

Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite le API

Titolo del progetto: Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite API
Alunno: Pierpaolo Casati
Classe: SAM I4AA
Anno scolastico: 2020/2021
Docente responsabile: Fabrizio Valsangiacomo

| | | |
|--------|--|----|
| 1 | Indice delle figure | 3 |
| 2 | Introduzione | 4 |
| 2.1 | Informazioni sul progetto | 4 |
| 2.2 | Abstract | 4 |
| 2.3 | Scopo | 4 |
| 3 | Analisi | 5 |
| 3.1 | Analisi del dominio | 5 |
| 3.2 | Analisi API Nagios | 6 |
| 3.3 | Analisi e specifica dei requisiti | 9 |
| 3.4 | Use case | 12 |
| 3.5 | Pianificazione | 12 |
| 3.6 | Analisi dei mezzi | 13 |
| 3.6.1 | Software | 13 |
| 3.6.2 | Librerie | 14 |
| 3.6.3 | Hardware | 15 |
| 4 | Progettazione | 15 |
| 4.1 | Design dell'architettura del sistema | 15 |
| 4.2 | Design dei dati e database | 17 |
| 4.2.1 | Schema logico | 17 |
| 4.2.2 | Descrizione tabella | 18 |
| 4.3 | Design delle interfacce | 21 |
| 4.4 | Design procedurale | 23 |
| 4.4.1 | UML Models | 23 |
| 4.4.2 | UML Controllers | 25 |
| 5 | Implementazione | 27 |
| 5.1 | Applicativo web | 27 |
| 5.1.1 | Struttura | 27 |
| 5.1.2 | Model View Controller | 28 |
| 5.1.3 | Configurazione del file config | 28 |
| 5.1.4 | Connessione al database | 29 |
| 5.1.5 | Sicurezza | 31 |
| 5.1.6 | Sistema di email | 34 |
| 5.1.7 | Popup | 36 |
| 5.1.8 | Sistema di visualizzazione sul monitor esterno | 38 |
| 5.1.9 | API Nagios | 38 |
| 5.1.10 | Raspberry | 39 |
| 5.1.11 | Interfacce delle pagine | 40 |
| 6 | Test | 44 |
| 6.1 | Protocollo di test | 44 |
| 6.2 | Risultati test | 50 |
| 6.3 | Mancanze/limitazioni conosciute | 54 |
| 7 | Consuntivo | 55 |
| 8 | Conclusioni | 56 |
| 8.1 | Sviluppi futuri | 56 |
| 8.2 | Considerazioni personali | 56 |
| 9 | Glossario | 57 |
| 10 | Bibliografia | 58 |
| 10.1 | Sitografia | 58 |
| 11 | Allegati | 58 |

1 Indice delle figure

| | |
|---|----|
| Figura 1: Struttura Nagios..... | 6 |
| Figura 2: Nagios query | 8 |
| Figura 3: Use case..... | 12 |
| Figura 4: GANTT Preventivo | 12 |
| Figura 5: Architettura del sistema | 16 |
| Figura 6: Sitemap | 16 |
| Figura 7: Schema ER | 17 |
| Figura 8: Design pagina di login | 21 |
| Figura 9: Design pagina di gestione degli utenti | 21 |
| Figura 10: Design pagina per creare un nuovo utente | 22 |
| Figura 11: Design pagina di amministrazione | 22 |
| Figura 12: Design pagina di visualizzazione degli allarmi | 23 |
| Figura 13: Model database | 23 |
| Figura 14: Model allarme | 24 |
| Figura 15: Model utente | 24 |
| Figura 16: Model mail | 25 |
| Figura 17: Model password | 25 |
| Figura 18: Controller login | 25 |
| Figura 19: controller utente..... | 26 |
| Figura 20: Controller allarme | 26 |
| Figura 21: Controller monitor | 27 |
| Figura 22: Struttura applicativo web | 27 |
| Figura 23: MVC..... | 28 |
| Figura 24: File di configurazione | 29 |
| Figura 25: Costanti per accedere al database | 29 |
| Figura 26: Metodo per connettersi al database | 30 |
| Figura 27: Esempio di prepared statement | 31 |
| Figura 28: Metodo per controllare i valori degli input | 32 |
| Figura 29: Pattern nome dell'utente | 32 |
| Figura 30: Controllo dell'email | 32 |
| Figura 31: Metodo per codificare le password..... | 33 |
| Figura 32: Metodo per verificare l'hash con la password in chiaro | 33 |
| Figura 33: Metodo per generare il token | 34 |
| Figura 34: Metodo send che permette di inviare email | 35 |
| Figura 35: Metodo send che permette di inviare email | 36 |
| Figura 36: Codice JavaScript popup | 37 |
| Figura 37: Codice AJAX di esempio..... | 38 |
| Figura 38: Costanti per potere accedere all'API Nagios | 38 |
| Figura 39: Metodo che permette di ottenere i dati da Nagios | 39 |
| Figura 40: Pagina di login | 40 |
| Figura 41: Pagina gestione utenti | 41 |
| Figura 42: Pagina di creazione utente..... | 42 |
| Figura 43: Pagina gestione amministrazione | 42 |
| Figura 44: Pagina di visualizzazione | 43 |
| Figura 45: GANTT consuntivo | 55 |

2 Introduzione

2.1 Informazioni sul progetto

Titolo: Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite le API

Candidato: Pierpaolo Casati, pierpaolo.casati@samtrevano.ch

Formatore: Fabrizio Valsangiacomo, fabrizio.valsangiacomo@edu.ti.ch

Perito: Claudio Bortoluzzi, claudio.bortoluzzi@rsi.ch

Azienda: Scuola Arti e Mestieri Trevano

Classe: I4AA

Data di inizio: 03.05.2021

Data di consegna: 27.05.2021

Durata: 80 ore

2.2 Abstract

In this project, I was asked to create a website that allows you to view the alarms and information in the Nagios application on an external monitor and via the Nagios API. For the moment the CPT does not have such a web application, so this project implementation would make it easier to manage alarms. The application must be run on all popular browsers and the product must have an administration page that allows you to manage the various alarm fields to be made visible. In the application there are two types of users, the first user is the administrator (at least one) and has the permission to create, delete, modify fields or make changes to the database directly from the web page without using MySQL. The administrator also has the ability to manage permissions or delete users. The second user is a limited user who can only view the alarm display page. An MVC (Model View Controller) is used to build the web application which allows me to better separate the code and functions. For the CSS implementation I will use a template called bootstrap.

2.3 Scopo

Lo scopo del progetto "Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite le API" consiste di creare un applicativo web con PHP che permette di visualizzare gli allarmi e le informazioni su un monitor esterno tramite le API di Nagios. Al momento nella sede della CPT non esiste un applicativo web che permette di visualizzare le informazioni e gli allarmi, in effetti è un progetto sperimentale che potrà essere modificato per degli sviluppi. L'applicativo Nagios è già stato implementato dagli sistemisti della CPT e permette di monitorare tutti gli equipaggiamenti produttivi nella rete informatica del CPT di Trevano. Nell'applicativo web esistono due tipi di utenti che sono gli amministratori e limitati. Nell'applicativo web ci deve essere una pagina di gestione per gli utenti e una pagina di amministrazione che possono solo accedere gli amministratori. Gli utenti limitati possono solo visualizzare gli allarmi e le informazioni. Gli strumenti principali che vengono utilizzati nel progetto sono il monitor esterno e un Raspberry per l'interfacciamento.

3 Analisi

3.1 Analisi del dominio

Attualmente nella sede della CPT non esiste un applicativo web che permette di visualizzare gli allarmi e le informazioni dei vari equipaggiamenti produttivi nella rete del CPT di Trevano. L'applicativo web permette di aiutare i sistemisti a visualizzare meglio le informazioni dei vari equipaggiamenti produttivi della rete informatica della CPT di Trevano e inoltre permette anche di avvisare con degli allarmi il sistemista in caso di problemi. Essendo un progetto sperimentale, nella documentazione verranno descritti in modo dettagliato le varie opzioni e le possibili varianti. In effetti la documentazione deve essere utile per potere mettere in produzione il prodotto in modo ottimale.

Nell'applicativo web ci sono due tipi di utenti che sono gli amministratori e gli utenti limitati. L'amministratore ha il permesso di creare, modificare e eliminare degli utenti direttamente dalla pagina di gestione degli utenti. L'amministratore può anche accedere alla pagina di amministrazione nella quale può scegliere tramite dei "check" i campi che vuole rendere visibile, come per esempio: Host, Status, Last Check, Duration e Status Information. Il campo Status dovrà essere identificato in modo inequivocabile con i colori in base all'importanza dell'allarme. L'amministratore può fare diventare in qualsiasi momento un utente amministratore con solo diritti limitati. L'utente limitato non può accedere alla pagina di gestione degli utenti, ma ha la possibilità di vedere quali informazioni verranno viste sul monitor esterno. Quando viene creato un nuovo utente ci deve essere un sistema di email che invia a quest'ultimo le credenziali per accedere all'applicativo web. Al primo login l'utente ha la possibilità di modificare la password e i suoi dati se è necessario. Nell'applicativo c'è anche un sistema che permette di recuperare la password dimenticata. Sul monitor esterno dovranno essere visualizzate le varie informazioni presenti nell'applicativo Nagios e per accedere. Lo sfondo di questa pagina deve essere bianco e deve essere suddiviso in due parti. Nella parte sinistra devono essere visibili i campi degli allarmi, invece sulla destra deve essere presente una mappa della rete del CPT di Trevano realizzata con Nagios MAP. Nella pagina di amministrazione deve anche essere un "check" per visualizzare la mappa di rete. Se l'utente non seleziona questo "check", le informazioni degli allarmi occupano tutto lo spazio dello schermo. Devo anche creare un versione desktop che permette di accedere alla pagina del monitor esterno. Per interfacciare lo schema con il collegamento di rete, viene utilizzato un Raspberry che deve essere raggiungibile via SSH. Per l'installazione e la configurazione del Raspberry dovrò preparare un manuale d'uso.

L'applicativo web dovrà essere compatibile su tutti i browser più utilizzati, come per esempio, Firefox, Google Chrome, Safari, ecc. L'applicazione verrà programmato con i linguaggi WEB (HTML, CSS, PHP) e quindi viene implementato un web server. In effetti per creare il web server viene installato sul Raspberry il software Apache e l'applicativo è accessibile solamente nella rete interna della CPT. In fase produttiva utilizzerò il software XAMPP che è una multiplatforma software libera costituita da Apache HTTP Server, il database MySQL e tutti gli strumenti necessari per utilizzare i linguaggi di programmazione PHP e Perl. Per separare meglio le varie funzionalità dalle varie interfacce grafiche, utilizzo il pattern MVC (Model View Controller).

Per realizzare un sito professionale con il ridimensionamento delle varie pagine utilizzo la libreria Bootstrap che implementa già del codice CSS per lo stile e del JS per le varie animazioni. Ho utilizzato la libreria Font Awesome che permette di aggiungere delle icone per rendere il sito web più bello. Per immagazzinare i dati dei vari utenti viene implementato una banca dati in MySQL.

Per potere operare efficacemente nel dominio bisogna acquisire buone conoscenze teoriche d'informatica. Bisogna sapere programmare nei linguaggi web (HTML, CSS, PHP), si deve anche sapere gestire un database conoscendo il linguaggio SQL e per finire bisogna conoscere come funziona un web server. Oltre alla programmazione web, bisogna sapere anche come configurare delle apparecchiature informatiche come il Raspberry all'interno di una rete.

3.2 Analisi API Nagios

Nagios permette di monitorare l'intera infrastruttura IT per potere garantire che i sistemi, applicazioni, servizi e processi aziendali funzionino correttamente. In effetti in caso di guasto, Nagios può avvisare il personale tecnico del problema, consentendo loro di avviare i processi di riparazione prima che le interruzioni influiscano sui processi aziendali, sugli utenti finali o sui clienti.

Questo capitolo non spiega come installare Nagios, perché il sistema è già stato installato e configurato dai sistemisti della scuola. Però viene descritto in modo dettagliato i vari parametri, le varie funzionalità e i vari processi che offre l'API Nagios.

Nagios include le seguenti funzionalità:

- Monitoraggio dei servizi di rete (SMTP, POP3, HTTP, NNTP, PING, ecc.)
- Monitoraggio delle risorse host (carico del processore, utilizzo del disco, ecc.)
- Design semplice del plug-in che consente agli utenti di sviluppare facilmente i propri controlli di servizio.
- Controlli di servizio paralleli
- Capacità di definire la gerarchia degli host di rete utilizzando un host "padre", consentendo il rilevamento e la distinzione tra host inattivi e non raggiungibili.
- Notifiche di contatto quando si verificano dei problemi con il servizio o l'host e vengono risolti (tramite e-mail, cerca persone o metodo definito dall'utente).
- Possibilità di definire gestori di eventi da eseguire durante eventi di servizio o host per la risoluzione proattiva dei problemi.
- Rotazione automatica dei file di registro.
- Supporto per l'implementazione di host di monitoraggio ridondanti.
- Interfaccia web opzionale per visualizzare lo stato corrente della rete, notifiche e cronologia dei problemi, file di registro, ecc.

Nella foto sottostante viene raffigurata la struttura gerarchica di Nagios e si può notare che quest'ultimo è suddiviso in diversi file di configurazione.

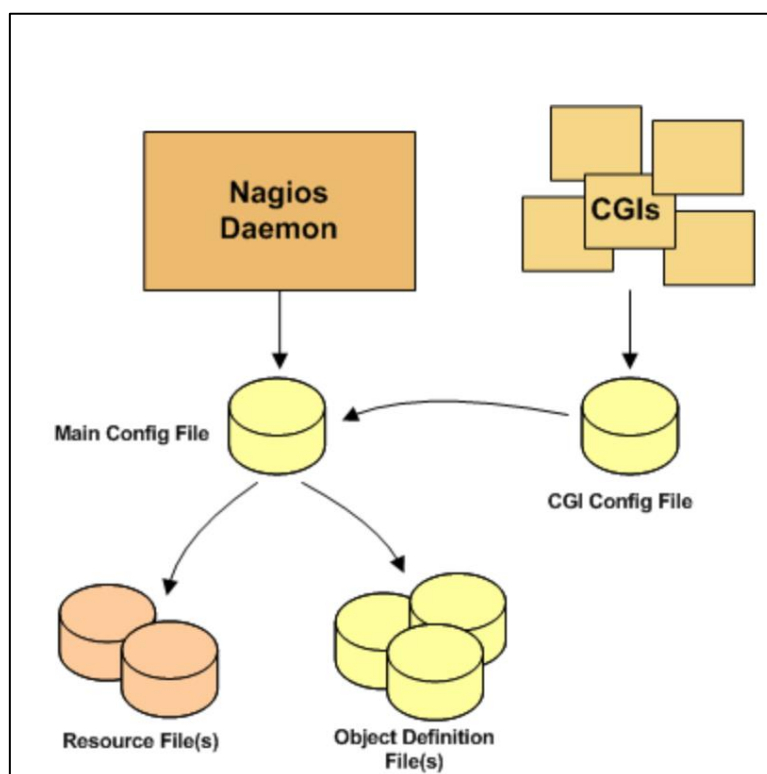


Figura 1: Struttura Nagios

Il file di configurazione principale (**Main Config file**) contiene una serie di direttive che influenzano il funzionamento del demone Nagios. Questo file di configurazione viene letto sia dal demone di Nagios che dai CGI.

I file di risorse (**Resource File**) possono essere utilizzati per memorizzare macro definite dall'utente. Principalmente il file di risorse viene utilizzato per memorizzare delle informazioni di configurazione molto sensibili (come le password) senza renderli disponibili ai CGI.

