Documentazione

[Introduzione 2](#_Toc59130447)

[1.1 Informazioni sul progetto 2](#_Toc59130448)

[1.2 Abstract 2](#_Toc59130449)

[1.3 Scopo 2](#_Toc59130450)

[2 Analisi 3](#_Toc59130451)

[2.1 Analisi del dominio 3](#_Toc59130452)

[2.2 Analisi e specifica dei requisiti 3](#_Toc59130453)

[2.3 Use case 5](#_Toc59130454)

[2.4 Pianificazione 6](#_Toc59130455)

[2.4.1 Analisi 7](#_Toc59130456)

[2.4.2 Design 7](#_Toc59130457)

[2.4.3 Informazioni 7](#_Toc59130458)

[2.4.4 Setup spazio di lavoro 7](#_Toc59130459)

[2.4.5 Implementazione 8](#_Toc59130460)

[2.4.6 Testing 8](#_Toc59130461)

[2.5 Analisi dei mezzi 8](#_Toc59130462)

[2.5.1 Software 8](#_Toc59130463)

[2.5.2 Hardware 8](#_Toc59130464)

[3 Progettazione 9](#_Toc59130465)

[3.1 Design dell’architettura del sistema 9](#_Toc59130466)

[3.1.1 Schema di rete 9](#_Toc59130467)

[3.1.2 Diagramma di flusso 9](#_Toc59130468)

[3.2 Design dei dati e database 9](#_Toc59130469)

[3.3 Design delle interfacce 10](#_Toc59130470)

[3.4 Design procedurale 12](#_Toc59130471)

[4 Implementazione 13](#_Toc59130472)

[5 Test 16](#_Toc59130473)

[5.1 Protocollo di test 16](#_Toc59130474)

[5.2 Risultati test 17](#_Toc59130475)

[5.3 Mancanze/limitazioni conosciute 17](#_Toc59130476)

[6 Consuntivo 18](#_Toc59130477)

[7 Conclusioni 19](#_Toc59130478)

[7.1 Sviluppi futuri 19](#_Toc59130479)

[7.2 Considerazioni personali 19](#_Toc59130480)

[8 Bibliografia 19](#_Toc59130481)

[8.1 Bibliografia per articoli di riviste: 19](#_Toc59130482)

[8.2 Bibliografia per libri 19](#_Toc59130483)

[8.3 Sitografia 19](#_Toc59130484)

[9 Allegati 19](#_Toc59130485)

# Introduzione

## Informazioni sul progetto

Le persone coinvolte in questo progetto sono:

* Trentin Ismael – creatore
* Petrini Geo – docente responsabile

Il progetto è stato realizzato nella scuola SAM Trevano, sezione d’informatica per il modulo 306. La data di inizio è 03/09/2020 mentre quella di consegna è 17/12/2020.

## Abstract

Spesso se si lavora con degli equipaggiamenti di rete, o semplicemente con dei computer connessi, è bene sapere se tutto funziona nel modo esatto. Usare il terminale della propria macchina diventa noioso e spesso è scomodo, per questo Net Scanner offre tutte le funzionalità necessarie per verificare una buona rete. Net scanner permette di mandare un ping ad un range di indirizzi IP e per ognuno di essi verificare se determinate porte sono aperte, il risultato dello scan viene salvato in un file csv che rende possibile l’importazione su Excel. Tutto questo viene mostrato su un’interfaccia grafica semplice e moderna. Con questo programma verificare il funzionamento corretto di una rete diventa facile e veloce.

## Scopo

Lo scopo del progetto è di creare un applicativo di tipo net-port scanner. Questo software dovrà avere un’interfaccia grafica e dovrà possedere le seguenti funzionalità usabili dall’utente:

* Mostrare le macchine connesse alla rete specificata
* Mostrare le porte aperte di ogni macchina connessa alla rete specificata
* Generare un report finale parametrizzato

Ognuna di queste funzionalità dovrà accettare dei parametri:

* Un range o dei singoli indirizzi IP che specificano dove viene eseguito il controllo
* Un range o dei singoli valori che specificano quali porte controllare
* Quali informazioni mostrare nel report finale
* Eventuali metodi di ricerca differenti

L’applicativo potrà essere scritto in qualsiasi linguaggio.

# Analisi

## Analisi del dominio

Il prodotto verrà utilizzato in ambienti di rete dove ci sono delle macchine connesse ad essa. Altri applicativi come *Advanced IP Scanner* esistono già. Gli utenti che useranno il prodotto avranno il bisogno di sapere le lo stato di ogni indirizzo IP specificato da un range e le porte aperte di ogni indirizzo, in più avranno il bisogno di parametrizzare il file di output csv con le informazioni che ritengono più necessarie. Le conoscenze richieste per utilizzare al meglio il prodotto non sono troppe siccome basta sapere cosa è un indirizzo IP ed una porta.

## Analisi e specifica dei requisiti

Requisiti:

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Applicativo SW |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | GUI |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Da vedere se con caratteri (CLI) o con paint. |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Deve individuare tutte le macchine connesse alla rete |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il range di IP viene specificato tramite parametri. |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Deve ottenere tutte le porte aperte di ogni macchina connessa alla rete |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il range di porte viene specificato tramite parametri. |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Deve generare un report finale |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Le informazioni del report sono parametrizzate. |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Accetta un range di indirizzi IP come parametro |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Accetta un range di porte come parametro |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Accetta che informazioni mostrare nel report come parametro |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

