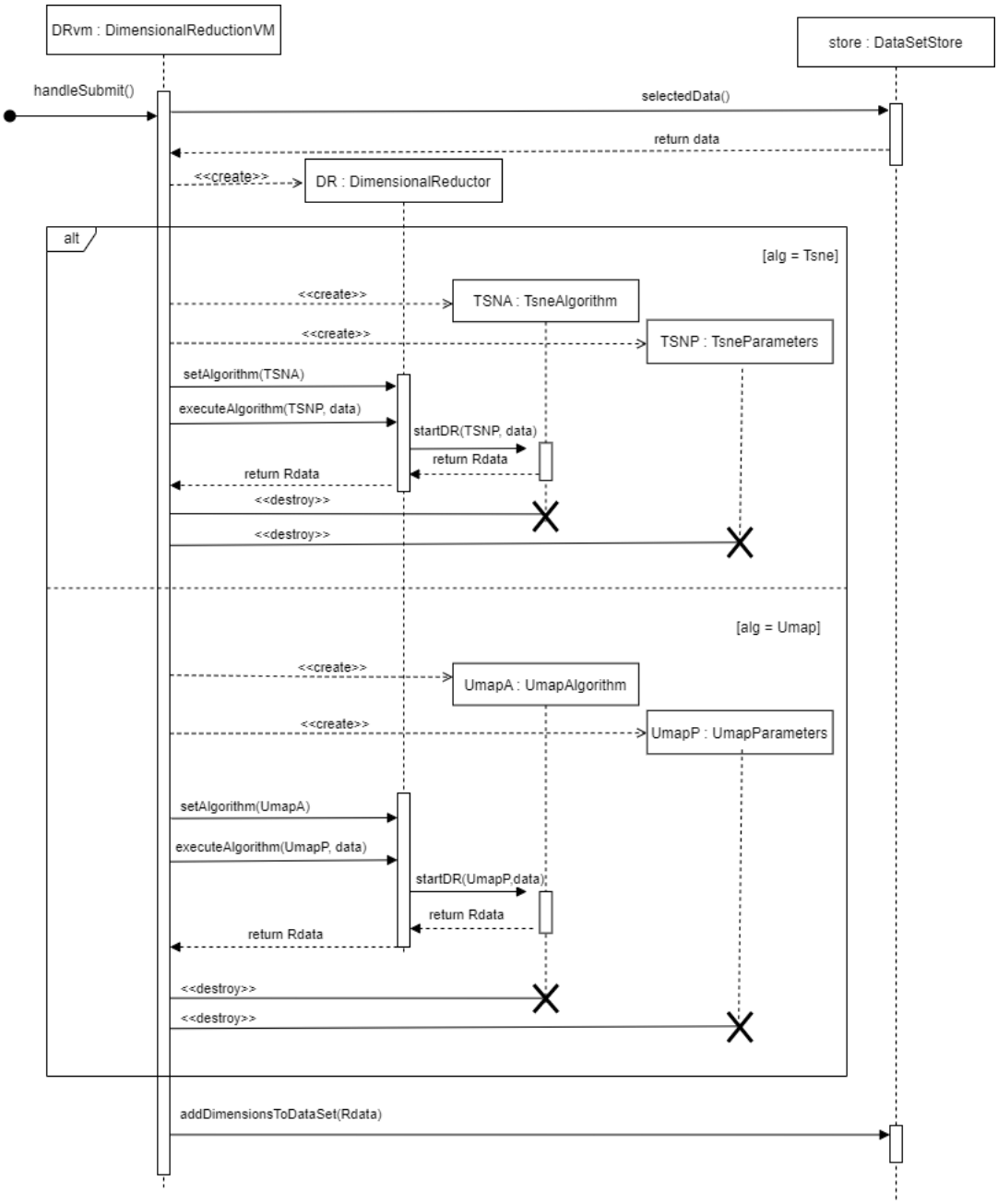


Figure 6: Diagramma di sequenza per l'operazione di riduzione dimensionale

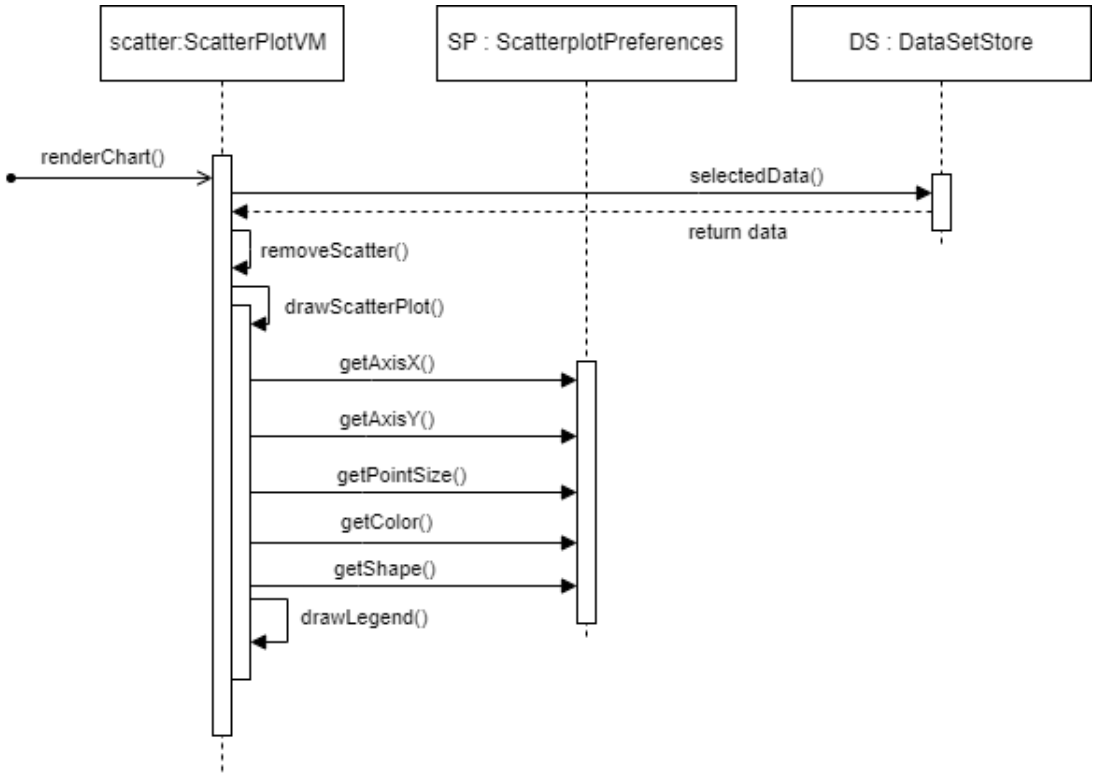


Figure 7: Diagramma di sequenza per la visualizzazione dello scatterplot

3 Soddisfacimento dei requisiti

3.1 Tabella di soddisfacimento dei requisiti

Codice	Classe	Soddisfacimento
R1F1	OB	Soddisfatto
R1F1.1	OB	Soddisfatto
R2F1.2	DE	Soddisfatto
R1F2	OB	Soddisfatto
R1F2.1	OB	Soddisfatto
R1F2.2	OB	Soddisfatto
R1F2.2.1	OB	Soddisfatto
R1F3	OB	Soddisfatto
R2F3.1	DE	Soddisfatto
R2F3.1.1	DE	Soddisfatto
R2F3.1.2	DE	Soddisfatto
R1F3.2	OB	Soddisfatto
R1F3.2.1	OB	Soddisfatto
R2F3.2.2	DE	Soddisfatto
R2F3.2.3	DE	Soddisfatto
R2F4	DE	Soddisfatto
R2F4.1	DE	Soddisfatto
R2F4.2	DE	Soddisfatto
R2F4.3	DE	Soddisfatto
R2F4.4	DE	Soddisfatto
R2F4.4.1	DE	Soddisfatto
R2F4.4.1.1	DE	Soddisfatto
R2F4.4.1.2	DE	Soddisfatto
R2F4.4.2	DE	Soddisfatto
R2F4.4.2.1	DE	Soddisfatto
R2F4.4.2.2	DE	Soddisfatto

R2F4.4.2.3	DE	Soddisfatto