I file di definizione degli oggetti (**Object Definition File**) vengono utilizzati per definire host, servizi, gruppi di host, contatti, gruppi di contatti, comandi, ecc. In effetti in questi file vengono definite tutte le cose che si desiderano monitorare e come si desiderano monitorarle.

Il file di configurazione CGI (**CGI Config File**) contiene una serie di direttive che influenzano il funzionamento dei CGI. Contengono anche un riferimento al file di configurazione principale, in modo che i CGI sappiano come è stato configurato Nagios e dove sono stati archiviati le definizioni degli oggetti.

Per potere interagire con l'API ho utilizzato principalmente i file CGI. Questi file richiedono un'autenticazione sul server web e che sei autorizzato a visualizzare tutte le informazioni che stai richiedendo. Qui sotto vengono descritti in modo dettagliato i vari file CGI, ma quelli che ho utilizzato per il progetto sono il file **statusjson.cgi** e **statusmap.cgi**.

Il file CGI più importante è **status.cgi** e quest'ultimo consente di visualizzare lo stato corrente di tutti gli host e servizi che vengono monitorati. Questo file può produrre due tipi principali di output: una panoramica dello stato di tutti i gruppi host (o un particolare gruppo host) e una vista dettagliata di tutti servizi (o di quelli associati a un particolare host)

Il file **statusmap.cgi** permette di creare una mappa di tutti gli host che hai definito sulla propria rete. Per creare l'immagine PNG del layout di rete viene utilizzata la libreria gd di Thomas Boutell (versione 1.6.3 o successiva). Le coordinate utilizzate per disegnare ogni host (insieme alle icone opzionali) sono prese automaticamente le coordinate del disegno.

Il file **statuswml.cgi** funge da interfaccia WAP per le informazioni sullo stato della rete. Se si dispone di un dispositivo abilitato per WAP (ovvero un telefono cellulare predisposto per Internet), si può visualizzare le informazioni sullo stato mentre si è in viaggio. Vengono incluse diverse visualizzazione dello stato che sono il riepilogo dell'host, i dettagli del servizio, tutti i problemi e i problemi non ancora gestiti. Oltre alla visualizzazione delle informazioni sullo stato, si può anche disabilitare le notifiche, i controlli e riconoscere i problemi del proprio cellulare.

Il file **statuswrl.cgi** permette di creare un modello VRML 3D di tutti gli host che sono stati definiti sulla rete. Le coordinate utilizzate per disegnare ogni host vengono prese dalle definizioni degli host. Per visualizzare questo tipo di file bisogna avere un browser VRML (come Cortona, Cosmo Player o WorldView) installato sul proprio sistema.

Il file **tac.cgi** è stato progettato per fungere da "vista dall'alto" di tutte le attività di monitoraggio della rete. Consente di visualizzare rapidamente le interruzioni di rete, lo stato dell'host e lo stato del servizio. Distingue tra problemi che sono stati "gestiti" e "non gestiti". Molto utile se si ha molti host o servizi da monitorare e si deve mantenere un unico schermo per avvisare del problema.

Il file **outages.cgi** permette di produrre un elenco dei vari host "problematici" sulla rete che causano delle interruzioni di rete. Molto utile se si dispone di una rete di grandi dimensioni e si desidera identificare rapidamente l'origine del problema. Inoltre, gli host vengono ordinati in base alla gravità dell'interruzione che ha causata.

Il file **config.cgi** consente di visualizzare gli oggetti (ad esempio host, gruppi, contatti, gruppi di contatti, periodi di tempo, servizi, ecc.) Questi oggetti vengono definiti nel file di configurazione degli oggetti.

Il file **cmd.cgi** consente di inviare dei comandi al processo Nagios. Sebbene questo CGI ha diversi argomenti, sarebbe meglio lasciarli soli. In effetti tra le diverse revisioni di Nagios ci sono tante differenze.

Il file **extinfo.cgi** consente di visualizzare le informazioni sul processo di Nagios, le statistiche sullo stato dell'host e del servizio, i commenti sull'host e sul servizio e altro ancora. Serve anche come punto di partenza per l'invio di comandi a Nagios tramite il comando CGI. Inoltre, è possibile raggiungere questo file cliccando sul collegamento "Stato della rete" e "Informazioni sul processo" nella barra di navigazione laterale e facendo clic su un collegamento host o servizio nell'output del CGI di stato.

Il file **showlog.cgi** permette di visualizzare il file di registro. Se la rotazione del registro è abilitata, è possibile sfogliare le notifiche presenti nei file di registro archiviati utilizzando i collegamenti di navigazione nella parte superiore della pagina.

Il file **history.cgi** viene utilizzato per potere visualizzare la cronologia dei problemi con un particolare host o con tutti gli host. L'output è fondamentalmente un sottoinsieme delle informazioni visualizzate dal file di registro. Si ha la possibilità di filtrare l'output per potere visualizzare solamente degli specifici tipi di problemi (ad esempio allarmi hard o soft, vari tipi di avvisi di servizio e host, tutti i tipi di avvisi, ecc.).

Il file **notifications.cgi** viene utilizzato per potere visualizzare le notifiche dell'host e del servizio che sono state inviate ai vari contatti. C'è anche la possibilità di visualizzare solamente delle specifiche notifiche (ad esempio notifiche di servizio, notifiche di host, notifiche inviate a contatti specifici, ecc.).

Il file **trends.cgi** viene utilizzato per creare un grafico degli stati dell'host o del servizio in un periodo di tempo arbitrario. Questo file è di grande utilità solo se è stato abilitato la rotazione dei log e se vengono mantenuti i log archiviati nel percorso specifico della direttiva (log_archive_path). Come il file **statusmap.cgi** necessita della libreria gd di Thomas Boutell.

Il file **avail.cgi** viene utilizzato per segnalare la disponibilità di host e servizio per un periodo di tempo specificato dall'utente. Anche in questo file è necessario abilitare la rotazione dei log e di mantenere i log archiviati nel percorso specificato da

Il file **histogram.cgi** permette di segnalare su un grafico istogramma la disponibilità di host e servizio per un periodo di tempo specificato dall'utente. In effetti per potere creare il grafico istogramma, bisogna installare la libreria gd di Thomas Boutell

Il file **summary.cgi** fornisce alcuni rapporti generici sui dati di avviso dell'host e del servizio, inclusi i totali degli avvisi, i principali produttori di avvisi, ecc.

Un dei file importanti per potere ricavare i dati dell'API è il file **statusjson.cgi**. L'obiettivo di questo file è di fornire tutte le informazioni negli attuali CGI in formato JSON. Tutte le operazioni vengono eseguite lato server e le CGI forniscono un'API per potere interrogare tutti gli oggetti, stato e informazioni storiche tramite richieste GET. L'autenticazione è uguale a quella degli altri file CGI e una volta interrogato il file, restituisce un JSON valido che può essere analizzato in oggetti JavaScript per i modelli e l'elaborazione lato client. L'API è molto robusta e fornisce diversi modi per limitare le query: nome / descrizioni, gruppi di host / servizi, tempi di aggiornamento / modifiche, ecc.

Nell'immagine sottostante si può notare un esempio di query per potere ricavare dei dati di un determinato host.

monitor.cpt.local/nagios/cgi-bin/statusjson.cgi?query=host&formatoptions=enumerate&hostname=sd2048_21

Figura 2: Nagios query

Per potere eseguire una query sul file **statusjson.cgi** bisogna scrivere inseguito la stringa **?query**. Il parametro **host** permette di ricavare tutte le informazioni di un host. Il parametro **formatoptions** permette di definire il formato da ritornare o da visualizzare sul browser, ovvero sul browser si visualizzerà in formato JSON tutte le informazioni. Il parametro **hostname** permette di definire il nome dell'host. Per potere ricavare la lista di tutti i servizi dei vari host, bisogna scrivere **servicelist** al posto del parametro **host**. Per potere ricavare tutti i dettagli dei vari servizi, bisogna aggiungere il parametro booleano **details**. Per finire un parametro per potere filtrare i tipi di stato dei servizi è **servicestatus**.

Nel file JSON si può notare che ogni stato del servizio corrisponde ad un codice numerico. Lo stato OK del servizio corrisponde al codice **2**, lo stato WARNING corrisponde al codice **4**, lo stato UNKNOWN corrisponde al codice **8** e per finire lo stato CRITICAL corrisponde al numero **16**.

3.3 Analisi e specifica dei requisiti

Spiegazione elementi tabella dei requisiti:

ID: identificativo univoco del requisito

Nome: breve descrizione del requisito

Priorità: indica l'importanza di un requisito nell'insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

Versione: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l'ultima versione, mentre le vecchie dovranno essere inserite nei diari.

Categoria: campo di competenze informatiche

Note: eventuali osservazioni importanti o riferimenti ad altri requisiti.

Sotto requisiti: elementi che compongono il requisito

| ID: REQ-001 | |
|------------------|--|
| Nome | Realizzare un applicativo web |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio web |
| Note | Bisogna implementare un web server e scaricare il template bootstrap |
| Sotto requisiti | |
| 001 | L'applicativo deve funzionare sui browser più utilizzati |
| 002 | L'applicativo web implementa una struttura web MVC |
| 003 | L'applicativo web implementa un template bootstrap |
| 004 | Le varie pagine possono essere ridimensionabili |

| ID: REQ-002 | |
|------------------|---|
| Nome | Creazione pagina di gestione degli utenti |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio SQL e gestione MySQL |
| Note | Tramite una pagina di gestione possono essere creati degli utenti |
| Sotto requisiti | |
| 001 | Realizzare gli utenti amministratori + permessi |

| | |
|------------|--|
| 002 | Realizzare gli utenti limitati+ permessi |
|------------|--|

| ID: REQ-003 | |
|------------------|--|
| Nome | Creazione pagina di login |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio web |
| Note | Per accedere all'applicativo web deve essere un login di accesso |
| Sotto requisiti | |
| 001 | Sistema di recupero password dimenticata |

| ID: REQ-004 | |
|------------------|--|
| Nome | Creazione pagina di amministrazione |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio web |
| Note | Pagina dove un amministratore può selezionare i campi da rendere visibile tramite "check". |
| Sotto requisiti | |
| 001 | Sistema per ottenere informazioni da Nagios |
| 002 | Sistema di "check" |

| ID: REQ-005 | |
|------------------|---|
| Nome | Creazione pagina di visualizzazione |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio web |
| Note | Pagina dove vengono visualizzate le informazioni presenti nell'applicativo Nagios |
| Sotto requisiti | |
| 001 | Creazione parte sinistra dove vengono visualizzate le informazioni |
| 002 | Creazione parte destra dove viene visualizzato la mappa della rete |

| ID: REQ-006 | |
|------------------|--|
| Nome | Creazione mappa della rete del CPT di Trevano. |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio e rete |
| Note | Raspberry che viene utilizzato per interfacciare il monitor esterno con la rete. |
| Sotto requisiti | |
| 001 | Creare un “check” per attivare la mappa della rete |
| 002 | Utilizzare le API Nagios MAP |

| ID: REQ-007 | |
|------------------|--|
| Nome | Installazione e configurazione Raspberry |
| Priorità | 1 |
| Versione | 1.0 |
| Categoria | Linguaggio e rete |
| Note | Raspberry che viene utilizzato per interfacciare il monitor esterno con la rete. |
| Sotto requisiti | |
| 001 | Collegarsi al Raspberry con SSH |

3.4 Use case

Per questo progetto ho deciso di separare in due parti lo use case che sono la parte sistemistica con l'interfacciamento del Raspberry e la seconda parte dedicata all'applicativo web. Nella parte sistemistica abbiamo l'attore Raspberry PI che è collegato al monitor esterno. Il monitor esterno avrà uno sfondo bianco sul quale verranno visualizzate le informazioni importanti, quindi gli allarmi e la possibilità di vedere anche graficamente gli equipaggiamenti presi in considerazione nel monitoraggio. Un utente potrà accedere al Raspberry via SSH. Invece per quanto riguarda l'applicativo web abbiamo l'attore utente che può avere due permessi. Se l'utente è amministratore può accedere completamente a tutte le funzionalità dell'applicativo web, invece se l'utente è limitato può solamente visualizzare le informazioni dei vari equipaggiamenti. Quando viene creato un nuovo utente c'è un sistema di email che permette di inviare ad un utente la password provvisoria che verrà modificata al primo login dell'utente. Nell'applicativo web abbiamo una pagina che permette di gestire gli utenti. Nella pagina di amministrazione l'utente amministratore può visualizzare in una tabella tutte le informazioni presenti nell'applicativo Nagios e tramite dei vari "check" può selezionare i campi che devono essere visibili sullo schermo. Inoltre sempre nella stessa pagina abbiamo anche un "check" che permette di attivare la mappa di rete della CPT da rendere visibile sullo schermo. Sia la pagina di gestione degli utenti e sia la pagina di amministrazione è accessibile solamente da un utente di tipo amministratore. La pagina di visualizzazione è la pagina web versione desktop che permette di visualizzare le informazioni importanti, ovvero gli allarmi.