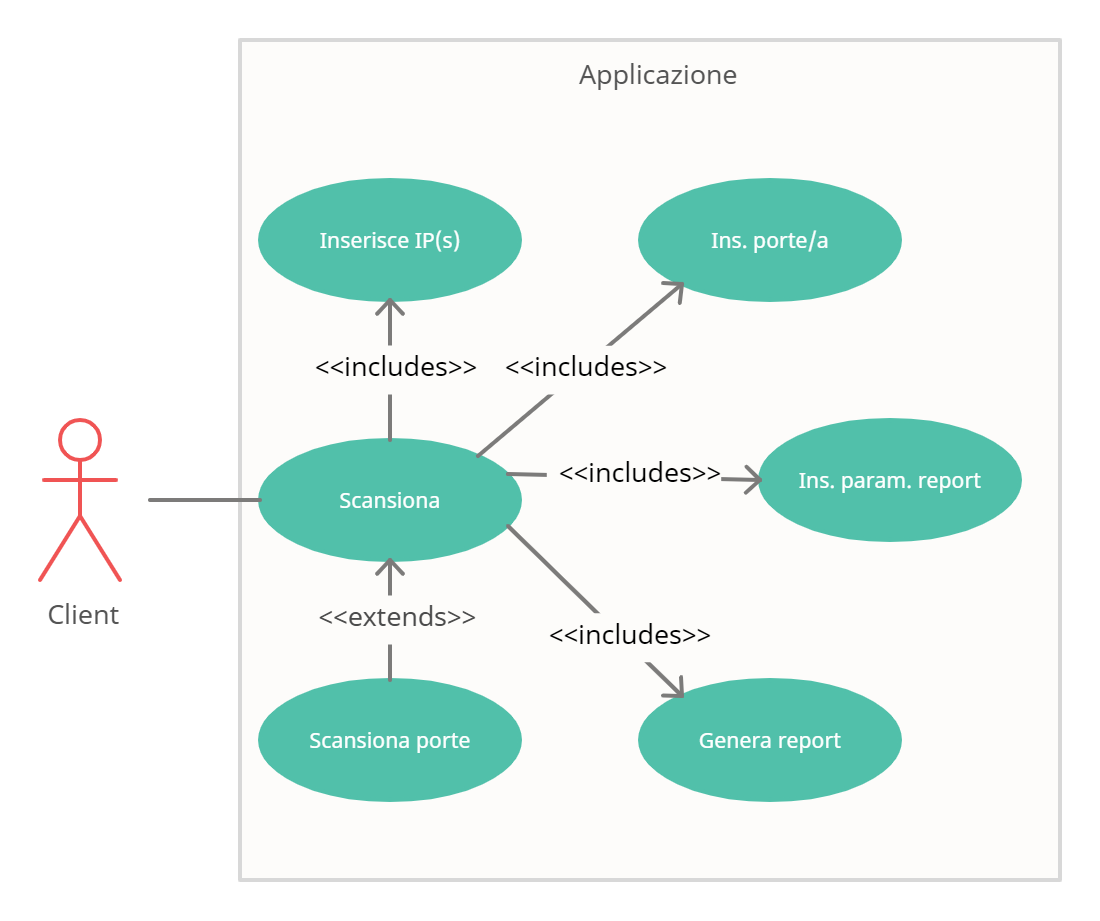
|  |  |
| --- | --- |
| **ID: REQ-009** | |
| **Nome** | Accetta metodi di ricerca differenti come parametro |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Eventuale. |

Specifica:

Il prodotto deve essere un software con un interfaccia grafica. Deve riuscire ad individuare tutte le macchine presenti in una rete. Le macchine vengono definite da un range o da singoli indirizzi IP. Per ogni macchina connessa deve ottenere tutte le porte aperte. Le porte anch’esse sono definite da un range o singolarmente. Alla fine, deve venir generato un file di report csv che può essere parametrizzato specificando quale informazioni mostrare.

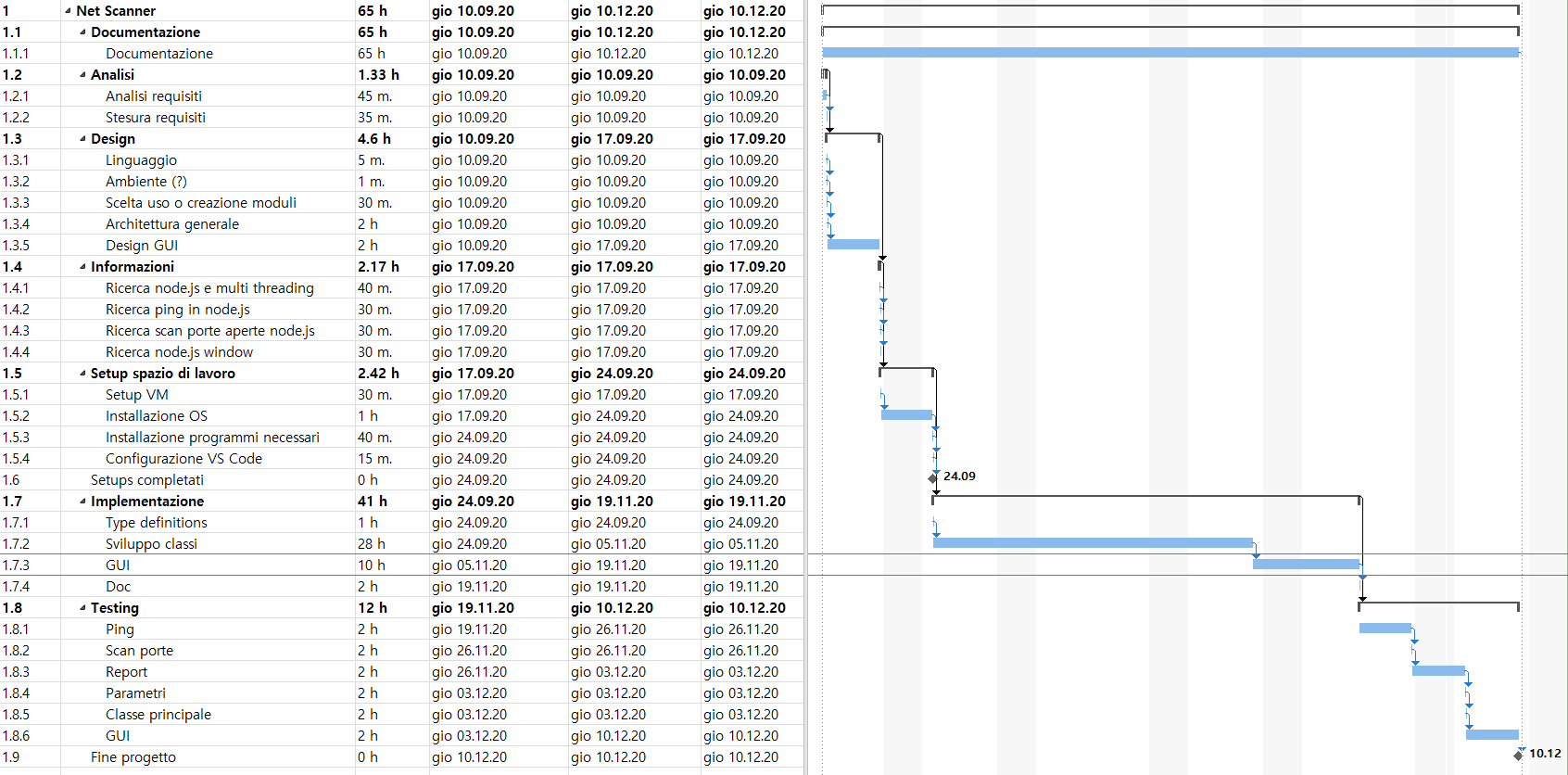
## Use case

Use case dell’applicazione:



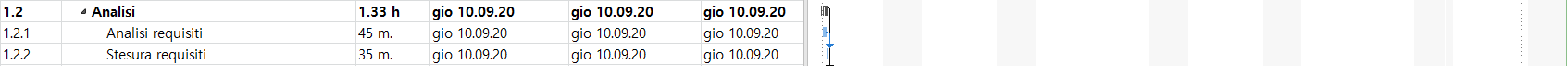
Per avviare una scansione l’utente deve inserire uno o degli indirizzi IP e un range o delle singole porte. In più può scegliere quali informazioni visualizzare nel report finale. Una volta finita la scansione viene generato un report finale.

