**Production Scheduler**

Documentazione



**Docente:** Patrizia Scandurra

**Studenti:** Luca Fioretti, Stefano Sanvito

**Esame:** Informatica III (Progettazione e Algoritmi)

**Corso:** Ing. Informatica Magistrale

Indice

[1. Prefazione 3](#_Toc72267649)

[1.1 Versioni 3](#_Toc72267650)

[1.2 Documentazione Versioni 3](#_Toc72267651)

[2. Introduzione 4](#_Toc72267652)

[2.1 Teoria della schedulazione 4](#_Toc72267653)

[2.2 Schedulazione della produzione 4](#_Toc72267654)

[2.3 Scopo applicazione 4](#_Toc72267655)

[2.4 Toolchain 6](#_Toc72267656)

[3. Iterazione 0 7](#_Toc72267657)

[3.1 Architettura Hardware 7](#_Toc72267658)

[3.2 Architettura Software 7](#_Toc72267659)

[3.3 Requisiti funzionali 9](#_Toc72267660)

[3.4 Casi d’uso 10](#_Toc72267661)

[**3.4.1 UC1 – Autenticazione** 10](#_Toc72267662)

[**3.4.2 UC2 – Visualizzazione Dati Utenti** 10](#_Toc72267663)

[**3.4.3 UC3 – Creazione/Modifica Dati Utenti** 10](#_Toc72267664)

[**3.4.4 UC4 – Visualizzazione Ruoli Utenti** 11](#_Toc72267665)

[**3.4.5 UC5 – Creazione/Modifica Ruoli Utenti** 11](#_Toc72267666)

[**3.4.6 UC6 – Eliminazione Ruolo Utente** 11](#_Toc72267667)

[**3.4.7 UC7 – Visualizzazione Prodotti** 12](#_Toc72267668)

[**3.4.8 UC8 – Creazione/Modifica Prodotto** 12](#_Toc72267669)

[**3.4.9 UC9 – Eliminazione Prodotto** 12](#_Toc72267670)

[**3.4.10 UC10 – Visualizzazione Tipi Macchina** 13](#_Toc72267671)

[**3.4.11 UC11 – Creazione/Modifica Tipo Macchina** 13](#_Toc72267672)

[**3.4.12 UC12 – Eliminazione Tipo Macchina** 14](#_Toc72267673)

[**3.4.13 UC13 – Visualizzazione Macchine** 14](#_Toc72267674)

[**3.4.14 UC14 – Creazione/Modifica Macchina** 14](#_Toc72267675)

[**3.4.15 UC15 – Eliminazione Macchina** 15](#_Toc72267676)

[**3.4.16 UC16 – Visualizzazione Ordini Produzione non chiusi** 15](#_Toc72267677)

[**3.4.17 UC17 – Creazione/Modifica Ordine Produzione** 15](#_Toc72267678)

[**3.4.18 UC18 – Eliminazione Ordine di Produzione** 16](#_Toc72267679)

[**3.4.19 UC19 – Visualizzazione Ordini Produzione chiusi** 16](#_Toc72267680)

[**3.4.20 UC20 – Chiusura Ordini Produzione in fase di produzione** 17](#_Toc72267681)

[3.5 Programma di test 19](#_Toc72267682)

[4. Iterazione 1 20](#_Toc72267683)

[4.1 UML Class Diagram 21](#_Toc72267684)

# **1. Prefazione**

## **1.1 Versioni**

1. Versione 0.0 – 20/04/2021
2. Versione 1.0 – 15/05/2021
3. Versione 2.0 - ?

## **1.2 Documentazione Versioni**

* Versione 0.0: Analisi strumenti utilizzati per lo sviluppo dell’applicazione. Studio dei requisiti funzionali e non funzionali, modellazione dei casi d’uso iniziali e della configurazione iniziale dell’architettura software e hardware.
* Versione 1.0: Sviluppo dell’iterazione 1, implementazione all’interno dell’applicazione delle prime specifiche funzionali. Svolgimento test.
* Versione 2.0: ?

# **2. Introduzione**

## **2.1 Teoria della schedulazione**

La schedulazione è una forma di processo decisionale, che consiste nell’allocare risorse finite in modo tale che un dato obiettivo venga ottimizzato. Esempi tipici di risorse limitate sono le macchine di un’impresa manifatturiera, le ore di manodopera settimanali di un operaio o di una commessa di un negozio, le piste di atterraggio/decollo di un aeroporto. Esempi di obiettivi da ottimizzare invece sono il tempo per completare un progetto, il tempo di attesa di un aereo in volo o di un lotto da produrre.