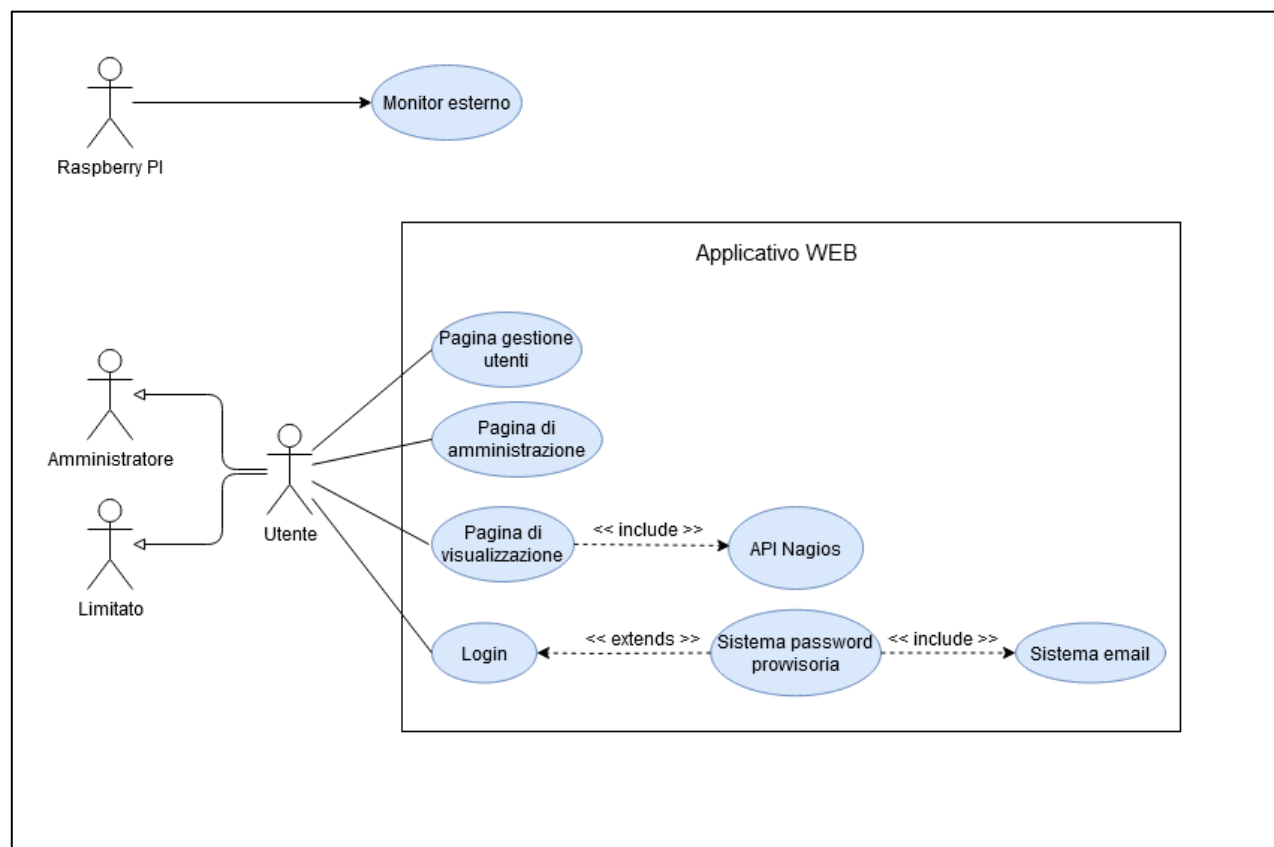


Figura 3: Use case

3.5 Pianificazione

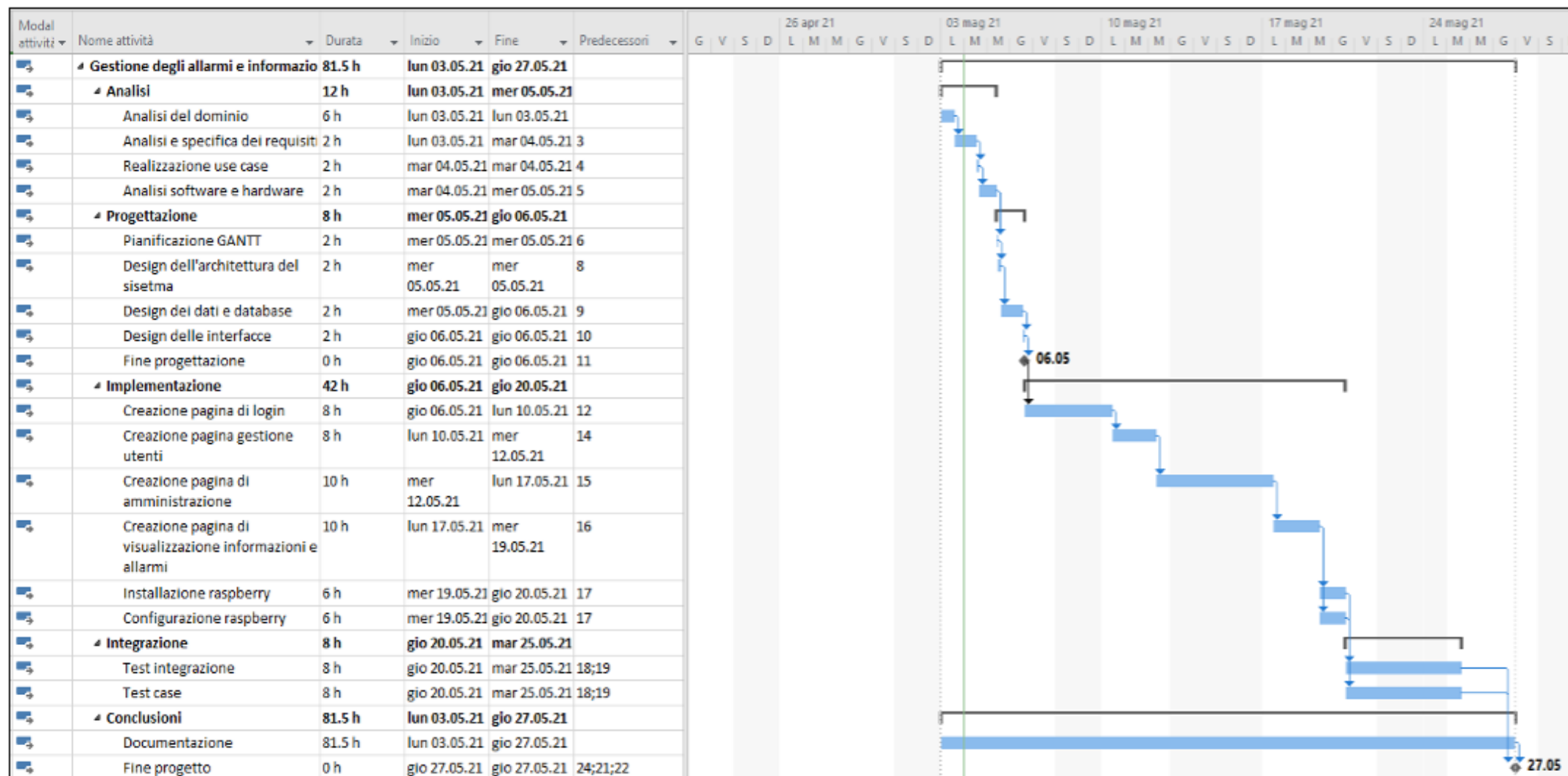


Figura 4: GANTT Preventivo


La pianificazione preventiva è stata suddivisa in sequenze lineare di fasi che corrispondono all'analisi, alla progettazione, all'implementazione, all'integrazione e alle conclusioni. L'analisi è una fase molto importante del progetto e quest'ultima è stata suddivisa in quattro attività. In queste attività ho cercato di comprendere lo scopo del mio progetto e i vari requisiti richiesti dal committente. In effetti nell'analisi ho realizzato anche la specifica dei requisiti. La progettazione è la seconda fase è l'ho suddiviso in cinque attività. Questa fase contiene tutte le attività legate al design, ovvero il design dell'architettura (schema di rete), design dei dati e database, design delle interfacce. Oltre al design, ho anche ho anche realizzato la pianificazione del GANTT che dovrò seguire per tutta la durata del progetto. Ritengo queste attività molto importanti perché mi serviranno per le fasi successive della pianificazione. Alla fine della progettazione ho creato una milestone che indica la fine della progettazione e l'inizio dell'implementazione. La fase principale e quella più lunga di questo progetto è l'implementazione. Questa fase corrisponde alla realizzazione dell'applicativo web e della configurazione del Raspberry. Tutte le attività presenti in questa fase sono legate strettamente all'analisi e alla progettazione in quanto si basa su di essa per lo sviluppo. Un'altra fase molto importante è l'integrazione e al suo interno sono presenti tutte le attività legate ai vari test che verranno eseguiti su ogni metodo e funzioni dell'applicativo web. Inoltre, vengono anche eseguiti i test case che permettono di verificare il funzionamento dei vari requisiti. L'ultima fase importante del progetto è la conclusione che consiste nella realizzazione della documentazione. La documentazione nel mio progetto è molto fondamentale, perché verrà utilizzata dagli sistemisti per realizzare il prodotto. La documentazione deve essere realizzata e aggiornata su tutto il periodo del progetto. Per finire ho aggiunto un milestone che indica la fine del progetto.

3.6 Analisi dei mezzi

3.6.1 Software

Per la realizzazione del mio progetto ho utilizzato i seguenti software:

| | |
|--|---|
| Apache 2.4.46 (XAMPP) Apache 2.4.38 (Raspberry) | Apache è già installato su XAMPP e dovrò installarlo sul Raspberry. Quest'ultimo permette di creare un web server sul quale implementare l'applicativo web. |
| MariaDB 10.4.17 (XAMPP) MariaDB 10.3.27 (Raspberry) | MariaDB è un sistema di gestione del database relazionale (DBMS) open source che è un sostituto drop-in compatibile per la tecnologia di database MySQL ampiamente utilizzata. Questo software è già installato su XAMPP. |
| PHP 8.0.1 (XAMPP) PHP 7.3 (Raspberry) | PHP è già installato su XAMPP e dovrò solamente installarlo sul Raspberry. Quest'ultimo mi permette di collegarmi al database e di realizzare tutte le funzioni dell'applicativo web |
| XAMPP Control Panel Version 3.2.4 | XAMPP è software multiplatforma e libera costituita da Apache http Server, il database MySQL e tutti gli strumenti necessari per utilizzare la programmazione PHP e Perl. Utilizzo questo software per implementare il mio applicativo web. Quando il mio prodotto dovrà andare in produzione installerò tutti i pacchetti software sul Raspberry per creare un web server. |
| phpMyAdmin 5.0.4 | PhpMyAdmin è uno strumento software gratuito implementato in PHP ed è destinato a gestire l'amministrazione di MySQL sul Web. Ho utilizzato il seguente software per collegarmi al database via web. |

| | | |
|--|--|-----------------|
|  | SAMT – Sezione Informatica | Pagina 14 di 59 |
| | Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite API | |

| | |
|-------------------------------|--|
| Draw.io | <p>Draw.io è una piattaforma online che permette di creare diagrammi direttamente dal proprio browser Web, senza dover utilizzare software o app. Ho utilizzato il seguente applicativo per creare i vari schemi ER, lo schema di rete e lo sitemap.</p> <p>Link: http://draw.io.</p> |
| Microsoft Project 2016 | <p>Microsoft Project è un software di pianificazione (utilizzato anche nel project management) sviluppato e venduto da Microsoft. Project è uno strumento che permette di assistere al project manager nella pianificazione, nell'assegnazione delle risorse, nella verifica del rispetto dei tempi, nella gestione dei budget e nell'analisi dei carichi di lavoro. Ho utilizzato questo strumento per pianificare le varie attività che devo realizzare per il progetto.</p> |
| Mockflow | <p>MockFlow è un potente strumento per disegnare wireframe dell'interfaccia utente che si estende anche come suite di pianificazione completa per la progettazione del prodotto. Ho utilizzato il seguente applicativo web per potere realizzare i mockup dell'applicativo web.</p> <p>Link: https://www.mockflow.com/</p> |

3.6.2 Librerie

| | |
|----------------------------|---|
| Bootstrap 5.0.0 | <p>Questo strumento è stato utilizzato per realizzare la parte grafica dell'applicativo web, dando degli strumenti performanti e puliti con un'ottima documentazione. Ad esempio, può essere utilizzata per il responsive della pagina web.</p> <p>Link: https://getbootstrap.com.</p> |
| Font-awesome 5.15.3 | <p>Font Awesome è un toolkit di icone distribuito sotto licenze libere. Font Awesome contiene delle icon fonts vettoriali già pronte che sono la soluzione molto usata per inserire nei propri progetti web. Utilizzo questa libreria per inserire delle icone all'interno del mio sito web.</p> <p>Link: https://fontawesome.com</p> |
| DataTables | <p>DataTables è un plug-in per la libreria JQuery Javascript. È uno strumento altamente flessibile, costruito sulle basi del miglioramento progressivo, che aggiunge tutte queste funzionalità avanzate a qualsiasi tabella HTML. Utilizzo questa libreria per aggiungere alle tabelle del mio sito delle funzionalità come la paginazione, ricerca istantanea, ordinamento di più colonne, ecc.</p> <p>Link: https://datatables.net</p> |
| PHPMailer | <p>PHPMailer è una libreria che permette di inviare e-mail in modo sicuro e facilmente tramite codice PHP da un server web (MUA al server MSA). L'invio di e-mail direttamente tramite codice PHP richiede una familiarità di alto livello con il protocollo standard SMTP e le questioni correlate (come il ritorno a capo) e le vulnerabilità sull'iniezione di posta elettronica per spamming. Ho utilizzato questa</p> |

| | |
|---------------------|--|
| | libreria per implementare un sistema che invia e-mail quando viene creato nuovo utente. Link: https://github.com/PHPMailer/PHPMailer . |
| JQuery 3.6.0 | JQuery è una libreria JavaScript per applicazioni web. Permette di semplificare la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML. Link: https://jquery.com/download/ |

3.6.3 Hardware

Visto che questo progetto è interamente lato web, non ho dovuto utilizzare delle macchine virtuali per lo sviluppo. Ho semplicemente utilizzato il computer fisso che mi è stato fornito dalla scuola e con quest'ultimo ho potuto implementare un applicativo web utilizzando il software XAMPP. Su XAMPP sono installati tutti i tools per lo svolgimento del progetto (Apache, MySQL, PHP, ecc).

Caratteristiche del computer:

- Hp elitedesk 800 G4 torre con 32 GB di memoria RAM, Intel Core I7 e 476 GB di archiviazione. Sopra è installato il sistema operativo Windows 10 Enterprise 64 bit.

Per interfacciare il monitor esterno nella rete della CPT ho utilizzato un Raspberry. Inoltre il Raspberry sarà anche il web server del mio applicativo web.

Caratteristiche del Raspberry:

- Raspberry Pi 4 Computer Model B con 8 GB di memoria RAM, due porte micro HDMI, 2 porte USB 2.0 e 2 porte USB 3.0, Gigabit Ethernet, alimentazione di ingresso 5,1V tramite connettore USB-C.

4 Progettazione

4.1 Design dell'architettura del sistema

Questo è lo schema dell'architettura del sito web semplificato, l'applicativo web interagisce con il database utilizzando dei metodi messi a disposizione da PHP per potere interrogare una banca dati MySQL. Nel database vengono memorizzati tutti gli utenti che vengono creati e le varie informazioni importanti dei vari equipaggiamenti che vengono ricavate tramite l'API Nagios. In effetti nell'architettura abbiamo anche l'API Nagios che permette di monitorare un sito che è già stato implementato dai sistemisti. Nel sito web è presente un sistema che permette di salvare in modo automatico le informazioni importanti del monitoraggio di Nagios. Inseguito le informazioni vengono salvate nel database, in effetti la tabella degli allarmi viene aggiornata continuamente. L'applicativo Nagios è stato implementato dagli sistemisti su un server della CPT e permette di monitorare tutti gli equipaggiamenti produttivi nella rete informatica del CPT di Trevano. Per potere interagire con l'API, il server è raggiungibile via web.

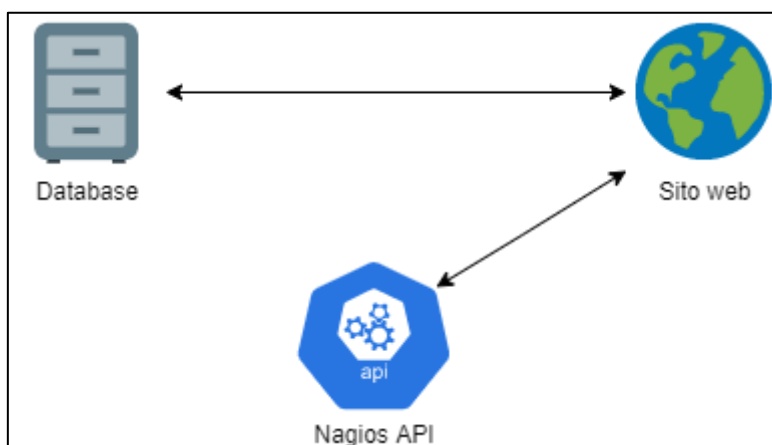


Figura 5: Architettura del sistema

L'applicativo web è composto da diverse pagine web, la prima pagina web che viene mostrata è la pagina di login dove un utente ha la possibilità di accedere all'applicativo web tramite un form. Nella pagina di login se l'utente non si ricorda della propria password può cliccare un link che lo porta alla pagina di modifica della password. Una volta eseguito il login l'utente può accedere a tre pagine in base ai propri permessi che gli sono stati attribuiti. Un amministratore può accedere alla pagina di gestione degli utenti dove ha la possibilità di creare, modificare e eliminare un utente. Inoltre, può anche cambiare i permessi degli utenti. Invece un utente limitato può solamente modificare i propri dati del suo profilo, come ad esempio la foto di profilo, l'email, il nome, ecc. Nella pagina di amministrazione può solamente accedere un amministratore e quest'ultimo può selezionare dei "check" per attivare o disattivare la visibilità di alcuni campi delle informazioni (host, duration, last check, ecc.). Per finire l'ultima pagina permette di visualizzare gli allarmi e quest'ultima è accessibile sia dall'amministratore e sia da un utente limitato. Esistono due versioni di questa pagina, la prima è la versione desktop che si può accedere tranquillamente dalla propria postazione e la seconda è solamente visibile sul monitor esterno.