## Pianificazione





### Analisi



Questa fase si occupa dell’analisi e della stesura dei requisiti. È molto importante siccome produrrà quello su cui poi ci si baserà per il design. In partica vengono analizzate le richieste del cliente e viene stesa una tabella dei requisiti.

### Design



La fase di design imposta le regole principali con cui si realizzerà il progetto. Di fatti in questa parte mi sono preso del tempo per decidere che linguaggio usare e su che ambiente. Siccome sapevo già di volerlo fare in TypeScript con NodeJS mi sono chiesto se avessi dovuto creare io da zero i moduli o se potevo usarne alcuni già fatti per realizzare il progetto. In più ho deciso l’architettura generale del programma assieme ad un design semplice dell’interfaccia grafica.

### Informazioni



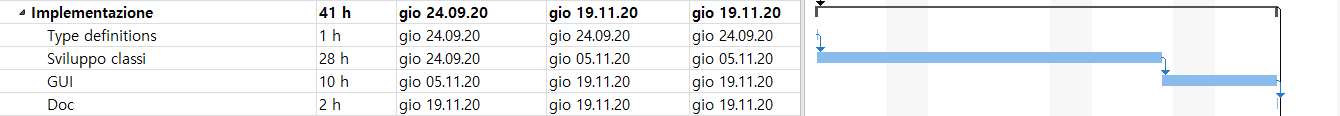
Siccome i requisiti riguardavano argomenti che non avevo mai trattato praticamente, ho deciso di dedicare del tempo per informarmi di più su cosa fare e su come farlo. Per garantire una buona performance dell’applicazione ho pensato di renderla multi thread. Peccato che si è rivelata una cosa decisamente contorta con JavaScript e quindi ho deciso di lasciarla perdere e di concentrarmi di più sui punti chiave.

### Setup spazio di lavoro



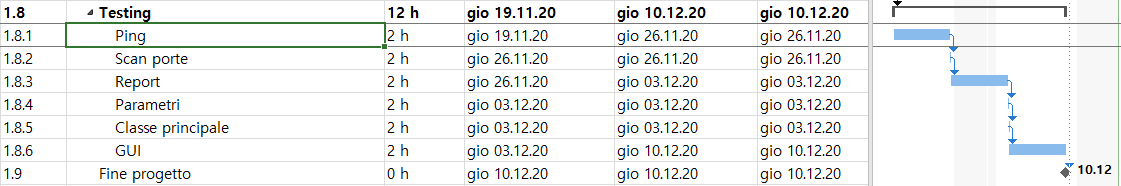
Siccome ho bisogno di NodeJS e altri vari software non presenti nel computer di scuola ho deciso di lavorare su una virtual machine. Come software di virtualizzazione ho usato VMware 14, programma che si è rivelato **estremamente** lento e causava mille problemi e che mi ha fatto perdere ben 2 intere lezioni. Nella foto si può vedere anche una milestone che specifica che il setup del necessario è completato e ora si può iniziare a lavorare.

### Implementazione



Questa è la vera fase di lavoro. Siccome verrà usato TypeScript bisognerà fare delle typedef per alcuni package che verranno scaricati da npm. Dopo di quello vanno realizzate le classi che faranno da scheletro del programma. Infine va sviluppata l’interfaccia grafica e va collegata alle classi. Ci terrei a fare qualche parola su una piccola parte di codice di cui vado fiero che è uscita durante la realizzazione di questo progetto che si trova nel capitolo [Implementazione](#_Implementazione). Per l’attività Doc non si intende la documentazione del progetto ma quella del codice.

### Testing



## Analisi dei mezzi

Il mezzo fisico utilizzabile è il computer fornito dalla scuola. I mezzi software invece sono:

**VMware** – software per la virtualizzazione. Siccome ho bisogno di installare un ambiente NodeJS ho deciso di partire da zero avendo una macchina virtuale pulita dove poi andrò ad installare tutto l’occorrente.  
**NodeJS** – piattaforma per JavaScript server side. Dato che ho ritenuto molto opportuno l’uso di JavaScript per la realizzazione di questo progetto e che mi reputo abbastanza competente nell’usarlo ho deciso di sfruttare Electron, che necessita di NodeJS, per creare un app desktop con esso.

**Electron** – permette di creare app desktop in JavaScript.

### Software

Software utilizzati per lo sviluppo:

* Visual Studio Code 1.52.0
* NodeJS 12.16.1
* Microsoft Project 16.0.13426.20308
* Microsoft Word 16.0.13426.20308
* Git 2.28.0
* Electron 9.3.0
* VMware 14

Librerie utilizzate:

* React 16.13.1

### Hardware

L’applicativo potrà girare su qualsiasi piattaforma (Mac, Windows e Linux) con architettura x64. La funzionalità è assicurata se viene usato un sistema Windows. Il minimo richiesto sono 2GB di memoria RAM.

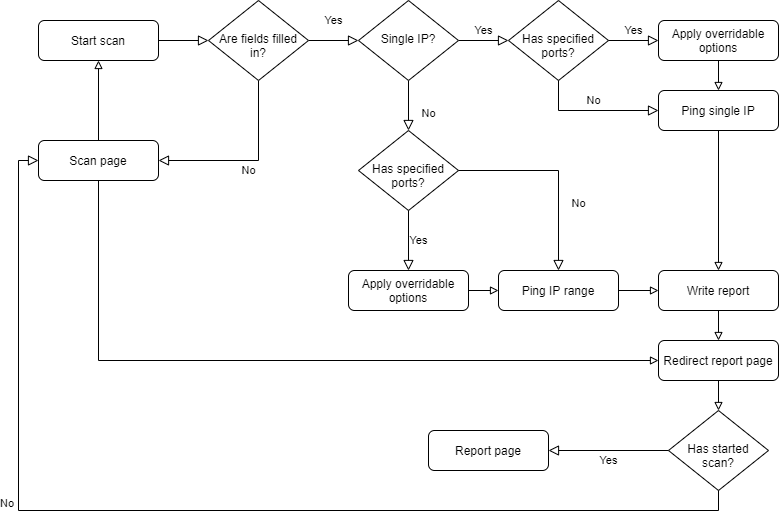
# Progettazione

## Design dell’architettura del sistema

### Schema di rete

La macchina deve essere semplicemente connessa ad una qualsiasi rete dove ci possono essere altre macchine che potranno rispondere ai ping.