La schedulazione ha il compito di sincronizzare e tempificare la sequenza delle operazioni. La schedulazione contiene cioè anche le informazioni temporali che specificano le tempistiche delle singole attività, la schedulazione include quindi il sequenziamento.

## **2.2** **Schedulazione della produzione**

La schedulazione della produzione è una programmazione delle attività di tipo esecutivo. È l’attività principale che un ufficio della programmazione deve compiere. Questo consente di riesaminare e riallineare costantemente i dati di produzione, cercando l’ottimizzazione riducendo errori ed inefficienze.

La schedulazione come descritto sopra è un processo decisionale che permette di creare e attribuire una sequenza di attività a ciascuna risorsa (attrezzature, macchine, operatori). Questa operazione tiene in considerazione i diversi vincoli tecnologici quali:

* Disponibilità
* Priorità
* Capacità produttiva

Gli obiettivi principali del processo di schedulazione sono:

* La minimizzazione dei costi di produzione
* Il raggiungimento dell’efficienza produttiva

## **2.3 Scopo applicazione**

Lo scopo dell’applicazione che si andrà a realizzare è quello di fornire ad una piccola azienda manifatturiera un sistema che possa aiutarla a schedulare i propri ordini di produzione in modo di minimizzare il tempo di produzione degli stessi e ottimizzare l’efficienza dell’attività produttiva. In particolare, gli operatori potranno gestire i dati relativi alle macchine produttive, ai prodotti che si vogliono realizzare e agli ordini di produzione programmati. Verrà inoltre fornita la possibilità di visualizzare gli ordini di produzione schedulati con i rispettivi tempi di completamento per consentire all’operatore di valutare eventuali ritardi e le conseguenti soluzioni che potrebbero essere prese per evitarli.

## **2.4 Toolchain**

Per lo sviluppo dell’applicazione è stata scelta la seguente toolchain:

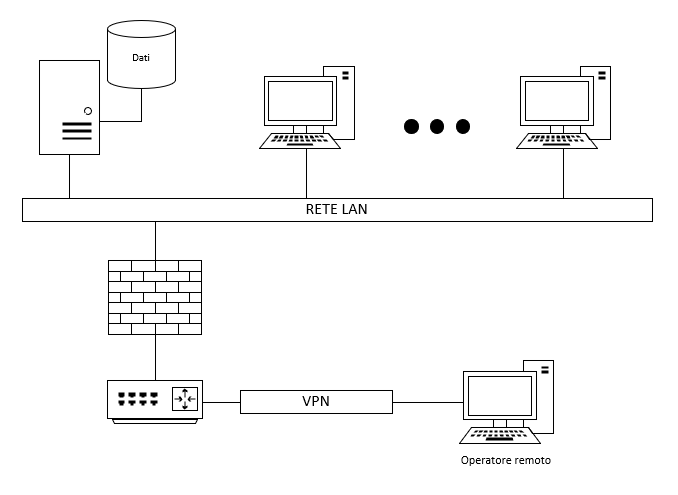
* **GitHub**, utilizzato per consentire il lavoro sul progetto a più persone. L’utilizzo di GitHub ha consentito anche di sviluppare su branch secondari per lo sviluppo di particolari sezioni di codice, così da conservare sempre una versione funzionante dell’applicazione nel branch “master”.
* **GitHub Desktop**, applicazione che fornisce un’interfaccia grafica per l’utilizzo di GitHub.
* **Microsoft Visio**, utilizzato per la realizzazione di diagrammi, in particolare per la creazione del modello dell’architettura software e hardware.
* **PHPStorm**, utilizzato per lo sviluppo dell’applicazione, scelto anche per la presenza di molti tool utili per velocizzare la scrittura di codice e la realizzazione di test.
* **Laravel,** framework open source di tipo MVC scritto in PHP sul quale si baserà tutta l’applicazione.
* **Psalm**, tool per l’analisi statica del codice PHP
* **PHP Unit**, testing framework per la realizzazione dei test.
* **Postman,** utilizzato per testare le API in fase di sviluppo del back-end dell’applicazione senza dover obbligatoriamente avere a disposizione il front-end.