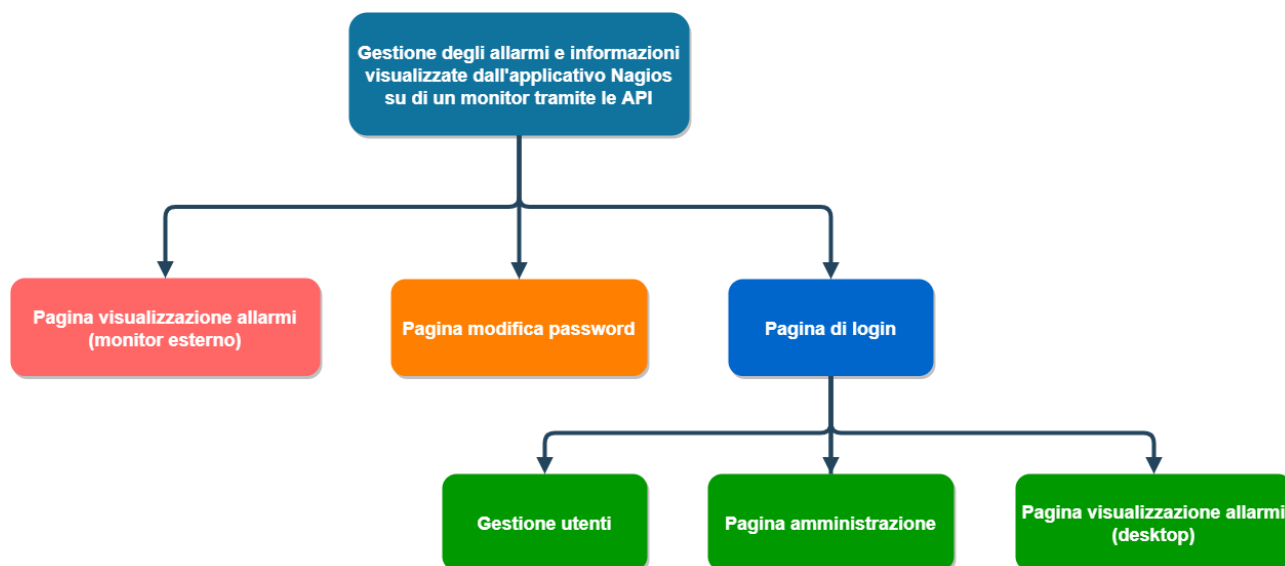


Figura 6: Sitemap

4.2 Design dei dati e database

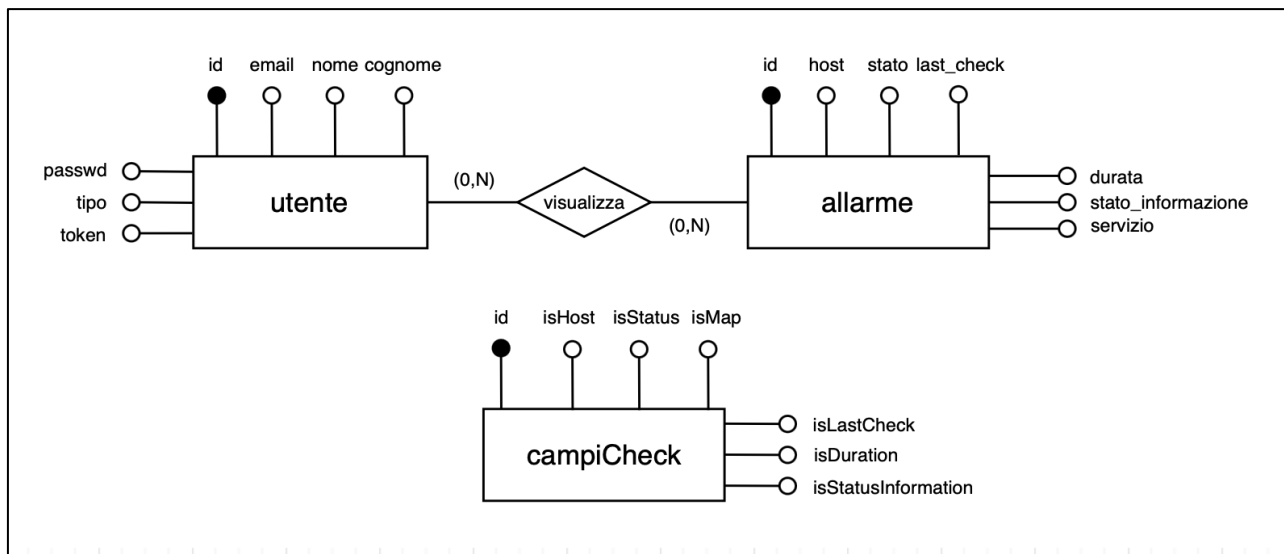


Figura 7: Schema ER

Questo è lo schema ER della banca di dati ed è composto da tre tabelle che sono **utente**, **allarme** e **campiCheck**. La tabella **utente** contiene tutte le informazioni degli utenti che possono accedere all'applicativo web. L'attributo **tipo** permette di definire i permessi degli utenti, cioè amministratore e limitato. L'attributo **token** è un codice che viene utilizzato per eseguire il recupero della password. La tabella **allarme** permette di memorizzare le informazioni importanti, ovvero gli allarmi che saranno visibili sul monitor esterno. Ogni attributo della tabella corrisponde ai vari campi che dovranno essere visibili sul monitor esterno. L'attributo **servizio** corrisponde al servizio dell'allarme, ovvero il servizio sul quale c'è un problema. Per finire la tabella **campiCheck** permette di memorizzare lo stato dei vari check che permettono di attivare o disattivare i campi da rendere visibili sullo schermo. L'attributo **isMap** permette di definire se bisogna visualizzare sullo schermo la mappa di rete di tutte le apparecchiature.

4.2.1 Schema logico

utente(id, email, nome, cognome, passwd, tipo token)

allarme(id, host, servizio, stato, last_check, durata, stato_informazione)

campiCheck(id, isHost, isStatus, isMap, isLastCheck, isDuration, isStatusInformation)

4.2.2 Descrizione tabella

| utente | |
|---------------|--|
| Attributo | Descrizione |
| id | Rappresenta l'identificativo di un utente. Attributo di tipo intero e viene generato automaticamente ogni volta che inserisco un utente nell'applicativo. Non può essere nullo. Esempio: 1 |
| email | Rappresenta l'e-mail di un utente. Attributo di tipo stringa con il limite è di 50 caratteri. Non può essere nullo, ma deve essere univoca. Esempio: pierpaolo.casati@samtrevano.ch |
| nome | Rappresenta il nome di un utente. Attributo di tipo stringa con il limite è di 50 caratteri. Non può essere nullo e univoco. Esempio: Pierpaolo |
| cognome | Rappresenta il cognome di un utente. Attributo di tipo stringa con il limite è di 50 caratteri. Non può essere nullo e univoco. Esempio: Casati |
| passwd | Rappresenta la password di un utente. Attributo di tipo stringa con il limite 255 caratteri. Non può essere nullo e univoco. Nel campo verrà salvata la hash della password che verrà generata dal sistema. Esempio: \$2y\$10\$cRc/Vhu2SSEdVd3VR6DI.Ogqk/w8000Gj1y6StFx8R63o9oK1/LLS |
| tipo | Rappresenta il tipo di utente. Attributo di stringa con un valore scelto dall'elenco di valori amministratore e limitato. Non può essere nullo e univoco. Esempio: amministratore |
| token | Rappresenta un codice che verrà utilizzato per eseguire il recupero della password. Questo codice verrà generato in modo casuale. All'interno dell'attributo verrà salvato un hash in SHA256 del token di recupero password. Attributo di tipo stringa con il limite è di 32 caratteri. Esempio: 462ce1d38046858a80132ba59cfcac82 |

| allarme | |
|--------------------|--|
| Attributo | Descrizione |
| id | Rappresenta l'identificativo di un allarme. Attributo di tipo intero e viene generato automaticamente ogni volta che l'API Nagios ha delle nuove informazioni da mostrare sullo schermo. Non può essere nullo. Esempio: 1 |
| host | Rappresenta il nome del host. Attributo di tipo stringa con limite di 255 limite. Non può essere nullo e univoco. Esempio: localhost |
| servizio | Rappresenta il tipo di servizio presente nell'API Nagios. Attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo e univoco. Esempio: HTTP |
| stato | Rappresenta lo stato dell'informazione. Attributo di tipo stringa con il limite è di 50 caratteri. Non può essere nullo e univoco. Esempio: UP |
| last_check | Rappresenta la data dell'ultimo controllo eseguito sull'equipaggiamento. Attributo di tipo datetime. Il formato della data deve essere YYYY-MM-DD. Non può essere nullo e univoco. Esempio: 2021-05-06 13:31:00 |
| durata | Rappresenta la durata di un'informazione. Attributo di tipo stringa con limite di 20 caratteri. Non può essere nullo e univoco. Esempio: 0d 13h 10m 18s |
| stato_informazione | Rappresenta lo stato di un'informazione presente nell'API Nagios. Attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo e univoco. Esempio: SERVICE_WARNING |

| campiCheck | |
|---------------------|--|
| Attributo | Descrizione |
| id | Rappresenta l'identificativo della tabella campiCheck, ovvero quei check permettono di attivare o disattivare i campi da rendere visibile sullo schermo. Non può essere nullo. Esempio: 1 |
| isHost | Rappresenta lo stato di visibilità del campo host. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: true |
| isStatus | Rappresenta lo stato di visibilità del campo stato. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: true |
| isLastCheck | Rappresenta lo stato di visibilità del campo last_check. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: false |
| isDuration | Rappresenta lo stato di visibilità del campo durata. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: false |
| isStatusInformation | Rappresenta lo stato di visibilità del campo informazione di stato. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: true |
| isMap | Rappresenta lo stato per rendere visibile la mappa di rete. Corrisponde ad un valore booleano che non può essere nullo e univoco. Esempio: false |

4.3 Design delle interfacce

Figura 8: Design pagina di login

Questo mockup rappresenta la pagina di login per accedere all'applicativo web. Al centro della pagina è presente un form con due campi. Il primo campo corrisponde allo username (l'utente deve scrivere la propria email) e il secondo campo rappresenta la password. Una volta compilato i campi un utente può cliccare il pulsante login per potere accedere alle varie pagine di gestione e alla pagina di visualizzazione degli allarmi. Sotto il pulsante di "Login" è presente un link che permette di reindirizzare l'utente alla pagina di modifica della password.

| Id | Nome | Cognome | Email | Tipo | Modifica | Elimina |
|----|------|---------|----------|----------|----------|---------|
| 1 | Pier | Casati | p.c@g.ch | limitato | Modifica | Elimina |

Figura 9: Design pagina di gestione degli utenti

Il seguente mockup rappresenta la pagina di gestione degli utenti. In alto è presente una barra di navigazione per potere navigare nelle varie pagine dell'applicativo web. Nella pagina è presente una tabella scritto al suo interno tutte le informazioni dei vari utenti. Per ogni utente un amministratore può cliccare il pulsante "Modifica"

per potere modificare i permessi o può cliccare il pulsante “Elimina” per potere eliminare un utente. Se viene cliccato il pulsante “Aggiungi” per potere creare un nuovo utente.

Figura 10: Design pagina per creare un nuovo utente

Questo mockup rappresenta la pagina per potere creare un nuovo utente. Nella pagina è presente un form con dei vari campi. Quando viene creato un utente bisogna fornire un nome, un cognome, un email e dei permessi. Se viene cliccato il pulsante “Aggiungi”, viene creato un nuovo utente. Invece se viene cliccato il pulsante “Esci” si ritorna alla pagina di gestione degli utenti.

Figura 11: Design pagina di amministrazione

Questo mockup rappresenta la pagina di amministrazione che è solamente accessibile da un amministratore. In alto è sempre presente una barra di navigazione per accedere alle varie pagine dell'applicativo web. Nella pagina è presente una tabella con le varie informazioni importanti dei vari equipaggiamenti. I dati presenti nella

tabella verranno anche visualizzati nella pagina di visualizzazione degli allarmi. L'utente può selezionare diversi "check" per potere attivare i vari campi da rendere visibili.

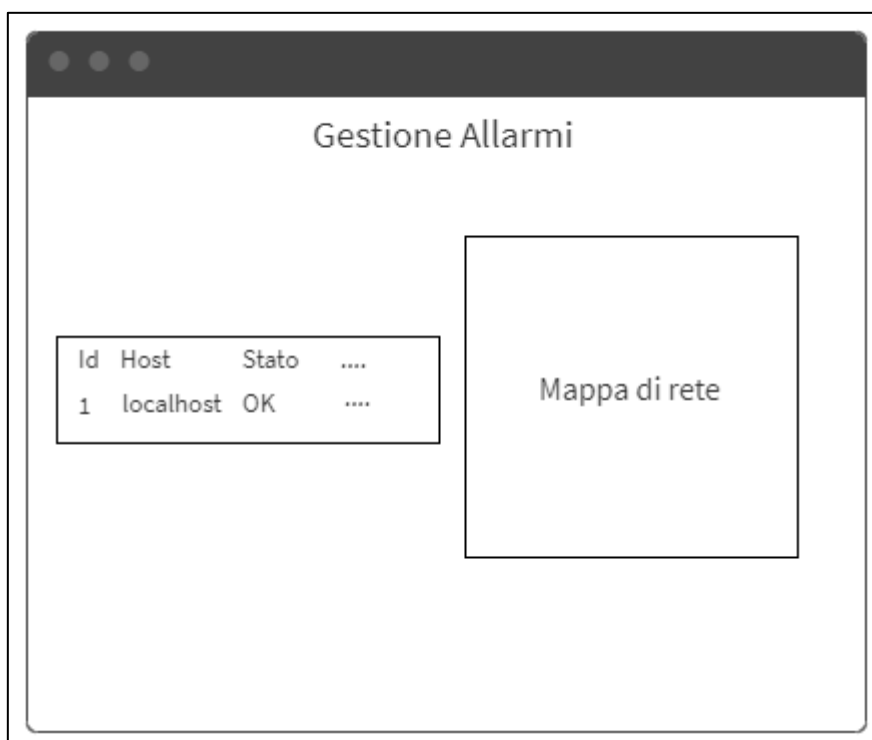


Figura 12: Design pagina di visualizzazione degli allarmi

Questo mockup rappresenta la pagina di visualizzazione degli allarmi. La pagina è suddivisa in due parti, sulla sinistra è presente la tabella con le varie informazioni importanti. Invece sulla destra è presente una mappa della rete dei vari equipaggiamenti produttivi della CPT. Se l'amministratore non ha attivato il "check" per la visualizzazione della mappa di rete, la tabella delle informazioni occupa tutto lo spazio della pagina.