### Diagramma di flusso



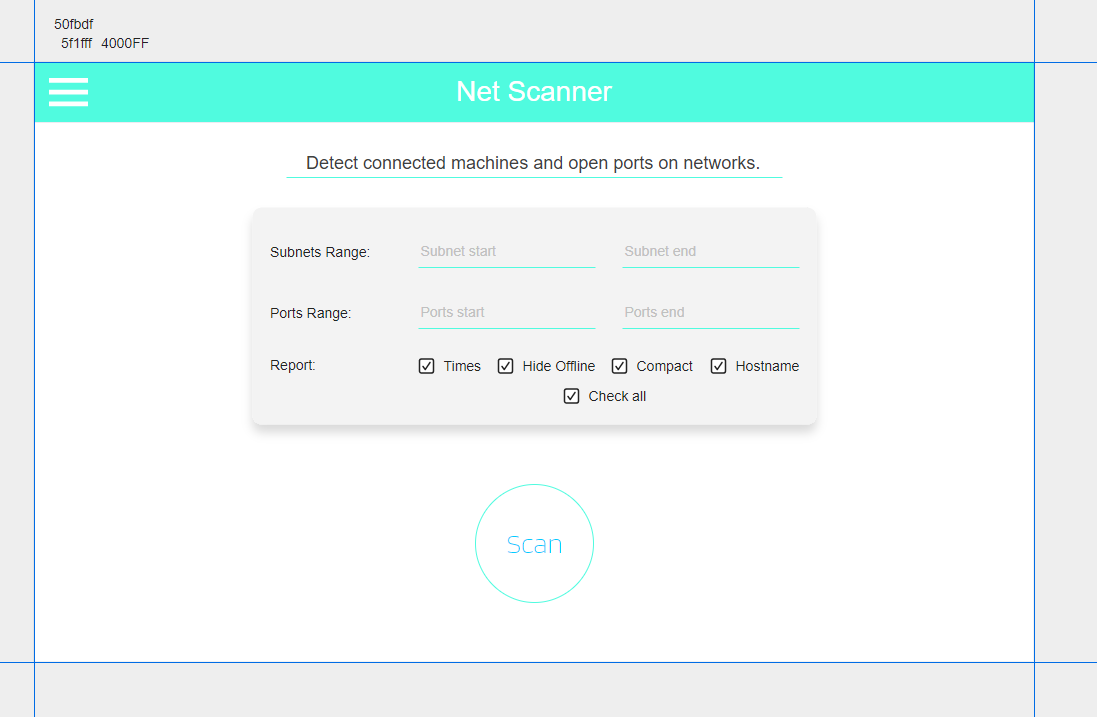
L’applicativo si mostra con la ‘Scan page’ come pagina iniziale. Se l’utente vuole scansionare un ip dovrà mettere un ip nel campo degli ip, altrimenti gli verrà dato un errore e verrà rimandato alla pagina principale. Se invece vuole scansionare un range di indirizzi dovrà riempire i 2 campi di inizio e fine. Lo scan avrà successo sia se vengono immesse delle porte che non. Se delle porte sono state specificate allora le userà nello scan. Quando l’utente prova ad accedere alla pagina di report senza aver fatto partire uno scan verrà reindirizzato sulla pagina principale.

## Design dei dati e database

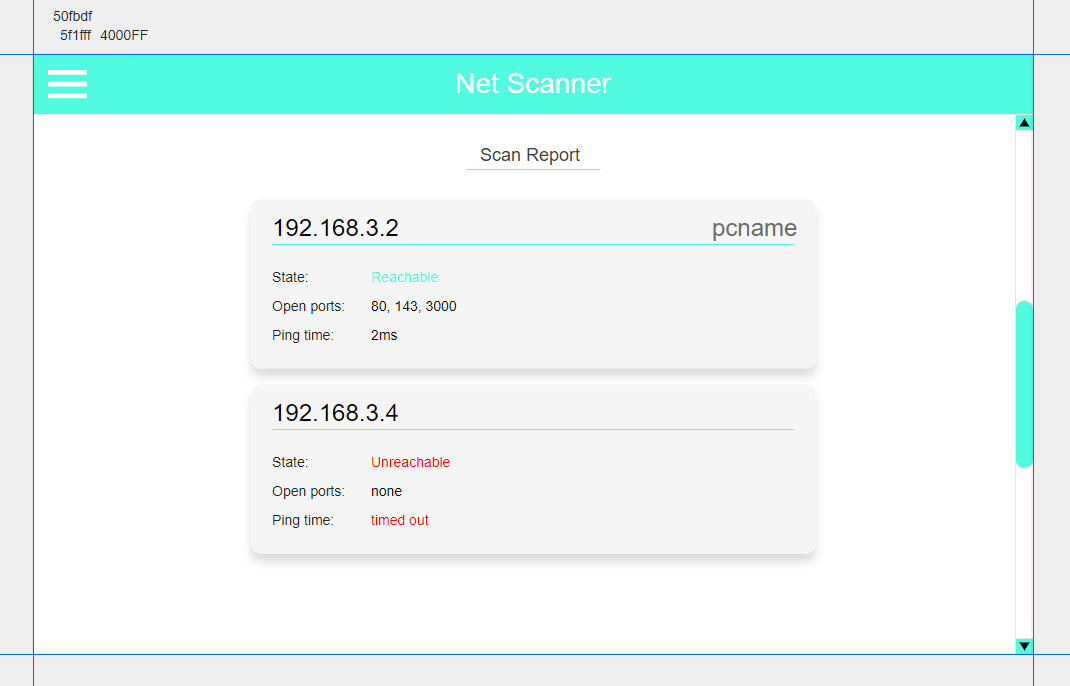
Per questo applicativo non è stato usato alcun database.