R1F5	OB	Soddisfatto
R1F5.1	OB	Soddisfatto
R1F5.2	OB	Soddisfatto
R1F6	OB	Soddisfatto
R1F6.1	OB	Soddisfatto
R1F6.2	OB	Soddisfatto
R1F6.3	OB	Soddisfatto
R1F6.4	OB	Soddisfatto
R1F7	OB	Soddisfatto
R1F7.1	OB	Soddisfatto
R1F7.1.1	OB	Soddisfatto
R1F7.1.2	OB	Soddisfatto
R1F7.1.3	OB	Soddisfatto
R1F7.1.4	OB	Soddisfatto
R1F7.2	OB	Soddisfatto
R1F7.2.1	OB	Soddisfatto
R1F7.2.2	OB	Soddisfatto
R1F7.3	OB	Soddisfatto
R1F7.3.1	OB	Soddisfatto
R1F7.3.2	OB	Soddisfatto
R1F7.3.3	OB	Soddisfatto
R1F7.4	OB	Soddisfatto
R1F7.4.1	OB	Soddisfatto
R1F7.4.2	OB	Soddisfatto
R1F7.4.3	OB	Soddisfatto
R2F8	DE	Soddisfatto
R2F8.1	DE	Soddisfatto
R1F9	OB	Soddisfatto
R1F10	OB	Soddisfatto

R2F11	DE	Non soddisfatto
R3F12	OP	Non soddisfatto
R3F13	OP	Non soddisfatto

Table 2: Tabella soddisfacimento dei requisiti

3.2 Valutazione di soddisfacimento dei requisiti funzionali

Tipologia	**Rapporto**	**Percentuale**
Obbligatorio	36/36	100%
Desiderabile	20/21	95%
Opzionale	0/2	0%

Table 3: Soddisfacimento requisiti funzionali