4.4 Design procedurale

4.4.1 UML Models

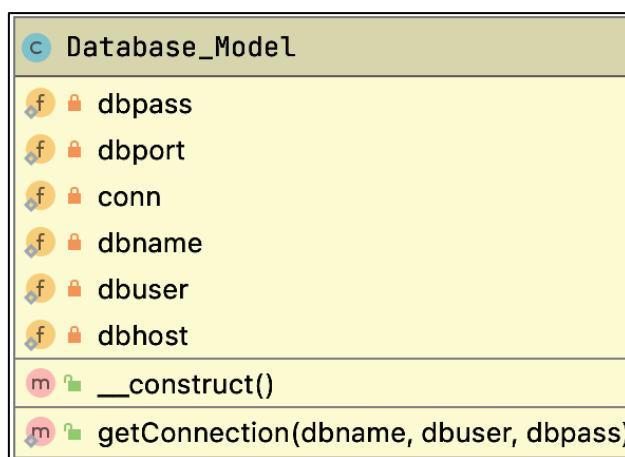


Figura 13: Model database

La classe **Database_Model** è il model che permette di accedere al database dell'applicativo web. L'attributo **dbpass** corrisponde alla password per accedere al database. L'attributo **dport** corrisponde alla di ascolto del database. L'attributo **conn** corrisponde alla connessione del database. L'attributo **dbname** corrisponde al nome del database. L'attributo **dbuser** al nome dell'utente per potere accedere al database. L'attributo **dbhost** corrisponde al nome del host per accedere al database. Come metodo abbiamo anche il **_construct** che corrisponde al costruttore della classe. Per finire il metodo **getConnection** permette di connettersi al database PDO.

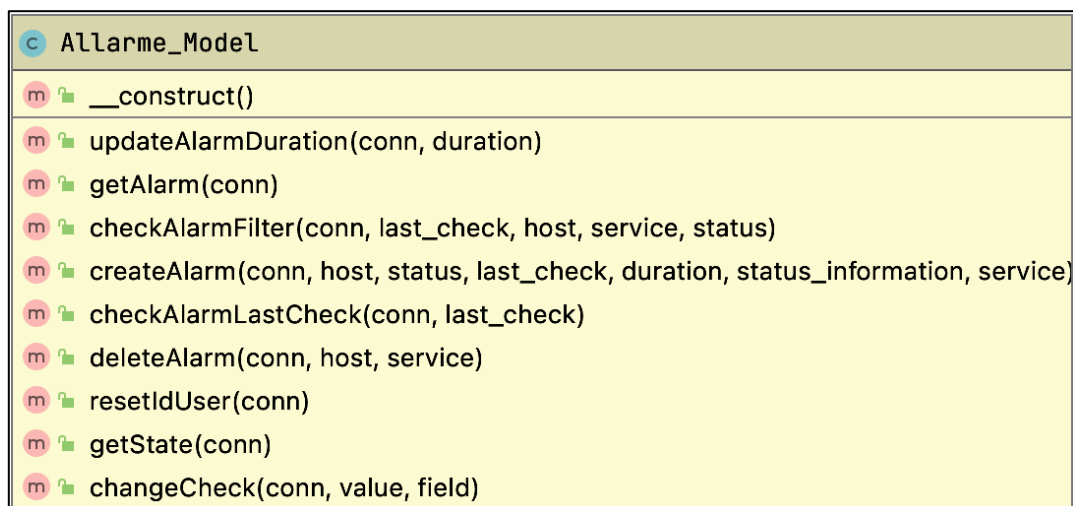


Figura 14: Model allarme

La classe **Allarme_Model** è il model che mi permette di eseguire dell'interrogazione sulla tabella allarme. Il metodo **_construct** corrisponde al costruttore della classe. Il metodo **updateAlarmDuration** permette di aggiornare il campo durata dell'allarme. Il metodo **getAlarm** permette di ricavare le informazioni dalla tabella allarme. Il metodo **checkAlarmFilter** permette di controllare se non ci sia lo stesso allarme con lo stesso campo ultimo check, servizio e stato del servizio. Il metodo **createAlarm** permette di creare un allarme da visualizzare sullo schermo esterno. Il metodo **checkAlarmLastCheck** permette di controllare che non esista un allarme con lo stesso campo ultimo check. Il metodo **deleteAlarm** permette di eliminare un allarme. Il metodo **resetIdUser** permette di effettuare il reset degli id. Il metodo **getState** permette di ricavare i vari stati dei vari check. Per finire il metodo **changeCheck** permette di cambiare lo stato di un determinato check.

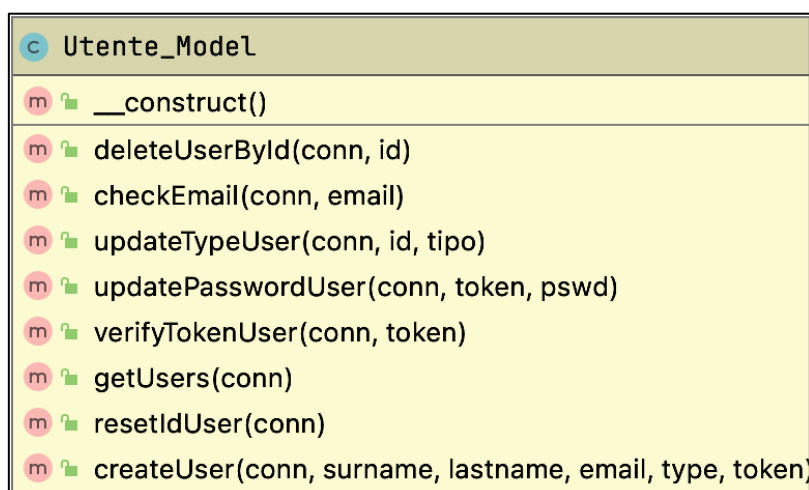


Figura 15: Model utente

La classe **Utente_Model** è il model che mi permette di eseguire delle 'interrogazioni sulla tabella **utente**. Il metodo **_construct** corrisponde al costruttore della classe. Il metodo **deleteUserById** permette di eliminare un determinato utente in base all'identificativo. Il metodo **checkEmail** permette di verificare se esiste un utente con la stessa email. Il metodo **updateTypeUser** permette di modificare i permessi degli utenti. Il metodo

updatePasswordUser permette modificare la password di un determinato utente. Il metodo **verifyTokenUser** permette di verificare se esiste un utente con il token passato come parametro. Il metodo **getUsers** permette di ricavare i dati degli utenti. Il metodo **resetIdUser** permette di effettuare il reset degli id. Per finire il metodo **createUser** permette di creare un utente.

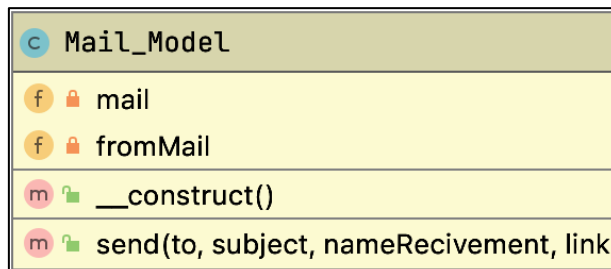


Figura 16: Model mail

La classe **Mail_Model** è il model che permette di inviare le email. L'attributo **mail** corrisponde all'oggetto PHPMailer che viene usato per creare le email. L'attributo **fromMail** corrisponde all'email mittente. Il metodo **_construct** corrisponde al costruttore della classe. Il metodo **send** permette di inviare delle email con la libreria PHPMailer.

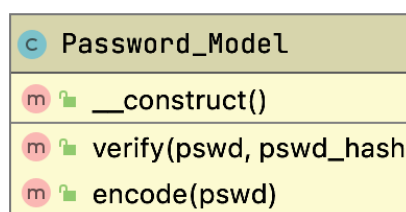


Figura 17: Model password

La classe **Password_Model** è un model che permette di gestire le password. Il metodo **_construct** corrisponde al costruttore della classe. La classe **verify** permette di verificare che la password in chiaro corrisponde alla password codificata. Per finire il metodo **encode** permette di codificare la password in chiaro.

4.4.2 UML Controllers

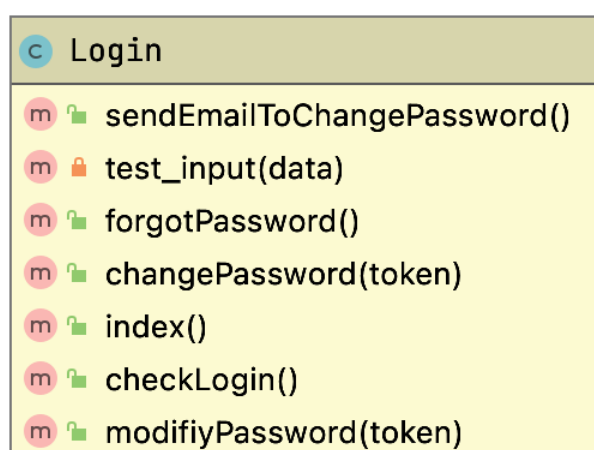


Figura 18: Controller login

La classe **Login** è il controller principale che permette di gestire la pagina di login. Il metodo **sendEmailToChangePassword** permette di inviare un email per potere cambiare la password. Il metodo **test_input** permette di controllare il dato inserito sia corretto. Il metodo **forgotPassword** permette di visualizzare la views dimentica password. Il metodo **changePassword** permette di visualizzare la pagina per potere modificare la password. Il metodo **index** permette di gestire la views login. Il metodo **checkLogin**

permette di autenticare un utente all'applicativo web. Per finire il metodo **modifyPassword** permette di modificare la password.



Figura 19: controller utente

La classe **Utente** corrisponde al controller che permette di gestire la pagina di gestione degli utenti. Il metodo **test_input** permette di controllare il dato inserito sia corretto. Il metodo **viewAddUser** permette di visualizzare la pagina dove un utente può creare un nuovo utente. Il metodo **changeType** permette di modificare i permessi degli utenti. Il metodo **deleteUserById** permette di eliminare un determinato utente. Il metodo **checkAddUsers** permette di controllare i dati inseriti nel form di creazione di un utente. Il metodo **createToken** permette di creare il token dell'utente. Il metodo **index** permette di gestire la views utente. Per finire il metodo **logout** permette di uscire dall'applicativo web.

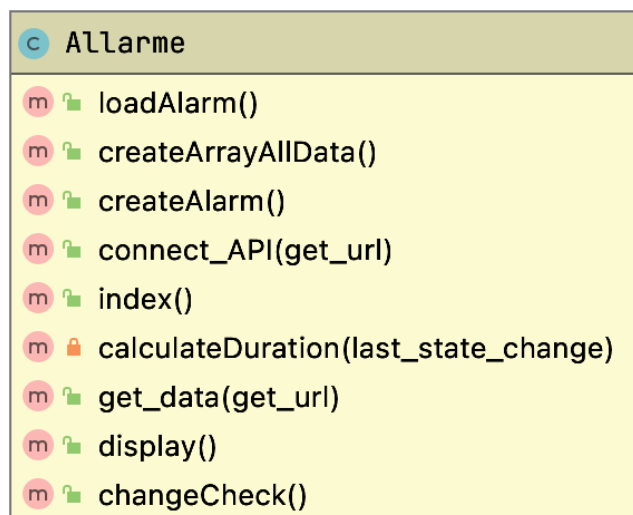


Figura 20: Controller allarme

La classe **Allarme** corrisponde al controller che permette di gestire la pagina di visualizzazione degli allarmi. Il metodo **loadAlarm** permette di ricavare i dati degli allarmi da potere stampare sullo schermo. Il metodo **createArrayAllData** permette di creare un array che contiene tutte le informazioni degli allarmi. Il metodo **createAlarm** permette di creare un allarme. Il metodo **connect_API** permette di connettersi all'API. Il metodo **index** permette di gestire la views allarme. Il metodo **calculateDuration** permette di calcolare la durata dell'host. Il metodo **get_data** permette di ricavare i dati dall'API. Il metodo **display** permette di visualizzare la pagina dove un utente può visualizzare gli allarmi. Il metodo **changeCheck** permette di modificare lo stato dei vari check.

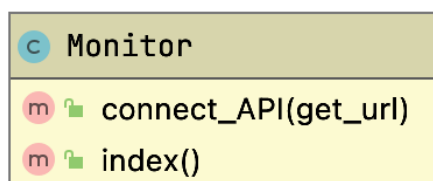


Figura 21: Controller monitor

La classe **Monitor** è un controller che permette di gestire la pagina che viene visualizzata sul monitor esterno. Il metodo **connect_API** permette di connettersi all'API. Il metodo **index** permette di gestire la views che viene è presente sul monitor esterno.

5 Implementazione

5.1 Applicativo web

5.1.1 Struttura

La struttura del progetto è la seguente:

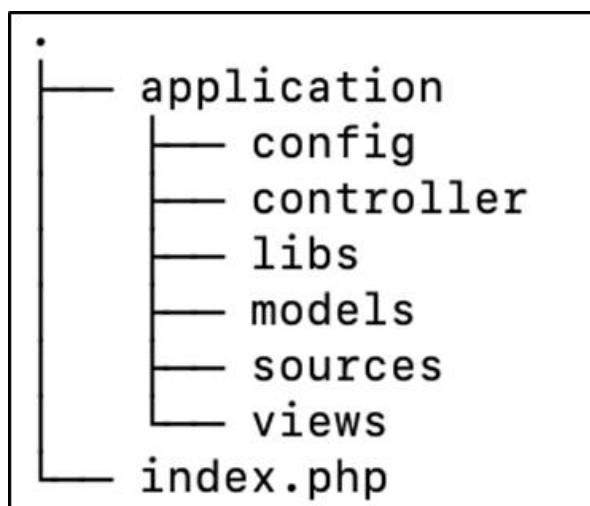


Figura 22: Struttura applicativo web

La cartella di root si chiama gestione-allarmi e al suo interno ci sono le seguenti cartelle:

- **application**: corrisponde alla cartella dell'applicazione web. Al suo interno ci sono i models, controllers e views.
- **config**: nella cartella si trova il file **config.php** che corrisponde al file di configurazione dell'applicazione web. Nel file **config.php** viene impostato l'URL del sito web.
- **libs**: nella cartella troviamo il file **application.php** che permette di costruire la struttura del sito web.
- **models**: nella cartella ci sono tutti models dell'applicazione che implementano le varie funzionalità che si possono fare nell'applicazione web. Ad esempio, accedere ad un database o creare un nuovo utente.
- **controller**: nella cartella ci sono tutti i controllers che permettono di fare interagire i models con le views.
- **sources**: nella cartella possiamo trovare tutte le librerie CSS e JavaScript. Inoltre, sono anche presenti i vari script JavaScript e i vari file CSS per lo stile delle varie pagine che ho creato personalmente.
- **views**: nella cartella troviamo tutte views dell'applicazione, ovvero tutte le pagine web dell'applicativo. Nelle views viene implementato solamente la parte grafico del sito web.

5.1.2 Model View Controller

L'applicativo web è stato sviluppato con il pattern **MVC** (Model View Controller) che permette di separare meglio il codice dalle funzionalità che posso eseguire sull'applicativo web.

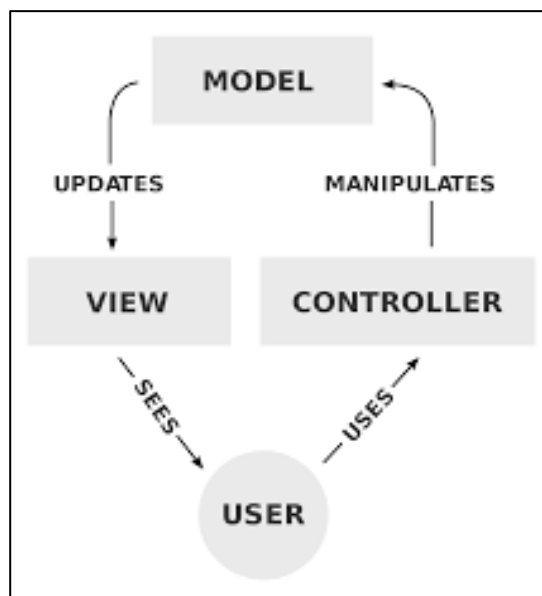


Figura 23: MVC

Il pattern **MVC** è suddiviso in tre componenti:

- **Controller** permette di ricevere i comandi dell'utente attraverso le **View** e reagiscono eseguendo delle operazioni con l'aiuto dei **Model** e che portano generalmente ad un cambiamento di stato delle **View**.
- **Model** contiene il metodo di accesso ai dati presenti nel database.
- **View** si occupa di visualizzare i dati dell'utente e gestisce l'interazione fra quest'ultimo e i **Controller**.