# **3. Iterazione 0**

## **3.1 Architettura Hardware**

L’applicazione che verrà sviluppata sarà un’applicazione client/server. Di conseguenza per poter utilizzare l’applicazione sarà necessaria la presenza di un server centrale per la fruizione di tutte le funzionalità del software. Il server dovrà essere accompagnato da un database per il salvataggio di tutti i dati.

Di seguito l’architettura software necessaria per l’utilizzo dell’applicazione.



## **3.2 Architettura Software**

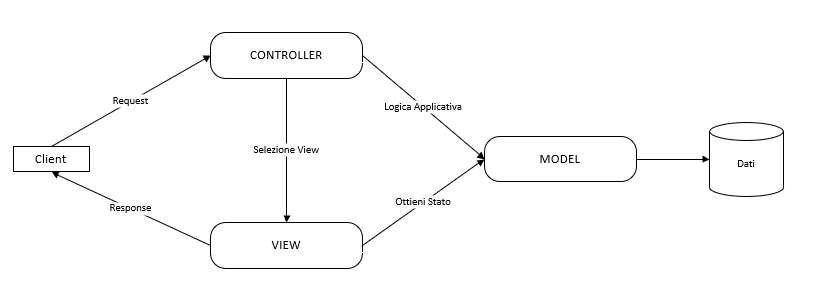
L’applicazione sarà progettata e realizzata sopra il framework Laravel; un framework open source di tipo MVC (Model View Controller) scritto in PHP.

L’utilizzo del framework Laravel impone quindi una struttura di tipo MVC (Model View Controller). Questo pattern architetturale permette di separare la logica front-end dalla logica back-end. I componenti principali di questa architettura sono:

* **Model**, contiene i metodi di accesso ai dati, è la componente che si interfaccia direttamente con il database per leggere, creare e modificare i dati.
* **View**, si occupa di visualizzare i dati all’utente e gestisce l’interazione fra quest’ultimo e l’infrastruttura sottostante.
* **Controller,** è il componente posto a metà tra il Model e la View. Contiene metodi pubblici chiamati **action** che contengono operazioni che possono interessare il Model e che portano generalmente ad un cambiamento di stato nella View.

Nell’implementazione PHP questi metodi e i loro parametri vengono messi in correlazione con gli URL delle richieste http provenienti dal componente View.

Questo pattern risulta particolarmente comodo nelle applicazioni Web perché permette lo sviluppo indipendente del front-end e del back-end portando numerosi vantaggi. Inoltre, rende il design flessibile, permette il riuso dei componenti sviluppati e semplifica le fasi di testing.



Linguaggi utilizzati:

* Back-end: PHP
* Front-end: HTML, CSS, JavaScript

## **3.3 Requisiti funzionali**

L’applicazione deve consentire all’utente di svolgere le seguenti operazioni:

* Autenticazione.
* Visualizzazione, creazione e modifica dei dati relativi agli utenti. Questa funzione dovrà essere disponibile solo agli utenti con ruolo di amministratori.
* Visualizzazione, creazione e modifica dei dati relativi ai ruoli degli utenti. Questa funzione dovrà essere disponibile solo agli utenti con ruolo di amministratori.
* Visualizzazione, creazione, modifica e cancellazione dei dati relativi alle macchine di produzione.
* Visualizzazione, creazione, modifica e cancellazione dei dati relativi alla tipologia di macchine di produzione.
* Visualizzazione, creazione, modifica e cancellazione dei dati relativi ai prodotti.
* Visualizzazione, creazione, modifica e cancellazione dei dati relativi agli ordini di produzione.

L’applicazione deve inoltre fornire le seguenti funzionalità:

* Fornire tutte le operazioni di lettura/scrittura dei dati sul database.
* Fornire tutte le operazioni di manipolazione dei dati inseriti dall’utente per poterli adattare alle esigenze del database.
* Verificare le seguenti condizioni in fase di salvataggio dei dati sul database:
  1. Una tipologia di macchina di produzione può essere eliminata se e solo se non è già assegnata ad una macchina e/o ad un prodotto.
  2. Un ordine di produzione non può essere modificato e/o eliminato quando si trova in fase di produzione o quando la sua produzione è già stata ultimata.
  3. Il tempo di produzione di un prodotto non può essere modificato quando è contenuto in un ordine in fase di produzione.
  4. Un prodotto può essere eliminato se e solo se non è assegnato ad un ordine di produzione.
* Comunicare l’esito di tutte le operazioni eseguite dall’utente.