## Design delle interfacce

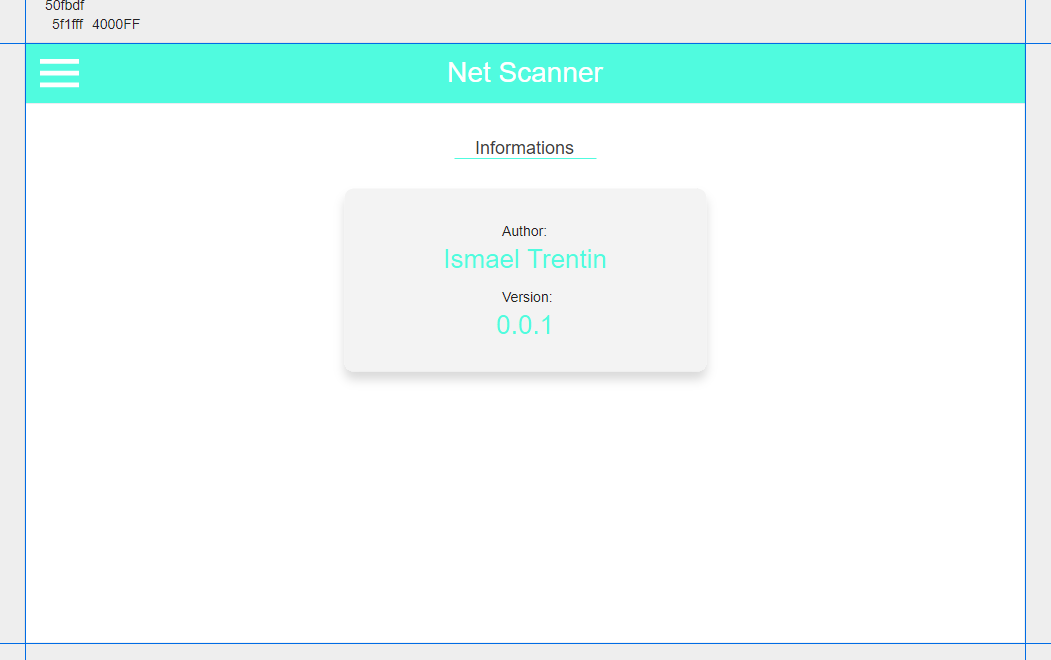
Pagina principale:



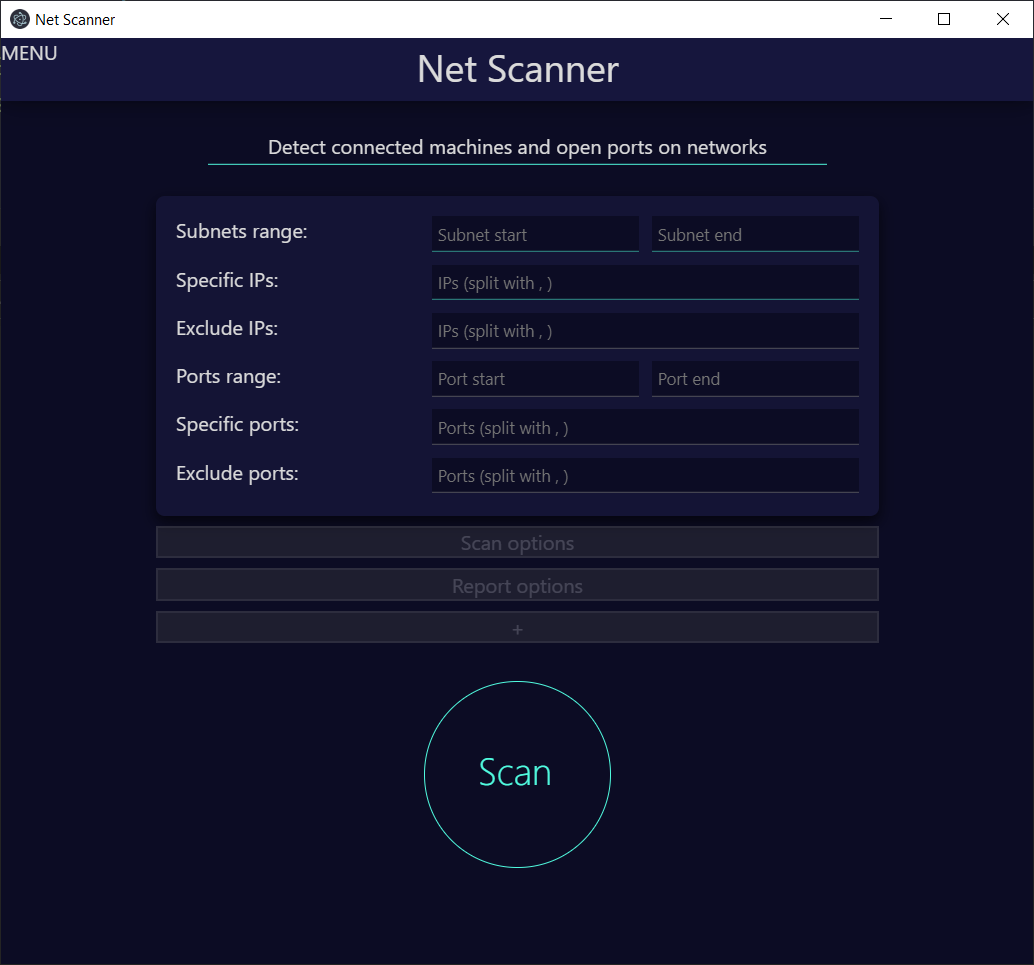
Pagina report:



Pagina delle informazioni sul software:

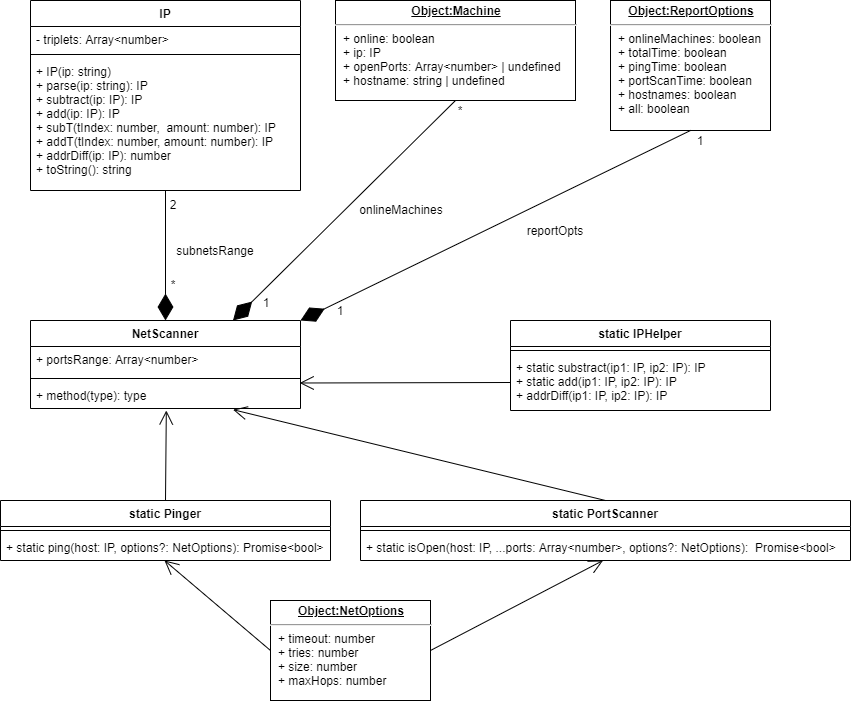


Per motivi di design e leggibilità le interfacce sono state cambiate in un tema più scuro rispetto a quella della progettazione. Esempio della pagina di scan:



## Design procedurale

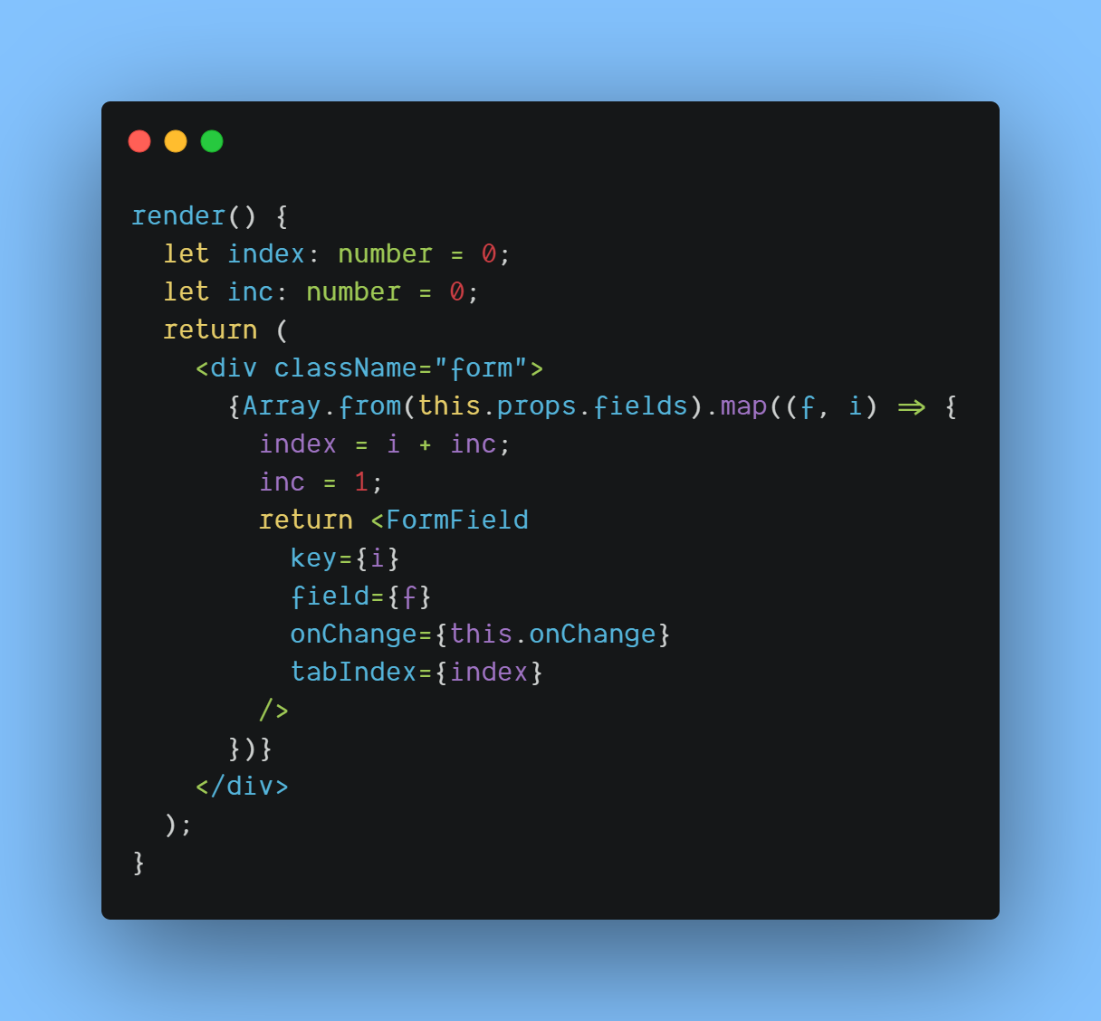
Questo è lo schema di partenza creato durante la fase di design della struttura del sistema.



Per comodità e/o complicità alcune classi e interfacce sono state alterate. Per esempio, IPHelper è andato a far parte della classe IP siccome era inutile avere un'altra classe in giro quando poteva fare benissimo il lavoro la classe IP. Invece, per lo scan, è stato tutto separato per via del funzionamento di React. Quando si trattano dei dati che arrivano in modo asincrono React ha uno specifico modo di lavorare per mantenere corretto il rendering di ogni componente.

# Implementazione

Per gestire l’input dell’utente ho deciso di adottare un sistema di tipo form e quindi ho sfruttato quest’occasione per renderlo il più modulare e riusabile possibile.



Questo è il render per la classe Form. Una classe Form accetta come proprietà un array di Fields (campi) e una callback per quando un campo viene modificato. I campi possono essere di 2 tipi, questi tipi sono definiti in un Enum.

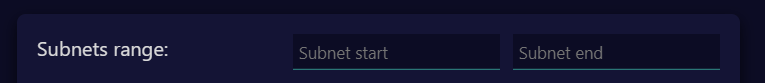


Questi 2 tipi di campo cambiano il modo in cui vengono renderizzati. Eccome come vengono mostrati:

Tipo SIMPLE:



Tipo RANGE:



Oltre al tipo, un campo possiede molte altre diverse proprietà che lo distinguono:

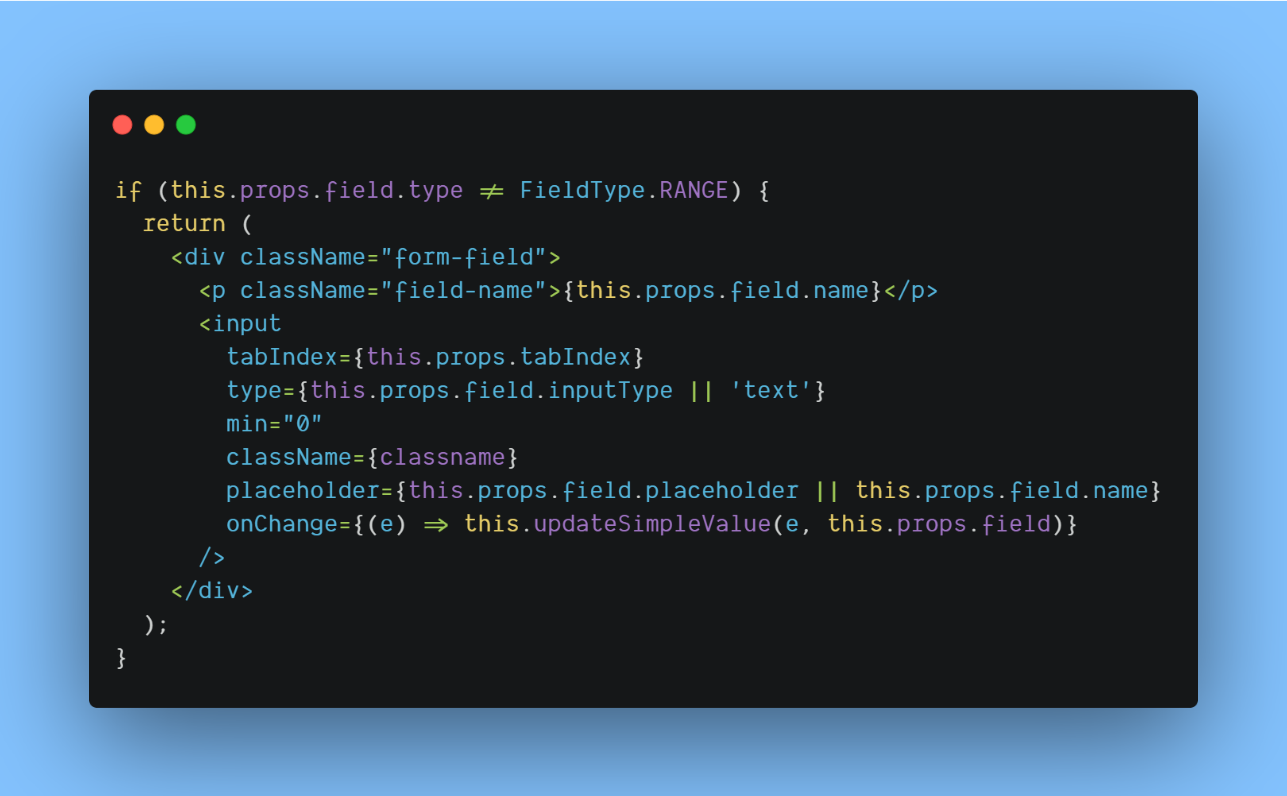


Ogni campo ha un nome che sarà poi quello stampato come etichetta del campo. I campi possono essere opzionali o obbligatori, questo ne influenza il loro style disegnando il bordo colorato se obbligatori o grigio se opzionali. Il tipo specifica se è un range o un valore semplice mentre inputType il tipo di dato che andrà inserito nel campo. Il placeholder è la stringa di testo che compare quando il campo è vuoto. Span prevede 3 possibilità:

* none: il campo ha la larghezza minima ed è allineato a destra
* span: il campo ha la larghezza di 2 campi semplici ed è allineato a destra
* big: il campo prenderà la larghezza massima nel form ed andrà a posizionarsi sotto il suo nome.

La proprietà value è semplicemente il tipo di valore che avrà, questo dipende se il campo è di tipo RANGE o SIMPLE.

Ecco come viene renderizzato un field di tipo SIMPLE:



Questo sistema è ancora da migliorare e ci sono ancora dei valori hardcoded, però è un’ottima base da cui continuare.

Per gestire le varie pagine, viene usato React-Router che permette all’applicazione di comportarsi quasi come un sito web ed avere un percorso per ogni pagina.

Il ping e lo scan delle porte sono gestiti da due package scaricati da npm chiamati:

* ping
* is-port-reachable

Entrambi ritornano i dati in modo asincrono, per questo è stata usata l’API Promise di JS.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-06 | **Nome:** | Definire un range di IP |
| **Descrizione:** | Verifica dell’ottenimento del range di indirizzi IP. | | |
| **Prerequisiti:** | Essere nella scan page | | |
| **Procedura:** | 1. Inserire l’IP di inizio del range nel campo start del range 2. Inserire l’IP di fine del range nel campo end del range 3. Cliccare sul bottone ‘Scan’ | | |
| **Risultati attesi:** | Nella pagina di report compaiono tutti gli indirizzi compresi nel range. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-07 | **Nome:** | Definire un range di porte |
| **Descrizione:** | Verifica dell’ottenimento del range di porte. | | |
| **Prerequisiti:** | Essere nella scan page.  Aver inserite almeno un IP o un range di IP.  Conoscere almeno una porta aperta della macchina di quell’indirizzo per verificarne il funzionamento. | | |
| **Procedura:** | 1. Inserire la porta di inizio nel campo start del range delle porte 2. Inserire la porte di fine nel campo start del range delle porte 3. Cliccare sul bottone ‘Scan’ | | |
| **Risultati attesi:** | La macchina di cui conosciamo le porte aperte deve avere le porte specificate nella lista di porte aperte. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-08 | **Nome:** | Definire le opzioni del report |
| **Descrizione:** | Verifica dell’ottenimento delle opzioni per il report | | |
| **Prerequisiti:** | Essere nella scan page.  Aver inserite almeno un IP o un range di IP. | | |
| **Procedura:** | 1. Scegliere le impostazioni del report | | |
| **Risultati attesi:** | Nel file compaiono solo i campi selezionati nelle opzioni. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-05 | **Nome:** | Generazione file di report |
| **Descrizione:** | Verifica che il file di report viene creato. | | |
| **Prerequisiti:** | Aver compilato correttamente il form della scan page. | | |
| **Procedura:** | 1. Cliccare sul bottone ‘Scan’ | | |
| **Risultati attesi:** | Nella directory del programma sotto la cartella ‘scan\_results’ deve comparire un file .csv con nome scanreport\_dd-mm-yyyy\_hh\_mm\_ss.csv | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-03 | **Nome:** | Scoperta macchine nella rete |
| **Descrizione:** | Verifica se riesce a trovare tutte le macchine nella rete | | |
| **Prerequisiti:** | Nessuno | | |
| **Procedura:** | 1. Immettere un range o un singolo IP nell’apposito campo 2. Cliccare sul bottone ‘Scan’ | | |
| **Risultati attesi:** | Nel report compaiono tutti gli indirizzi con il loro stato e altre informazioni | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-04 | **Nome:** | Scoperta porte aperte di una macchina |
| **Descrizione:** | Verifica se riesce a trovare se le porte specificate sono aperte nelle macchine online. | | |
| **Prerequisiti:** | Aver scelto un range o un singolo IP | | |
| **Procedura:** | 1. Specificare le porte tramite range o espressione 2. Cliccare sul bottone ‘Scan’ | | |
| **Risultati attesi:** | Per ogni macchina online devono comparire le sue porte aperte | | |