All'interno dell'applicativo web i **Controller** sono delle classi che si occupano di collegare cioè che sono le **Views** con i dati ricavati dalla banca dati attraverso i **Model**. I **Model** sono anche delle classi che implementano dei metodi per facilitare le interrogazioni della banca dati (come ad esempio: inserimento, aggiornamento e eliminazione). Mentre le **Views** sono semplicemente delle pagine HTML e PHP che verranno mostrate all'utente attraverso i **Controller**.

5.1.3 Configurazione del file config

Il file **config.php** contiene tutte le configurazioni per la creazione dell'applicativo web, ovvero l'URL del sito web e le costanti che contengono le informazioni per accedere al database **gestione-allarmi**. Per dichiarare tutte le costanti dell'applicazione utilizzo il costrutto **define** rispetto al costrutto **const**, perché verranno utilizzate in tutta la struttura **MVC**.

Il codice seguente permette di costruire l'URL dell'applicativo web. Nella variabile **\$actual_link** viene salvato il tipo di protocollo web e il contenuto del host (intestazione della richiesta corrente, solo se è presente). Per sapere quale tipo di protocollo web utilizza il server, viene utilizzato il metodo **isset** che permette di verificare se il server utilizza il protocollo HTTPS. Se viene utilizzato il protocollo HTTPS, viene aggiunto all'URL la stringa **https://**, altrimenti viene aggiunto la stringa **http://**. Nella variabile **\$documentRoot** viene memorizzato il nome della cartella root del progetto (**gestione-allarmi**). La variabile **\$final** corrisponde al URL completo del sito web. L'URL non deve contenere dei backslash e quindi con il metodo **str_replace** vado a rimuovere quest'ultimi con degli slash.

```
/**
 * Configurazione di : URL del progetto
 */
$actual_link = (isset($_SERVER['HTTPS']) &&
$_SERVER['HTTPS'] === 'on' ? "https" : "http") . "://$_SERVER[HTTP_HOST]";
$documentRoot = $_SERVER['DOCUMENT_ROOT'];
$dir = str_replace('\\', '/', getcwd().'/');
$final = $actual_link.str_replace($documentRoot, '', $dir);
define('URL', $final);
```

Figura 24: File di configurazione

Le seguenti costanti permettono di collegarsi al database **gestione_allarmi** nel quale ci sono le tabelle del progetto, ovvero le tabelle **utente** e **allarme**.

```
/**
 * Costanti per accedere al database gestione_allarmi
 */
define('DBNAME', 'gestione_allarmi');
define('DBUSER', 'efof_gestione_allarmi');
define('DBPASS', ' ');
define('DBHOST', 'localhost');
define('DBPORT', '3306');
```

Figura 25: Costanti per accedere al database

5.1.4 Connessione al database

Ho creato la classe **Database_Model** che contiene il metodo **getConnection** che permette di connettersi ad un database MySQL attraverso l'utilizzo dell'oggetto PDO. Il metodo **getConnection** deve essere statico, perché verrà utilizzato da tutte le classi dell'applicativo web e quindi non dovrò creare sempre una nuova istanza. I parametri della connessione vengono passati direttamente al costruttore dell'oggetto PDO. Ho anche impostato la modalità di errori di PDO in modo che vengono sollevate le eccezioni in caso di errori. Il metodo dell'oggetto di PDO **setAttribute** permette di impostare degli attributi specifici per i driver. L'attributo **PDO::ATTR_ERRMODE_EXCEPTION** memorizza tutte le eccezioni che sono state sollevate.

```
/**
 * Permette connettersi al database con la classe PDO.
 * @param dbname Nome del database, il valore di default null.
 * @return Connessione al database
 */
public static function getConnection($dbname = null, $dbuser = null, $dbpass = null){
    try{

        //modifico il nome del database solo se viene passato l'argomento dbname
        if(!is_null($dbname)){
            self::$dbname = $dbname;
        }

        //modifico il nome dell'utente solo se viene passato l'argomento dbuser
        if(!is_null($dbuser)){
            self::$dbuser = $dbuser;
        }

        //modifico la password dell'utente solo se viene passato l'argomento dbpass
        if(!is_null($dbpass)){
            self::$dbpass = $dbpass;
        }

        //permette di creare una sola connessione
        if(!self::$conn){
            $dsn = 'mysql:host=' . self::$dbhost . ';dbname=' . self::$dbname . ';port=' . self::$dbport;
            self::$conn = new PDO($dsn, self::$dbuser, self::$dbpass);
            self::$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        }

        return self::$conn;
    }
    catch (PDOException $e){
        //messaggio di errore di connessione
        echo $e->getMessage();
    }
}
```

Figura 26: Metodo per connettersi al database

Il metodo può anche ricevere come parametri il nome del database, lo username e la password per accedere al database. Se quest'ultimi non vengono passati verranno utilizzate i valori dei default per accedere al database **gestione_allarmi**. Inoltre, ho creato la variabile **\$conn** che permette di memorizzare una vecchia connessione al database in modo da creare una sola connessione che potrà essere utilizzata in tutte le classi dell'applicativo web.

5.1.5 Sicurezza

5.1.5.1 Prepared statement

Quando eseguo delle query con PDO utilizzo le prepared statement che mi permettono di proteggere dalle SQL injection, perché i valori dei parametri delle query vengono trasmessi successivamente utilizzando un protocollo diverso e non devono essere correttamente sottoposti a escape. Se il modello di istruzione originale non è derivato da un input esterno, non è possibile eseguire SQL injection.

Il seguente codice permette di inserire un nuovo utente. Il metodo **prepared** permette di preparare la query che verrà eseguita solamente una sola volta (sebbene l'istruzione venga eseguita più volte). Nella query i parametri vengono inseriti con il metodo **bindParam**. Per finire il metodo **execute** permette di eseguire la prepared statement.

```
/**
 * Permette di creare un nuovo utente.
 * @param conn Connessione al database.
 * @param surname Nome dell'utente.
 * @param lastname Cognome dell'utente.
 * @param email Email dell'utente.
 * @param type Tipo di utente.
 */
public function createUser($conn, $surname, $lastname, $email, $type, $token){
    $sth = $conn->prepare('insert into utente (email, nome, cognome, tipo, token)
    values(:email, :surname, :lastname, :type, :token)');
    $sth->bindParam(':email', $email, PDO::PARAM_STR);
    $sth->bindParam(':surname', $surname, PDO::PARAM_STR);
    $sth->bindParam(':lastname', $lastname, PDO::PARAM_STR);
    $sth->bindParam(':type', $type, PDO::PARAM_STR);
    $sth->bindParam(':token', $token, PDO::PARAM_STR);
    $sth->execute();
}
```

Figura 27: Esempio di prepared statement

5.1.5.2 Controllo dei valori degli input

Non bisogna mai fidarsi dei controlli JavaScript e HTML e quindi bisogna sempre eseguire dei controlli di sicurezza quando si trattano dei form. Il metodo **test_input** permette di controllare i valori che vengono inseriti nei campi input del form. La funzione **htmlspecialchars** permette di convertire i caratteri speciali di una stringa trasformandoli nei corrispondenti codici HTML. I caratteri speciali possono essere <, >, ", & e potrebbero causare problemi durante l'esecuzione dello script, soprattutto se la stringa viene salvata su un database SQL. Quest'ultimi vengono convertiti nei codici HTML <, >, ", &.

La funzione **stripslashes** permette di restituire una stringa con i backslash rimossi. I doppi backslash (\\) vengono trasformati in un singolo backslash (\). La funzione **trim** permette di rimuovere i caratteri non necessari (spazio extra, tabulazione, nuova riga). Questo metodo permette di proteggere contro attacchi di tipo Cross Site Scripting (XSS). Gli attacchi XSS sono un tipo di iniezione, in cui script dannosi vengono iniettati in siti Web.


```
/**
 * Permette di controllare il dato inserito sia corretto
 * @param data Valore inserito nel campo.
 */
private function test_input($data) {
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    return $data;
}
```

Figura 28: Metodo per controllare i valori degli input

5.1.5.3 preg_match e filter_var

Oltre a controllare i valori inseriti negli input, con le **preg_match** verifico che il valore inserito corrisponde ad un pattern (espressione regolare). Le espressioni regolari sono utili per ricercare, sostituire o suddividere una stringa in più sottostringhe tramite un criterio logico. Alla funzione **preg_match** si deve passare due parametri. Il primo parametro è il pattern che corrisponde al modello da cercare come stringa. Invece il secondo parametro è la stringa di input da verificare.

Ad esempio, la seguente espressione regolare imposta una regola che bisogna rispettare per definire il nome dell'utente. Il nome dell'utente deve essere sempre composto da solo lettere dell'alfabeto. La prima lettera deve essere maiuscola.

```
//espressione regolare per il campo nome e cognome
$pattern = "/^[A-Z]+[a-z]{0,}/";
```

Figura 29: Pattern nome dell'utente

La funzione **filter_var** permette di filtrare una variabile con dei filtri specifici. Utilizzo questa funzione per verificare che un'e-mail corrisponde effettivamente ad un indirizzo di posta elettronica.

```
filter_var($_POST["email"], FILTER_VALIDATE_EMAIL)
```

Figura 30: Controllo dell'email

5.1.5.4 Password codificate

Per aumentare il livello di sicurezza dell'applicazione web evito di memorizzare le password in chiaro nel database. In effetti nel database memorizzo la hash della password in modo che si possa verificare che la password inserita dall'utente sia corretta. Ho creato la classe **Password** che implementa due metodi. Il primo metodo si chiama **encode** e permette di codificare la password in hash.

```
/**
 * Permette di codificare la password in hash.
 * @param pswd Password in chiaro.
 */
public function encode($pswd){
    $hash = password_hash($pswd, PASSWORD_DEFAULT);
    return $hash;
}
```

Figura 31: Metodo per codificare le password

La funzione **password_hash** permette di creare un hash di una password. La funzione riceve due parametri che sono la password in chiaro e il tipo di algoritmo da utilizzare come codifica. Come algoritmo utilizzo la costante **PASSWORD_DEFAULT** che corrisponde all'algoritmo **bcrypt**. Quest'ultimo oltre ad espandersi a 60 caratteri, incorpora una salt che permette di proteggere meglio la password contro attacchi.

Invece il secondo metodo si chiama **password_verify** e permette di verificare che la password passata corrisponde al hash.

```
/**
 * Permette di verificare che la password
 * passata corrisponde al hash.
 * @param pswd Password in chiaro
 * @param pswd_hash Password codificata.
 */
public function verify($pswd, $pswd_hash){
    return password_verify($pswd, $pswd_hash);
}
```

Figura 32: Metodo per verificare l'hash con la password in chiaro

La funzione **verify** permette di verificare che una password in chiaro corrisponde ad un hash. In effetti la funzione riceve due parametri che sono la password in chiaro e l'hash da confrontare.

5.1.5.5 Token

Il token rappresenta un codice che verrà utilizzato per eseguire il recupero della password. Questo codice viene generato in modo casuale e codificato in SHA256. Nella classe **utente** ho creato il metodo **createToken** che permette di generare il token. Questo metodo viene richiamato ogni volta che viene creato un nuovo utente.

```
/**
 * Permette di creare il token
 */
private function createToken(){
    $token = bin2hex(random_bytes(20));
    return hash("sha256", $token);
}
```

Figura 33: Metodo per generare il token

Per potere cambiare la password bisogna accedere al metodo **changePassword** del controller **login**. Il metodo riceve come parametro anche il token, in modo che si può verificare che quest'ultimo appartiene ad un utente. In effetti il token viene inviato per email all'utente insieme al link.

5.1.6 Sistema di email

Per inviare l'email quando vengono creati dei nuovi utenti o per potere modificare la password, ho creato una classe che si occupa di inviare la posta elettronica. Per realizzare questo sistema la libreria PHPMailer.

Il metodo **send** si occupa di inviare le email e quest'ultimo accetta i seguenti parametri:

- **to:** indirizzo email del destinatario.
- **subject:** oggetto dell'email.
- **nameRecivement:** nome del destinatario (nome e cognome dell'utente).
- **link:** link da inviare all'utente per potere accedere alla pagina di modifica password.

```
/**
 * Permette di inviare le email.
 * @param to Indirizzo email del destinatario.
 * @param subject Oggetto dell'email.
 * @param nameRecivement Nome del destinatario (nome e cognome).
 * @param link Link da inviare all'utente
 */
public function send($to, $subject, $nameRecivement, $link){
    try{
        //imposto l'indirizzo email e nome mittente
        $this->mail->From = $this->fromMail;
        $this->mail->FromName = "Gestione allarmi";

        //imposto l'email destinatario
        $this->mail->addAddress($to, $nameRecivement);

        //imposto il soggetto dell'email
        $this->mail->Subject = $subject;

        //charset UTF-8
        $this->mail->CharSet = 'UTF-8';

        $this->mail->isHTML(true);
    }
}
```

Figura 34: Metodo send che permette di inviare email

Gli attributi **From** e **FromName** permettono di impostare il nome e l'indirizzo di posta elettronica del mittente. Come e-mail mittente ho creato la seguente e-mail noreply: system@gestione-allarmi.ch. Per il sistema utilizzo questo tipo di e-mail, perché non ci deve essere la possibilità di rispondere al mittente, ma l'e-mail deve essere solamente inviata senza ritorno.

L'attributo **CharSet** permette di impostare la visualizzazione corretta per una pagina HTML. In effetti un browser Web deve sapere quale set di caratteri deve utilizzare. Con la libreria **PhpMailer**, il contenuto del corpo dell'e-mail può anche essere del codice HTML. Il metodo **addAddress** permette di impostare l'indirizzo e l'e-mail del destinatario e anche il nome del destinatario. Con il metodo **isHTML** posso attivare la modalità che mi permette di utilizzare del codice HTML come messaggio da inviare. L'attributo **Subject** corrisponde all'oggetto dell'e-mail.

```
//corpo del messaggio
$this->mail->Body = $message;

//invio l'email
$this->mail->send();

}
catch (Exception $e)
{
    /* PHPMailer exception. */
    echo $e->errorMessage();
}
catch (\Exception $e)
{
    /* PHP exception (nota la barra rovesciata per selezionare
    la classe di eccezione dello spazio dei nomi globale). */
    echo $e->getMessage();
}
}
```

Figura 35: Metodo send che permette di inviare email

L'attributo **Body** corrisponde al corpo dell'e-mail e utilizzando il metodo **send** della libreria posso inviare l'e-mail.

Non ho impostato nella classe la porta TCP per connettersi al mail server, perché nella libreria **PhpMailer** di default l'attributo **Port** è impostato su 25 (SMTP) e utilizzerò quest'ultima come porta.

5.1.7 Popup

Per realizzare i popup ho utilizzato il plug-in Modal che corrisponde ad una finestra di dialogo/finestra popup che viene visualizzata dove si vuole nella pagina corrente. Per realizzare un Modal ho utilizzato la libreria bootstrap (<https://getbootstrap.com/docs/5.0/components/modal/>).