## **3.4 Casi d’uso**

### **3.4.1 UC1 – Autenticazione**

* Descrizione: autenticazione utente nell’applicazione.
* Attori coinvolti: utente.
* Precondizioni: presenza dell’utente nel database dell’applicazione.
* Post condizioni: l’utente ha accesso alle funzioni dell’applicazione.
* Procedimento:

1. L’utente raggiunge la pagina di login dell’applicazione.
2. L’utente inserisci le sue credenziali nel form di login.
3. L’applicazione controlla se i dati dell’utente sono corretti.
4. L’utente viene reindirizzato alla homepage dell’applicazione.

* Eccezioni:
  1. Viene visualizzato l’errore prodotto (es. credenziali errate).

### **3.4.2 UC2 – Visualizzazione Dati Utenti**

* Descrizione: visualizzazione in tabella degli utenti registrati nell’applicazione.
* Attori coinvolti: utente amministratore.
* Precondizioni: l’utente deve avere il ruolo di amministratore.
* Post condizioni: i dati degli utenti registrati vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione degli utenti.
  2. L’applicazione controlla che l’utente sia un amministratore.
  3. Vengono letti i dati relativi agli utenti dal db.
  4. Vengono mostrati a schermo i dati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’utente non abbia il permesso di visualizzare i dati degli utenti.

### **3.4.3 UC3 – Creazione/Modifica Dati Utenti**

* Descrizione: creazione/modifica degli utenti registrati nell’applicazione.
* Attori coinvolti: utente amministratore.
* Precondizioni: l’utente deve avere il ruolo di amministratore.
* Post condizioni: l’utente creato viene inserito nel database. I dati dell’utente modificato vengono aggiornati nel database
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi all’utente che si vuole creare/modificare.
  2. L’applicazione controlla che l’utente sia un amministratore.
  3. Vengono inseriti/modificati i dati del nuovo utente/utente da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’utente non abbia il permesso di visualizzare i dati degli utenti.

### **3.4.4 UC4 – Visualizzazione Ruoli Utenti**

* Descrizione: visualizzazione in tabella dei ruoli utente.
* Attori coinvolti: utente amministratore.
* Precondizioni: l’utente deve avere il ruolo di amministratore.
* Post condizioni: i dati dei ruoli vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione dei ruoli.
  2. L’applicazione controlla che l’utente sia un amministratore.
  3. Vengono letti i dati relativi ai ruoli dal db.
  4. Vengono mostrati a schermo i dati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’utente non abbia il permesso di visualizzare i dati degli utenti.

### **3.4.5 UC5 – Creazione/Modifica Ruoli Utenti**

* Descrizione: creazione/modifica dei ruoli utente.
* Attori coinvolti: utente amministratore.
* Precondizioni: l’utente deve avere il ruolo di amministratore.
* Post condizioni: il ruolo creato viene inserito nel database. I dati del ruolo modificato vengono aggiornati nel database
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi al ruolo che si vuole creare/modificare.
  2. L’applicazione controlla che l’utente sia un amministratore.
  3. Vengono inseriti/modificati i dati del nuovo ruolo/ruolo da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’utente non abbia il permesso di visualizzare i dati degli utenti.

### **3.4.6 UC6 – Eliminazione Ruolo Utente**

* Descrizione: eliminazione ruolo utente.
* Attori coinvolti: utente amministratore.
* Precondizioni: l’utente deve avere il ruolo di amministratore. Il ruolo che si vuole cancellare non deve essere assegnato a nessun utente registrato nell’applicazione.
* Post condizioni: il ruolo viene eliminato dal database.
* Procedimento:
  1. L’utente seleziona il ruolo che vuole eliminare.
  2. L’applicazione controlla che l’utente sia un amministratore e che il ruolo da cancellare non sia assegnato a nessun utente registrato nell’applicazione.
  3. Viene eliminato il ruolo dal db.
  4. Viene mostrato a schermo l’esito dell’operazione.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’utente non abbia il permesso di visualizzare i dati degli utenti.
  2. Viene visualizzato un messaggio di errore qualora il ruolo non possa essere eliminato per i motivi sopra citati.