## Risultati test

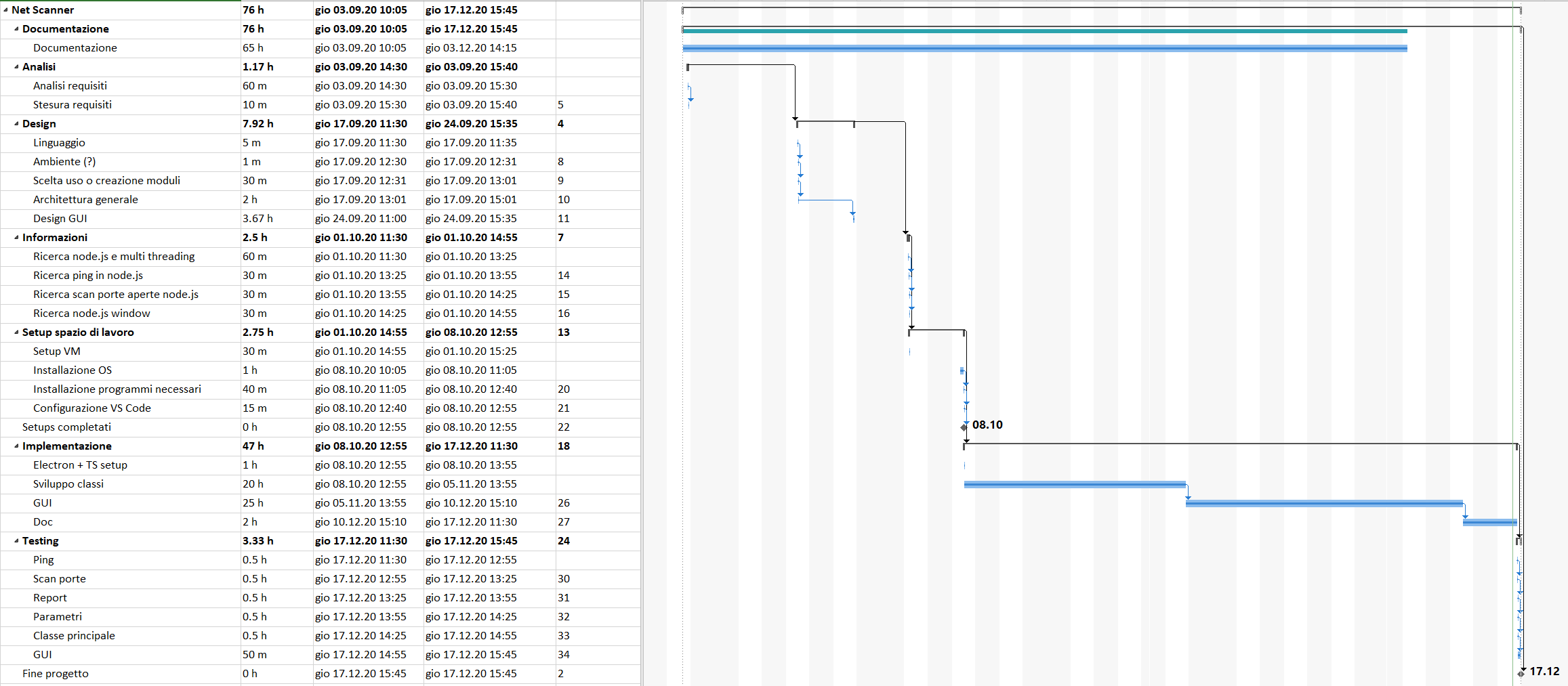
|  |  |
| --- | --- |
| **TC CODE** | **OUTCOME** |
| TC-001 | PASSED |
| TC-002 | PASSED |
| TC-003 | FAILED |
| TC-004 | PASSED |
| TC-005 | PASSED |
| TC-006 | FAILED\* |

\* A volte le porte vengono trovate, ma la maggior parte dei test non sono stati superati.

## Mancanze/limitazioni conosciute

I parametri della scansione non vengono usati, la GUI e la classe di scan ne hanno la funzionalità per usarle ma purtroppo per mancanza di tempo non sono riuscito a collegarli. Questo vale anche per i parametri del report. Le porte non vengono scannate penso per un problema con l’asincronismo e React. Ho dovuto sacrificare questa funzionalità per far andare le altre e non rischiare di sbilanciare tutto. Il report può essere esportato solo come csv.

# Consuntivo



Ciò che mi ha fatto perdere moltissime ore è stato VMware 14. Ogni lezione ci mettevo almeno 15 minuti per avviare la macchina virtuale se non di più. Ho avuto un sacco di crash e dovuto persino rifare un'altra macchina virtuale con virtual box perché quella su VMware non si connetteva più a internet.

# Conclusioni

Sono molto felice del risultato anche se non è completamente funzionante. Questo progetto mi ha aiutato un sacco a migliorare con React e in generale nell’ambiente NodeJS. Sono fiero di me stesso perché’ sono riuscito ad imparare le basi di React. Questa libreria mi ha da sempre interessato ma non ho mai messo del tempo nell’impararla e provarla, con questo progetto finalmente ho potuto scoprire in che cosa consiste e come si usa. Ho solo scoperto la superficie e sinceramente penso proprio che continuerò ad apprendere di più sull’argomento, così da poterlo usare per i miei progetti. Infine, ho trovato che CSS è un molto più utile di quel che pensassi e mi ha fatto appassionare ancora di più al mondo del web developing.

## Sviluppi futuri

Si potrebbe generare un ulteriore report ma in formato JSON così da poterlo usare in altre applicazioni. Un'altra miglioria molto importante sarebbe di rendere il programma multi-thread, soltanto che sembra molto complicato e contorto in JavaScript.

## Considerazioni personali

Ho imparato le basi di React e anche come gestire dei piccoli ma pur sempre corposi progetti. Di sicuro mi servirà per quando deciderò di svolgere dei progetti personali.

# Bibliografia

## Bibliografia per articoli di riviste:

Non sono stati visitati alcuni articoli o riviste.

## Bibliografia per libri

Non sono stati usati alcuni libri.

## Sitografia

<http://stackoverflow.com>, visitato molteplici volte dal 09-2020 al 12-2020

<https://reactjs.org/docs/react-api.html>, React Top-Level API, visitato molteplici volte dal 11-2020 al 12-2020

https://www.electronjs.org/docs, Documentazione di Electron, visitato molteplici volte dal 09-2020 al 10-2020

https://www.npmjs.com/package/ping, NODE-PING, visitato il 17.09.2020

https://www.npmjs.com/package/is-port-reachable, is-port-reachable, visitato il 17.09.2020

https://carbon.now.sh, carbon, visitato il 17.12.2020

https://www.gloomaps.com, visitato il 17.12.2020

https://app.diagrams.net/?libs=general;flowchart, visitato il 17.12.2020

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o Qdc
* Prodotto
* …