Nel codice HTML ho aggiunto gli attributi **data-bs-toggle** e **data-bs-target** per ogni pulsante "Elimina". Il primo attributo permette di agganciare al pulsante la classe bootstrap modal. Invece il secondo attributo permette di definire quale finestra di dialogo deve essere aperta quando viene cliccato il pulsante. Ho anche aggiunto l'attributo **data-url** che viene utilizzato nello script JavaScript e permette di definire il metodo PHP da richiamare dal modal.

```

/* quando l'utente clicca sul pulsante "Elimina" compare il popup di conferma */
deleteModal.addEventListener('show.bs.modal', function (event) {
  /* quando l'utente clicca sul pulsante "Elimina" */
  var button = event.relatedTarget;

  /* setto il messaggio del popup */
  description.innerHTML = button.getAttribute('data-information');

  /* ottengo l'URL */
  var url = button.getAttribute('data-url');

  /* l'url deve essere definito nell'attributo data-url del pulsante "Elimina" */
  if (typeof url !== 'undefined') {
    /* Aggiungo la proprietà href al pulsante "Sì, conferma". */
    /* Quando verrà premuto il pulsante verrà richiamato */
    /* il metodo deleteInformazione del controller Home. */
    $("#confirm").attr("href", url);
  }
});

```

Figura 36: Codice JavaScript popup

La variabile **deleteModal** corrisponde al modal che verrà visualizzato quando l'utente clicca il pulsante **Elimina**. La variabile **description** corrisponde al paragrafo nel quale viene scritto il messaggio del popup. La variabile **confirm** corrisponde al pulsante "Elimina" del popup. La funzione **addEventListener** permette di eseguire una funzione quando è visibile il popup. La funzione **relatedTarget** corrisponde all'evento del mouse. Sul pulsante "Elimina" del popup viene aggiunto la proprietà **href** con il metodo JavaScript **attr**. Quando un utente clicca il pulsante di conferma viene richiamato un metodo PHP che permette di eliminare un utente.

5.1.8 Sistema di visualizzazione sul monitor esterno

Per la visualizzazione degli allarmi e della mappa di rete ho utilizzato in JavaScript il metodo AJAX che permette di scambiare dei dati in background fra il web browser e il server senza per forza aggiornare la pagina web.

Il seguente codice mostra come costruire un metodo AJAX con i suoi opportuni parametri.

```
$.ajax({
  url: url,
  type: 'POST',
  data:{status: "critical"},
  success: function(e){
    count = 0;
    $.each(JSON.parse(e), function(index, value){
      //riempio l'array che mantiene lo stato dei check.
      stateCheck[count] = (value==1?true:false);
      ++count;
    });
  }
});
```

Figura 37: Codice AJAX di esempio

Come linguaggio di programmazione non utilizzo JavaScript, ma utilizzo la sintassi di JQuery. Quest'ultima è una libreria di JavaScript molto veloce, piccola e ricca di funzionalità.

Nel parametro **type** bisogna impostare il tipo di richiesta HTTP, ovvero se viene utilizzato il POST o il GET. Nel parametro **url** bisogna impostare l'url, ovvero il metodo PHP che AJAX deve richiamare. Il parametro **success** è una funzione che viene richiamato se la richiesta ha esito positivo. Il parametro **e** corrisponde al risultato della richiesta.

5.1.9 API Nagios

Nel file configurazione dell'applicativo ho creato delle costanti che corrispondono alle credenziali per potere accedere all'API Nagios.

```
/**
 * Costanti API Nagios
 */
define('NAGIOS_URL', 'http://monitor.cpt.local/nagios/');
define('NAGIOS_USER', 'reporter');
define('NAGIOS_PASS', ' ');
define('NAGIOS_URL_REFRESH', '5');
define('NAGIOS_TIME', 'Europe/Zurich');
```

Figura 38: Costanti per potere accedere all'API Nagios

La prima costante corrisponde all'URL per potere accedere all'API. La seconda e la terza costanti corrispondono alle credenziali (username e password) per potere autenticare all'API. Le ultime costanti sono opzionali. In effetti la quarta costante permette di ricaricare l'URL in un intervallo di tempo e la quinta costante corrisponde alla località in cui ci troviamo.

Nella controller ho creato il metodo **get_data** che permette di ottenere i dati dal modulo Nagios tramite il file **statusjson.cgi**.

```
/**
 * Permette ottenere i dati dal modulo Nagios tramite l'API JSON.
 */
public function get_data($get_url){
    //connessione all'API NAGIOS
    $get_status = curl_init(NAGIOS_URL . $get_url);
    curl_setopt($get_status, CURLOPT_RETURNTRANSFER, true);
    //imposto la password e lo username
    curl_setopt($get_status, CURLOPT_USERPWD, NAGIOS_USER . ":" . NAGIOS_PASS);
    //eseguo il curl per ricavare le risorse
    $res = curl_exec($get_status);
    //chiudo la connessione
    curl_close($get_status);
    return json_decode($res, true);
}
```

Figura 39: Metodo che permette di ottenere i dati da Nagios

Il metodo **curl_init** permette di inizializzare una nuova sessione e restituisce un handle CURL da utilizzare con le funzioni **curl_setopt**, **curl_exec** e **curl_close**. Il metodo **curl_setopt** permette di impostare sull'handle di sessione CURL specificato. Come opzione utilizzo la costante **CURLOPT_RETURNTRANSFER** in modo che restituisce il trasferimento come una stringa di ritorno di **curl_exec** invece di inviarlo direttamente. Inoltre, utilizzo anche la costante **CURLOPT_USERPWD** che permette di impostare nell'handle le credenziali di autenticazione. IL metodo **curl_exec** permette di eseguire il curl per potere ricavare i dati dall'API. Per potere chiudere la connessione curl bisogna utilizzare il metodo **curl_close**.

5.1.10 Raspberry

Nel progetto ho utilizzato un raspberry che mi permetteva di interfacciare lo schema con il collegamento di rete. Quest'ultimo doveva essere raggiungibile via SSH ed avere un indirizzo IPv4 statico (10.20.4.50). Il raspberry è collegato con un cavo adattatore micro-HDMI al HDMI del televisore che si trova in segreteria. Quando il raspberry si accende viene eseguito il file di avvio sul desktop LXDE. LXDE è un ambiente desktop estremamente veloce, performante e con un buon risparmio energetico. Nel file di avvio viene eseguito il comando **unclutter** per potere rimuovere dallo schermo il cursore del mouse e per finire viene eseguito il comando **chromium-browser** che permette di aprire il browser chromium in modalità kiosk, ovvero l'applicazione verrà avviata automaticamente a schermo intero. Il raspberry rappresenta anche il web server locale, dove un utente può accedere all'applicativo web da qualsiasi computer che appartiene alla rete della CPT. Inoltre, ho preparato un manuale dettagliato per la sua implementazione.

5.1.11 Interfacce delle pagine

5.1.11.1 Pagina di login

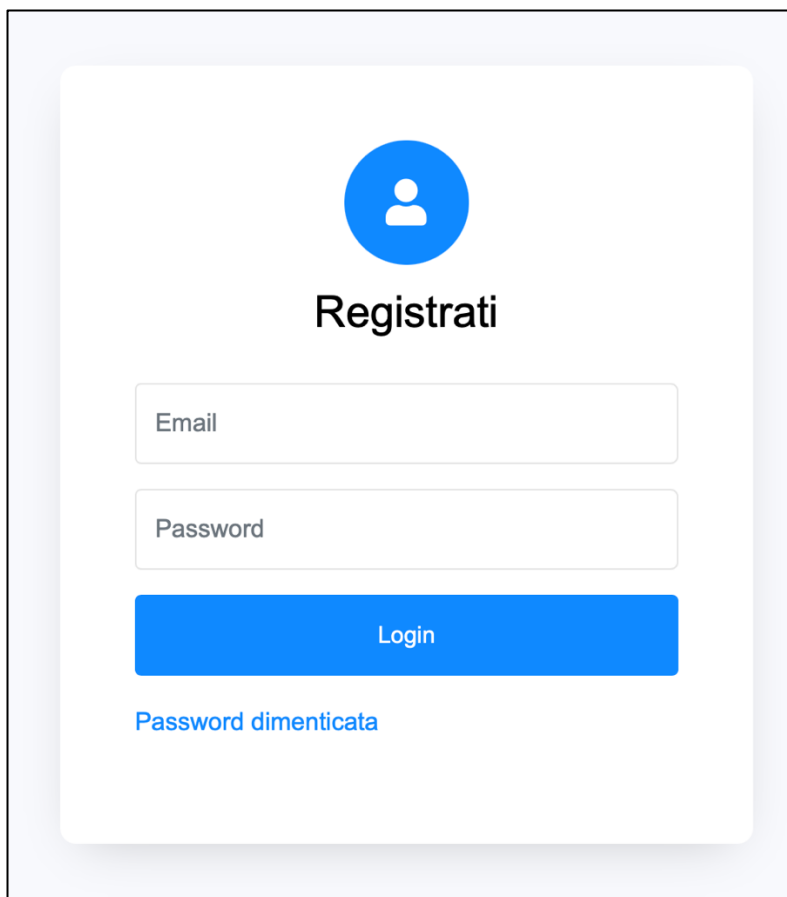
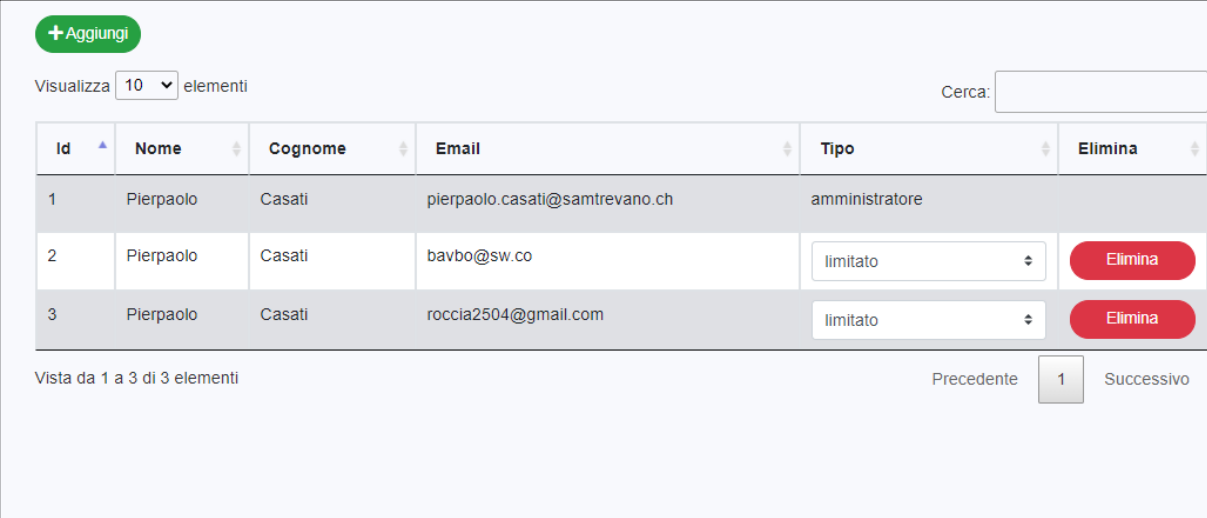


Figura 40: Pagina di login

Questa è la pagina principale che viene mostrata quando un utente accede al sito web. In effetti corrisponde al login dove l'utente deve accedere con l'e-mail e la password del proprio account. Nel form è presente un link "Password dimenticata" che permette di accedere ad una pagina web dove ha la possibilità di modificare la vecchia password.

5.1.11.2 Pagina di gestione utenti



| Id | Nome | Cognome | Email | Tipo | Elimina |
|----|-----------|---------|--------------------------------|----------------|---------|
| 1 | Pierpaolo | Casati | pierpaolo.casati@samtrevano.ch | amministratore | |
| 2 | Pierpaolo | Casati | bavbo@sw.co | limitato | Elimina |
| 3 | Pierpaolo | Casati | roccia2504@gmail.com | limitato | Elimina |

Figura 41: Pagina gestione utenti

Questa è la pagina dove un utente può visualizzare in una tabella tutte le informazioni degli utenti. In alto a sinistra l'utente amministratore può cliccare il pulsante "+ Aggiungi" per creare un nuovo utente. Sotto quest'ultimo c'è un menu a tendina che permette di scegliere il numero di elementi che si vogliono visualizzare nella tabella. Su ogni colonna della tabella sono presenti delle frecce che permettono di ordinare la tabella. Invece in alto a destra è presente un campo di ricerca che permette di cercare un'informazione in base ad una parola chiave. Per potere modificare i permessi di ogni utente ho creato un menu a tendina sotto il campo tipo. Quando l'utente clicca sul pulsante "Elimina" compare un popup per confermare l'eliminazione dell'utente. La tabella è suddivisa in viste sulle quali ci possono essere almeno dieci elementi e per spostarsi da una vista all'altra l'utente ha la possibilità di utilizzare il controllo di paginazione che si trova sotto la tabella.

5.1.11.3 Pagina di creazione utenti

Figura 42: Pagina di creazione utente

Questa è la pagina dove un utente amministratore ha la possibilità di creare un nuovo utente tramite un form. Nel form sono presenti i diversi campi che permettono di impostare il nome, il cognome e l'email dell'utente.

5.1.11.4 Pagina di amministrazione

| <input checked="" type="checkbox"/> Host <input checked="" type="checkbox"/> Stato <input checked="" type="checkbox"/> Ultimo check <input checked="" type="checkbox"/> Durata <input checked="" type="checkbox"/> Informazione stato <input checked="" type="checkbox"/> Mappa di rete | | | | | |
|---|-----------------|-----------------------------|---------------------|--------------------|---|
| Visualizza 10 elementi | | Cerca: <input type="text"/> | | | |
| Host | Servizio | Stato | Ultimo check | Durata | Informazione stato |
| altair | Current Load | Ok | 26.05.2021 12:09:43 | 4 d 16 h 19 m 20 s | OK - load average: 0.00, 0.04, 0.01 |
| altair | Current Users | Ok | 26.05.2021 12:09:36 | 4 d 16 h 19 m 20 s | USERS OK - 0 users currently logged in |
| altair | HTTP | Ok | 26.05.2021 12:09:38 | 4 d 16 h 19 m 20 s | HTTP OK: HTTP/1.1 200 OK - 264 bytes in 0.001 second response time |
| altair | Root Partition | Ok | 26.05.2021 12:09:38 | 4 d 16 h 19 m 20 s | DISK OK - free space: / 24292 MIB (88.01% inode=94%): |
| altair | SSH | Ok | 26.05.2021 12:09:39 | 4 d 16 h 19 m 20 s | SSH OK - OpenSSH_8.0 (protocol 2.0) |
| altair | Swap Usage | Ok | 26.05.2021 12:09:39 | 4 d 16 h 19 m 20 s | SWAP OK - 100% free (1018 MB out of 1023 MB) |
| altair | Total Processes | Ok | 26.05.2021 12:09:36 | 4 d 16 h 19 m 20 s | PROCS OK: 99 processes with STATE = RSZDT |
| ariel.nunnari.ch | HTTP | Ok | 26.05.2021 12:07:28 | 7 d 3 h 25 m 28 s | HTTP OK: HTTP/1.1 301 Moved Permanently - 534 bytes in 0.080 second response time |

Figura 43: Pagina gestione amministrazione

Questa è la pagina dove un utente può visualizzare in una tabella tutte le informazioni delle apparecchiature di rete della CPT che sono monitorate da Nagios. In alto a sinistra c'è un menu a tendina che permette di scegliere il numero di elementi che si vogliono visualizzare nella tabella. Su ogni colonna della tabella sono presenti delle frecce che permettono di ordinare la tabella. Invece in alto a destra è presente un campo di ricerca che permette di cercare un'informazione in base ad una parola chiave. La tabella è suddivisa in viste sulle quali ci possono essere almeno dieci elementi e per spostarsi da una vista all'altra l'utente ha la possibilità di utilizzare il controllo di paginazione che si trova sotto la tabella. Per finire in alto alla pagina sono presenti i vari check che permettono di attivare o disattivare i campi da visualizzare sul monitor.