### **3.4.7 UC7 – Visualizzazione Prodotti**

* Descrizione: visualizzazione in tabella dei prodotti.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: i dati dei prodotti vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione dei dati dei prodotti.
  2. Vengono letti dal database i dati relativi ai prodotti.
  3. Vengono mostrati a schermo i dati relativi ai prodotti tramite una tabella.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora la pagina di visualizzazione dei prodotti non sia disponibile.

### **3.4.8 UC8 – Creazione/Modifica Prodotto**

* Descrizione: creazione/modifica prodotto.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: il prodotto creato viene inserito nel database. I dati del prodotto modificato vengono aggiornati nel database.
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi al prodotto che si vuole creare/modificare.
  2. L’applicazione controlla che i dati inseriti dall’utente siano coerenti.
  3. Vengono inseriti/modificati i dati del nuovo ruolo/ruolo da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’operazione di creazione/modifica non sia andata a buon fine.

### **3.4.9 UC9 – Eliminazione Prodotto**

* Descrizione: eliminazione prodotto.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato. Il prodotto che si vuole cancellare non deve essere assegnato a nessun ordine di produzione.
* Post condizioni: il prodotto viene eliminato dal database.
* Procedimento:
  1. L’utente seleziona il prodotto che vuole eliminare.
  2. L’applicazione controlla che il prodotto da cancellare non sia assegnato a nessun ordine di produzione.
  3. Viene eliminato il prodotto dal db.
  4. Viene mostrato a schermo l’esito dell’operazione.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora il prodotto non possa essere eliminato per i motivi sopra citati.

### **3.4.10 UC10 – Visualizzazione Tipi Macchina**

* Descrizione: visualizzazione in tabella dei tipi macchina.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: i dati dei tipi macchina vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione dei dati dei tipi macchina.
  2. Vengono letti dal database i dati relativi ai tipi macchina.
  3. Vengono mostrati a schermo i dati relativi ai tipi macchina tramite una tabella.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora la pagina di visualizzazione dei tipi macchina non sia disponibile.

### **3.4.11 UC11 – Creazione/Modifica Tipo Macchina**

* Descrizione: creazione/modifica tipo macchina.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: il tipo macchina creato viene inserito nel database. I dati del tipo macchina modificato vengono aggiornati nel database.
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi al tipo macchina che si vuole creare/modificare.
  2. L’applicazione controlla che i dati inseriti dall’utente siano coerenti.
  3. Vengono inseriti/modificati i dati del nuovo tipo macchina/tipo macchina da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’operazione di creazione/modifica non sia andata a buon fine.

### **3.4.12 UC12 – Eliminazione Tipo Macchina**

* Descrizione: eliminazione tipo macchina.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato. Il tipo macchina che si vuole cancellare deve rispettare i seguenti requisiti:
  1. Non deve essere assegnato a nessun prodotto.
  2. Non deve essere assegnato a nessuna macchina.
* Post condizioni: il tipo macchina viene eliminato dal database.
* Procedimento:
  1. L’utente seleziona il tipo macchina che vuole eliminare.
  2. L’applicazione controlla che il tipo macchina da cancellare rispetti i requisiti sopra esposti.
  3. Viene eliminato il tipo macchina dal db.
  4. Viene mostrato a schermo l’esito dell’operazione.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora il tipo macchina non possa essere eliminato per i motivi sopra citati.

### **3.4.13 UC13 – Visualizzazione Macchine**

* Descrizione: visualizzazione in tabella delle macchine.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: i dati delle macchine vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione dei dati delle macchine.
  2. Vengono letti dal database i dati relativi alle macchine
  3. Vengono mostrati a schermo i dati relativi alle macchine tramite una tabella.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora la pagina di visualizzazione delle macchine non sia disponibile.

### **3.4.14 UC14 – Creazione/Modifica Macchina**

* Descrizione: creazione/modifica macchina.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: la macchina creata viene inserita nel database. I dati della macchina modificata vengono aggiornati nel database.
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi alla macchina che si vuole creare/modificare.
  2. L’applicazione controlla che i dati inseriti dall’utente siano coerenti.
  3. Vengono inseriti/modificati i dati della nuova macchina/ macchina da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’operazione di creazione/modifica non sia andata a buon fine.

### **3.4.15 UC15 – Eliminazione Macchina**

* Descrizione: eliminazione macchina.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: la macchina viene eliminata dal database.
* Procedimento:
  1. L’utente seleziona la macchina che vuole eliminare.
  2. Viene eliminata la macchina dal db.
  3. Viene mostrato a schermo l’esito dell’operazione.

