

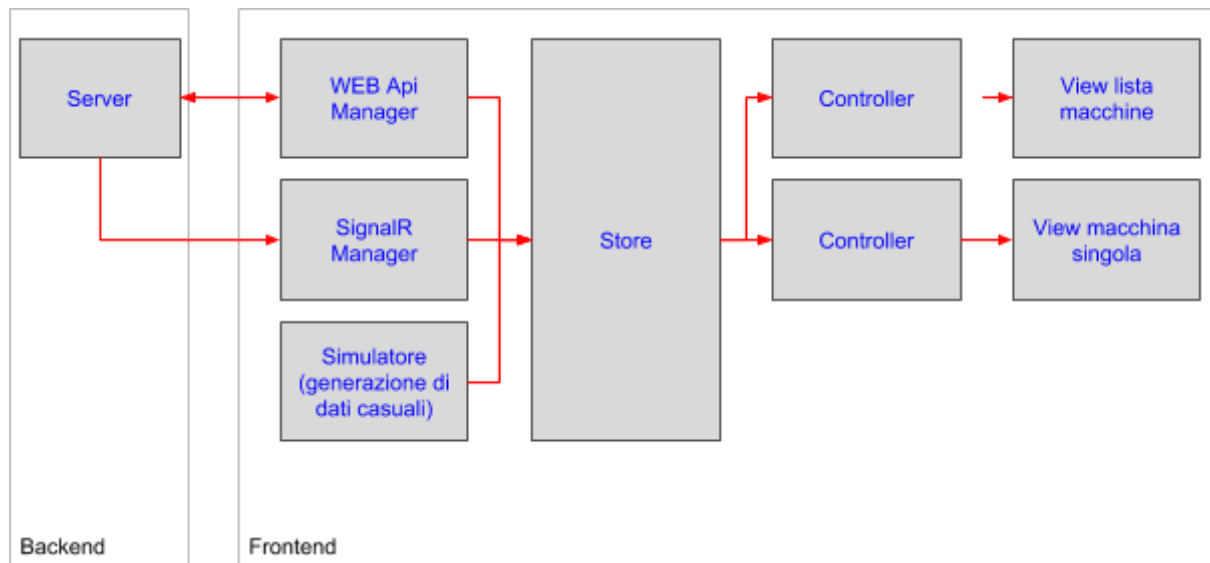
# Sviluppo prototipo MP/MON

## Prefazione

Vengono raggruppate in questo documento tutte le informazioni necessarie allo sviluppo del prototipo MP/MON.

## Analisi Tecnica

L'applicazione verrà sviluppata utilizzando i canoni standard MVC o MVVM. In particolare il seguente grafico rappresenta l'architettura che verrà sviluppata (moduli):



- Backend di proprietà Steamware a cui non vanno eseguite modifiche da Parte di Nicola Carminati.
- Il modo in cui si possono popolare le strutture dati nello store sono 3:
  - Tramite un simulatore (solo per test).
  - Tramite un API Manager in Polling che continua ad eseguire chiamate ogni N secondi. Modalità valida, ma non ottimizzata perchè si è vincolati a delle tempistiche gestite dal client e soprattutto perchè una chiamata API necessita di più elaborazione che sovraccaricano CPU Server ed allunga i tempi di aggiornamento.
  - Tramite un canale Signal-R, basato su Websocket, in cui si possono gestire le chiamate Push. Modalità ottimizzata: Canale sempre aperto tra Client e Server a basso consumo; le tempistiche son dettate dal Server che aggiorna i dati su sua decisione.
- Le informazioni salvate nello store vengono gestite in modo automatico dai controller e dalle View. Ad ogni cambiamento di un dato aggiornano solo le parti coinvolte.

# Specifiche

Attualmente l'applicazione esiste già ed è un applicazione MVC asp.net c#, NON interattiva (display distribuiti stato produzione), che effettua refresh timer-based.

Il nuovo prodotto sarà un'applicazione web html/css, che sia in grado di mostrare le informazioni delle macchine in modo dinamico senza il continuo refresh totale di tutta la pagina. In particolare le caratteristiche saranno:

- Il prodotto dovrà essere la base di partenza dell'applicazione finale, e quindi avrà delle caratteristiche di scalabilità e di stratificazione utili ad aggiungere o modificare delle parti di esso in futuro.
- Il linguaggio di sviluppo sarà Vue.js per la parte di logica e Less per la parte Css.
- Verrà utilizzato Bootstrap come libreria di componenti.
- Boilerplate in Node.js, Npm e Webpack
- Tutti i metodi che non eseguono override delle funzioni standard di Vue.js dovranno avere un commento antecedente con la spiegazione della sua funzione.
- Viene lasciato allo sviluppatore l'utilizzo degli altri commenti nel codice, considerando il buon senso di programmazione.
- Non è richiesta nessuna gestione di autenticazione utente.
- Non verrà sviluppato il modulo di reperimento dei dati tramite Signal-R (SOLO predisposizione) perchè non disponibile da Backend.
- Per sviluppare il modulo di reperimento delle informazioni tramite Api ci sarà la necessità di avere un server con i esposti metodi descritti nel documento "Sviluppo\_MP\_MON.pdf" (anche un server-simulatore).
- Verrà rispettato il comportamento dei componenti descritto nel documento "Sviluppo\_MP\_MON.pdf".
- La parte visibile all'utente finale saranno le 2 view descritte nel capitolo di Analisi Tecnica, le cui immagini sono allegate
- La grafica della nuova applicazione dovrà essere il più simile possibile a quella mostrata nelle immagini allegate, anche se alcune accortezze da parte del programmatore potranno essere implementate, se ritenute migliorative.

## Stima delle tempistiche

Attività	Ore/Uomo
Creazione del Boilerplate e analisi dei componenti necessari	1
Preparazione delle basi del progetto e architettura, compreso il modulo di routing delle pagine	3
Sviluppo modelli con metodi get e set nello Store	2
Sviluppo della grafica delle View tramite pagine statiche	5
Modifica delle View e dei controller per rendere dinamiche la pagine (aggancio allo store)	3
Sviluppo dei modulo Simulatore	2
Sviluppo del modulo WebApi	2
Testing prototipale con Steamware	2

Il totale delle ore è 20.