### **3.4.16 UC16 – Visualizzazione Ordini Produzione non chiusi**

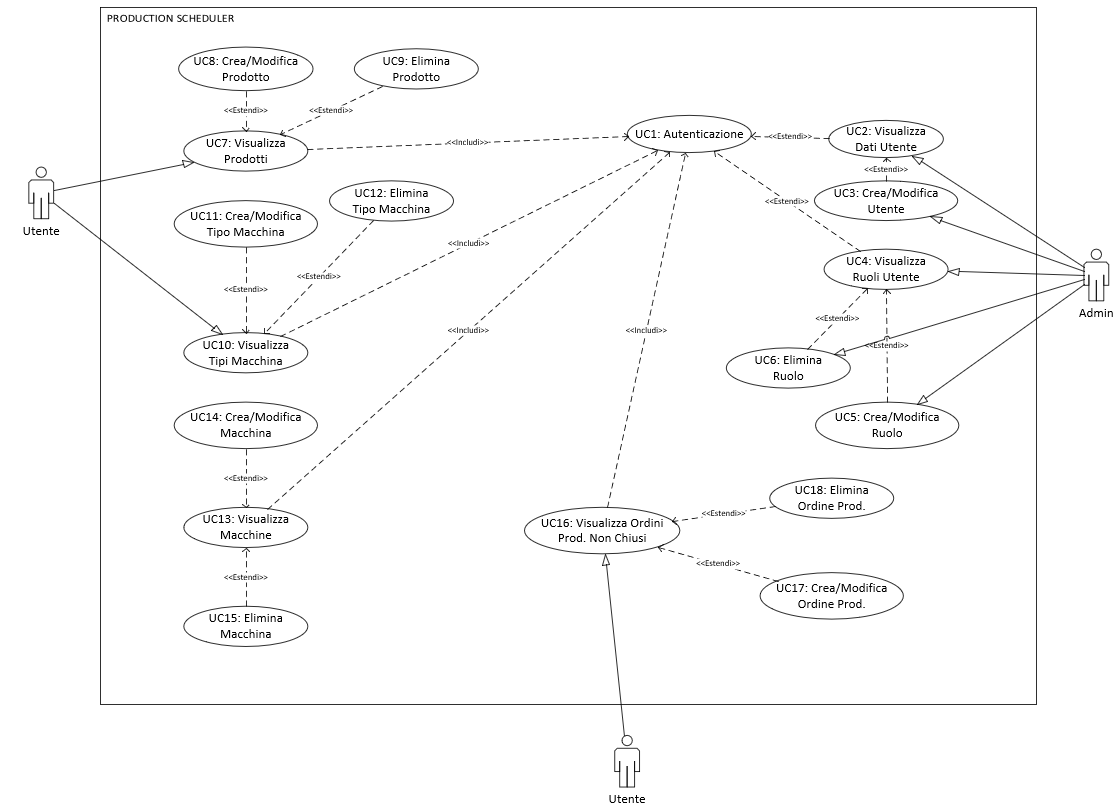
* Descrizione: visualizzazione in tabella degli ordini di produzione non chiusi.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato.
* Post condizioni: i dati degli ordini di produzione non chiusi vengono visualizzati in una tabella.
* Procedimento:
  1. L’utente richiede la visualizzazione degli ordini di produzione non chiusi.
  2. Vengono letti dal database i dati relativi agli ordini di produzione non chiusi.
  3. Vengono mostrati a schermo i dati relativi agli ordini di produzione non chiusi tramite una tabella.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora la pagina di visualizzazione degli ordini di produzione non chiusi non sia disponibile.

### **3.4.17 UC17 – Creazione/Modifica Ordine Produzione**

* Descrizione: creazione/modifica ordine produzione.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato. In caso di modifica devono essere rispettate le seguenti condizioni:
  1. Il numero ordine non può essere modificato.
  2. Non si possono modificare ordini di produzione chiusi o in fase di produzione.
* Post condizioni: l’ordine di produzione viene inserito nel database con stato “Aperto”. I dati dell’ordine di produzione modificato vengono aggiornati nel database.
* Procedimento:
  1. L’utente inserisce/modifica i dati relativi al tipo macchina che si vuole creare/modificare.
  2. L’applicazione controlla che i dati inseriti dall’utente siano coerenti e che siano rispettate le condizioni esposte precedentemente.
  3. Vengono inseriti/modificati i dati del nuovo ordine di produzione/ordine di produzione da modificare nel db.
  4. Viene mostrato a schermo l’esito dell’operazione.
  5. Vengono mostrati a schermo i dati aggiornati letti dal db.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’operazione di creazione/modifica non sia andata a buon fine.

### **3.4.18 UC18 – Eliminazione Ordine di Produzione**

* Descrizione: eliminazione ordine di produzione.
* Attori coinvolti: utente.
* Precondizioni: l’utente deve essere autenticato. L’ordine di produzione che si vuole cancellare deve rispettare i seguenti requisiti:
  1. Non deve essere in stato chiuso.
  2. Non deve essere in fase di produzione.
* Post condizioni: l’ordine di produzione viene eliminato dal database.
* Procedimento:
  1. L’utente seleziona l’ordine di produzione che vuole eliminare.
  2. L’applicazione controlla che l’ordine di produzione da cancellare rispetti i requisiti sopra esposti.
  3. Viene eliminato l’ordine di produzione dal db.
  4. Viene mostrato a schermo l’esito dell’operazione.
* Eccezioni:
  1. Viene visualizzato un messaggio di errore qualora l’ordine di produzione non possa essere eliminato per i motivi sopra citati.



## **3.5 Programma di test**

Verranno principalmente testate le porzioni di codice non fornite da terze parti, assumendo che le funzionalità messe a disposizione siano già state testate dagli sviluppatori.

Il programma di test sarà composto di:

* Analisi statica del codice tramite il tool PSalm.
* Casi di test per ogni classe di tipo Model e Controller tramite PHPUnit.
* Verifica di Code Coverage tramite PHPUnit.
* Simulazione di utilizzo dell’applicazione da parte di un utente.
* Utilizzo di Postman per le chiamate http REST.

# **4. Iterazione 1**

Nell’iterazione 1 sono stati implementati i requisiti funzionali analizzati nell’iterazione 0.

Come da analisi, l’applicazione è stata sviluppata attraverso il framework Laravel che fornisce una struttura di progetto e una numerosa serie di librerie che aiutano lo sviluppatore nella costruzione di un software di tipo MVC.

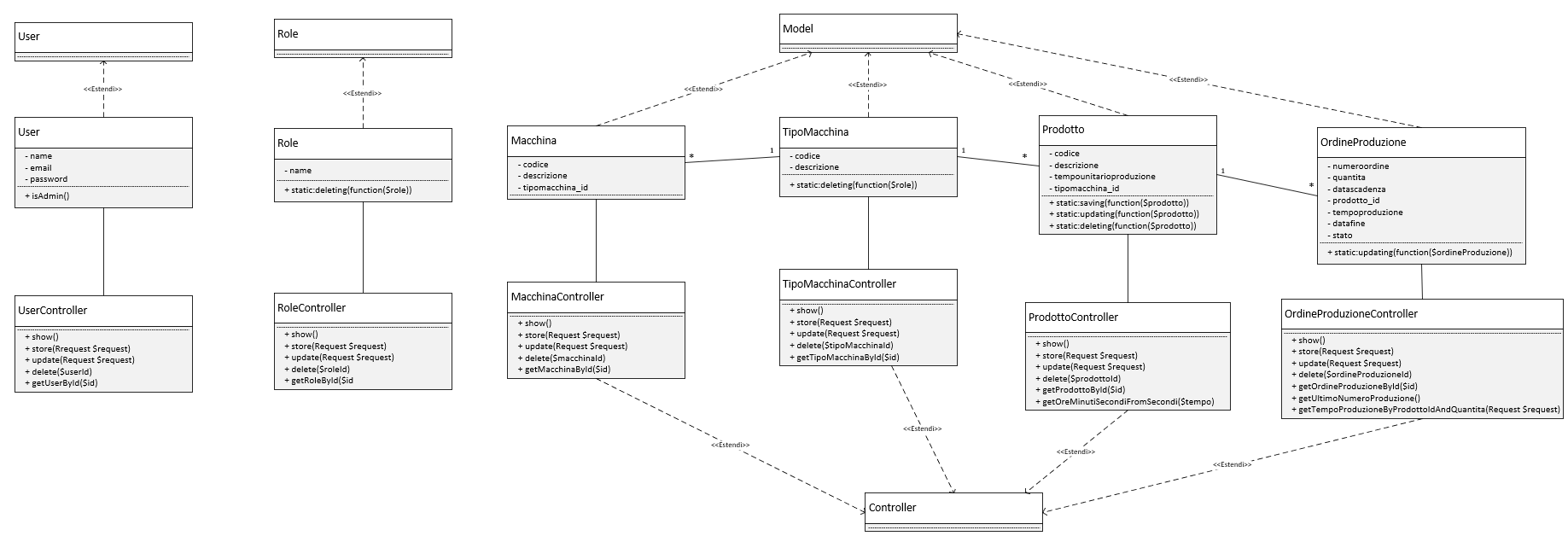
Seguendo la struttura del design pattern architetturale MVC (Model View Controller) sono stati implementate le seguenti classi tramite linguaggio PHP:

* **Model:**
  1. User
  2. Role
  3. Prodotto
  4. Tipo Macchina
  5. Macchina
  6. OrdineProduzione
* **Controller:**
  1. UserController
  2. RoleController
  3. ProdottoController
  4. TipoMacchinaController
  5. MacchinaController
  6. OrdineProduzioneController

Per quanto riguarda le **View** sono stati utilizzati i seguenti linguaggi:

* HTML
* CSS
* JavaScript

## **4.1 UML Class Diagram**



## **4.2 Testing**

### **4.2.1 Analisi statica**

Durante lo sviluppo dell’applicazione è stato utilizzato il tool **Psalm** per l’analisi statica del codice in modo da monitorare e prevenire gli errori di runtime.

Psalm viene installato all’interno dell’applicazione tramite Composer e consente di essere personalizzato tramite il file di configurazione “psalm.xml”. All’interno del file di configurazione può essere specificato il livello di controllo sull’errore (8 il meno restrittivo, 1 il più restrittivo) e l’elenco di directory sulle quali il tool eseguirà l’analisi.

Immagine che contiene testo

Descrizione generata automaticamente

Di seguito un esempio di analisi eseguita alla fine dello sviluppo dell’iterazione 1.

Immagine che contiene testo

Descrizione generata automaticamente

## **4.2.2 Analisi Dinamica**

In questa prima iterazione sono state sottoposte a unit test le classi scritte manualmente da noi. I test sulle classi fornite da terze parti non sono stati effettuati, in quanto già testate dai loro sviluppatori prima di essere rese disponibili.

Di seguito vengono riportate le classi con i metodi testati:

* UserController
  1. public function \_\_construct()
  2. public function show()
  3. public function store(Request $request)
  4. public function update(Request $request)
  5. public function getUserById($id)
* RoleController
  1. public function \_\_construct()
  2. public function show()
  3. public function store(Request $request)
  4. public function update(Request $request)
  5. public function delete($roleId)
  6. public function getRoleById($id)
* MacchinaController
  1. public function show()
  2. public function store(Request $request)
  3. public function update(Request $request)
  4. public function delete($macchinaId)
  5. public function getMacchinaById($id)
* TipoMacchinaController
  1. public function show()
  2. public function store(Request $request)
  3. public function update(Request $request)
  4. public function delete($tipoMacchinaId)
  5. public function getTipoMacchinaById($id)
* ProdottoController
  1. public function show()
  2. public function store(Request $request)
  3. public function update(Request $request)
  4. public function delete($prodottoId)
  5. public function getProdottoById($id)
  6. protected function getOreMinutiSecondiFromSecondi($tempo)
* OrdineProduzioneController
  1. public function show()
  2. public function store(Request $request)
  3. public function update(Request $request)
  4. public function delete($ordineProduzioneId)
  5. public function getOrdineProduzioneById($id)
  6. public function getUltimoNumeroProduzione()
  7. protected function getTempoProduzioneByProdottoIdAndQuantita(Request $request)
  8. protected function getOreMinutiSecondiFromSecondi($tempo)
* User
  1. public function isAdmin()
* Role
  1. public static function boot()
* TipoMacchina
  1. public static function boot()
* Prodotto
  1. public static function boot()
* OrdineProduzione
  1. public static function boot()

Per eseguire i test è stato utilizzato il tool PHPUnit. Per la visualizzazione dei risultati di code coverage invece ci siamo affidati al tool PCOV. Questo tool lavorando inisieme a PHPUnit produce un file html contenente i risultati dei test di unità eseguiti consultabili attraverso il browser.

Di seguito i risultati ottenuti dopo l’esecuzione dei test.