5.1.11.5 Pagina di visualizzazione

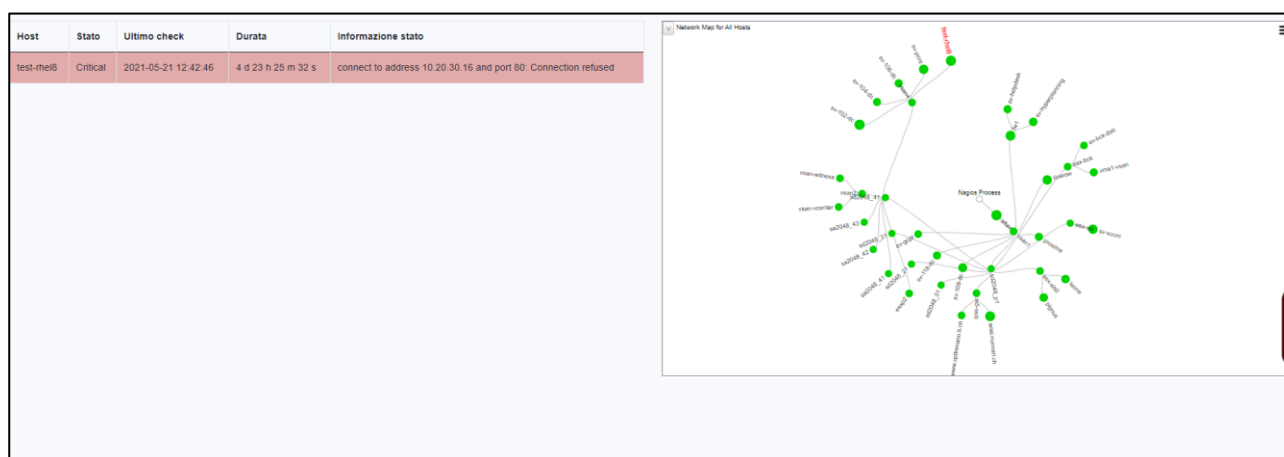


Figura 44: Pagina di visualizzazione

Questa è la pagina che verrà visualizzata sul desktop di un computer, ma che sarà anche uguale per quanto riguarda il monitor esterno. Sul monitor esterno però non è presente la barra di navigazione e il piè di pagina. Sulla pagina viene stampato sulla sinistra la tabella degli allarmi che viene aggiornata automaticamente. Invece sulla destra viene stampata una mappa di rete. Se l'utente disattiva il check per la visualizzazione della mappa di rete, la tabella degli allarmi occupa tutto lo spazio. Invece se non ci sono degli allarmi la mappa di rete se è attiva occupa tutto lo schermo.

6 Test

6.1 Protocollo di test

| | | | |
|--------------------------|---|--------------|---|
| Test Case: | TC-001 | Nome: | Compatibilità dell'applicativo sui browser più utilizzati |
| Riferimento: | REQ-001 | | |
| Descrizione: | L'applicativo deve essere compatibili sui diversi browser più utilizzati. Come ad esempio Firefox, Google Chrome e Safari. | | |
| Prerequisiti: | Installare diversi browser per potere testare il seguente test. | | |
| Procedura: | <ol style="list-style-type: none"> 1. Aprire un browser qualsiasi e scrivere nella barra di ricerca il seguente link: https://raspberrypi/gestione-allarme/. 2. Navigare all'interno dell'applicativo e verificare che le varie funzionalità. 3. Verificare che siano state scaricate in maniera corretta le varie librerie che permettono di far funzionare l'applicativo. Ad esempio, la libreria bootstrap (CSS del sito), la libreria fontawesome (icone), ecc. 4. Replicare gli stessi procedimenti sugli altri browser. | | |
| Risultati attesi: | L'applicativo deve essere compatibile con tutti i browser più utilizzati. | | |

| | | | |
|--------------------------|--|--------------|----------------------------|
| Test Case: | TC-002 | Nome: | Struttura del sito web MVC |
| Riferimento: | REQ-001 | | |
| Descrizione: | L'applicativo è implementato con il pattern MVC del linguaggio web PHP. Il pattern MVC (Model View Controller) permette di suddividere la parte grafica (View) dal codice con le sue funzioni (Model). | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Aprire l'applicativo web. 2. Verificare che la navigazione tra ogni pagina sia realizzata dal proprio controller. 3. Verificare che le varie funzionalità dell'applicativo siano gestite dai vari metodi dell'applicativo. 4. Controllare il codice dell'applicativo e verificare che sia strutturata rispetto il pattern MVC. | | |
| Risultati attesi: | L'applicativo deve essere implementato con il pattern MVC. La navigazione tra ogni views deve essere gestita dai controllers e le varie funzionalità dai vari metodi di quest'ultimi. | | |

| | | | |
|----------------------|---|--------------|--------------------|
| Test Case: | TC-003 | Nome: | Template bootstrap |
| Riferimento: | REQ-001 | | |
| Descrizione: | La grafica del sito web è gestita dalla libreria CSS bootstrap. | | |
| Prerequisiti: | Scaricare la libreria bootstrap dal seguente link: https://getbootstrap.com . | | |
| Procedura: | <ol style="list-style-type: none"> 1. Controllare che la libreria bootstrap sia importata nella maniera corretta. Per importare la libreria bootstrap scrivere il seguente codice. <pre><!-- Libreria bootstrap --> <link rel="stylesheet" href="<?php echo URL; ?> application/sources/bootstrap/css/bootstrap.min.css" rel="stylesheet"></pre> | | |

| | |
|--------------------------|---|
| | <p>2. Controllare che nel codice HTML vengono utilizzate le classi di bootstrap. Ad esempio, per avere un pulsante di colore rosso bisogna utilizzare la classe btn per creare un pulsante e btn-danger per impostare il colore di sfondo rosso del pulsante.</p> <p>3. Verificare che tutte le classi bootstrap impostate ai vari elementi HTML dell'applicativo WEB funzionano correttamente.</p> |
| Risultati attesi: | <p>La libreria bootstrap deve funzionare nel modo corretto. Ad esempio, il pulsante che permette di eliminare un utente è stato realizzato con le classi btn (pulsante) e btn-danger (colore di sfondo rosso del pulsante).</p> <div style="border: 1px solid black; padding: 5px; display: inline-block; background-color: red; color: white;">elimina</div> |

| | | | |
|--------------------------|--|--------------|---------------------|
| Test Case: | TC-004 | Nome: | Sito web responsive |
| Riferimento: | REQ-001 | | |
| Descrizione: | La pagina web deve adattarsi ai diversi dispositivi di visualizzazione (computer, portatile, tablet, smartphone ...) e adattarsi per un'esperienza utilizzatore ottimale. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Aprire l'applicazione web su un browser (Firefox). 2. Digitare il comando F12 per potere aprire gli strumenti per gli sviluppatori di siti web. 3. Selezionare la modalità visualizzazione flessibile che permette di visualizzare il sito web in diverse visualizzazioni rispetto al dispositivo scelto. 4. Controllare il responsive in tutte le pagine del sito web. 5. Provare anche fisicamente sui diversi dispositivi. | | |
| Risultati attesi: | L'applicativo deve essere implementato con il pattern MVC. La navigazione tra ogni views deve essere gestita dai controllers e le varie funzionalità dai vari metodi di quest'ultimi. | | |

| | | | |
|----------------------|---|--------------|------------------|
| Test Case: | TC-005 | Nome: | Creazione utenti |
| Riferimento: | REQ-002 | | |
| Descrizione: | Nella pagina Gestione utenti è presente un'icona +Aggiungi che permette di aprire una pagina dove è presente un form per la creazione di un utente. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Aprire la pagina Gestione utenti. 2. Cliccare l'icona. + Aggiungi 3. Assicurare che venga aperta una nuova pagina dove è presente un form. 4. Non riempire i campi del form e cliccare sul pulsante "Crea". 5. Assicurare che vengono visualizzati i tooltips su ogni campo del form. 6. Inserire dei valori sbagliati nei campi e assicurare che non appaiono dei messaggi di errore. 7. Inserire un'e-mail di un utente che esiste già nell'applicativo e verificare che compare un messaggio di errore, perché non si può creare un utente con la stessa e-mail. 8. Inserire nei campi dei valori corretti e assicurare che viene ricaricata la pagina Gestione utenti. 9. Assicurare che nella tabella utenti venga aggiornata. | | |

| | |
|--------------------------|---|
| Risultati attesi: | Se l'utente clicca sull'icona + Aggiungi , compare una pagina dove c'è un form. Nella seguente pagina l'utente può creare dei nuovi utenti. Se l'utente inserisce dei dati corretti, viene creato un nuovo utente. |
|--------------------------|---|

| | | | |
|--------------------------|---|--------------|----------------------------------|
| Test Case: | TC-006 | Nome: | Cambiare i permessi degli utenti |
| Riferimento: | REQ-002 | | |
| Descrizione: | Nella pagina Gestione utenti per ogni utente della tabella abbiamo la colonna tipo nella quale si può cambiare i permessi degli utenti tramite un menu a tendina. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> Andare nella pagina Gestione utenti Cambiare i permessi di un utente amministratore tramite il menu a tendina. Eseguire un logout Rientrare nell'applicativo e assicurare che un utente limitato non possa vedere la pagina di amministrazione e che non possa creare degli utenti. | | |
| Risultati attesi: | Un utente amministratore ha tutti i permessi su ogni pagina dell'applicativo web. L'utente limitato non può gestire gli utenti e gli allarmi, ma può solamente visualizzare gli allarmi. | | |

| | | | |
|--------------------------|---|--------------|----------------------|
| Test Case: | TC-006 | Nome: | Eliminare gli utenti |
| Riferimento: | REQ-002 | | |
| Descrizione: | Nella pagina Gestione utenti per ogni utente della tabella abbiamo la colonna elimina nella quale è presente un pulsante che permette di eliminare l'utente | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> Andare nella pagina Gestione utenti Assicurare che è presente nella tabella la colonna tipo, dove al suo interno c'è un pulsante che permette di eliminare un utente. Assicurare che quando viene cliccato l'icona appare un popup di conferma. Assicurare che se si clicca sulla icona "x" viene chiuso il popup Assicurare che si clicca sul pulsante "Chiudi" non viene eliminato l'utente. Assicurare che si clicca sul pulsante "Elimina" viene eliminato l'utente. Assicurare che l'utente venga eliminato e non più nella tabella degli utenti. | | |
| Risultati attesi: | Cliccando sul pulsante elimina si può eliminare un utente. Per eliminare effettivamente quest'ultimo deve cliccare sul pulsante "Elimina" del popup | | |

| | | | |
|----------------------|---|--------------|---------------------------------------|
| Test Case: | TC-007 | Nome: | Sistema recupera password dimenticata |
| Riferimento: | REQ-003 | | |
| Descrizione: | Se l'utente non si ricorda la password per entrare nel suo account, può recuperare la sua password cliccando il link "Password dimenticata" che è presente nel login. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> Andare nel login dell'applicativo. Cliccare il link "Password dimenticata". Assicurare che compare un form dove bisogna inserire la propria e-mail. | | |

| | |
|--------------------------|---|
| | <ol style="list-style-type: none"> Inserire un'e-mail sbagliata e cliccare sul pulsante "Richiedi una nuova password". Assicurare che si è nella pagina di login e che è apparso un messaggio di errore. Cliccare il link "Password dimenticata". Inserire la propria e-mail e cliccare sul pulsante "Richiedi una nuova password". Vedere se un'e-mail è arrivata e cliccare il link che permette di cambiare subito la password. Verificare che compaiono degli errori se non si compilano i campi che permettono di modificare la password. Cambiare la password e assicurare che si ritorna nella pagina di login. Eseguire un login per verificare che la nuova password funziona. |
| Risultati attesi: | Quando l'utente clicca sul link "Password dimenticata" presente nel login, quest'ultimo ha la possibilità di modificare la vecchia password. |

| | | | |
|--------------------------|--|--------------|-------------------------------|
| Test Case: | TC-008 | Nome: | Funzionamento pagina di login |
| Riferimento: | REQ-003 | | |
| Descrizione: | Nella pagina di login un utente può accedere al proprio account o può cliccare sul link "Password dimenticata" per potere modificare la vecchia password. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> Inserire dei valori sbagliati nel login. Assicurare che viene ricaricata la pagina di login e che compaiono degli errori. Eseguire il login con le credenziali corrette. | | |
| Risultati attesi: | L'utente riesce ad eseguire il login con l'e-mail e password. | | |

| | | | |
|--------------------------|--|--------------|---|
| Test Case: | TC-009 | Nome: | Funzionamento pagina di amministrazione |
| Riferimento: | REQ-004 | | |
| Descrizione: | La pagina di amministrazione è accessibile solamente da un utente amministratore. Nella pagina sono presente dei check che permettono di attivare o disattivare i campi da rendere visibili sullo schermo. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> Entrare nell'applicativo con delle credenziali di un utente limitato Assicurare che non si possa accedere alla pagina di amministrazione. Entrare nell'applicativo con delle credenziali utente amministratore Disattivare o attivare i check e verificare che nella pagina di visualizzazione scompaiono i relativi campi. | | |
| Risultati attesi: | L'utente limitato non può visualizzare questa pagina. Quando disattiva i check sullo schermo vengono nascosti i relativi campi dell'allarme. | | |

| | | | |
|---------------------|---|--------------|---|
| Test Case: | TC-010 | Nome: | Funzionamento pagina di visualizzazione |
| Riferimento: | REQ-005 REQ-006 | | |
| Descrizione: | Nella pagina di visualizzazione vengono visualizzati gli allarmi. Se non ci sono degli allarmi viene solamente visualizzato la mappa di rete. | | |

| | |
|--------------------------|---|
| Prerequisiti: | - |
| Procedura: | <ol style="list-style-type: none"> 1. Interrompere un servizio. 2. Assicurare che nella pagina di visualizzazione compare l'allarme. 3. Verificare che quanto è attivo il check "mappa di rete", viene stampato sulla destra la mappa di rete. 4. Verificare che quando è disattivo il check "mappa di rete", viene stampata la tabella su tutto lo schermo. 5. Riavvio il servizio 6. Assicurare che l'allarme scompaia e che la mappa di rete occupa tutto lo spazio disponibile sullo schermo. |
| Risultati attesi: | Un'utente visualizza gli allarmi quando viene interrotto un servizio. Se non ci sono degli allarmi viene visualizzato su tutto lo schermo la mappa di rete. |

| | | | |
|--------------------------|---|--------------|--|
| Test Case: | TC-011 | Nome: | Funzionamento della rete del raspberry |
| Riferimento: | REQ-007 | | |
| Descrizione: | Quando il raspberry si collega alla rete deve avere l'indirizzo IP 10.20.4.50. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Avviare il raspberry. 2. Assicurare che il raspberry sia connesso alla rete della CPT. 3. Aprire il terminale e digitare il comando ip a per vedere l'indirizzo IP dell'interfaccia di rete. 4. Assicurare che l'indirizzo IP sia 10.20.4.50. 5. Aprire un browser e verificare che l'utente riesca a navigare in internet. | | |
| Risultati attesi: | L'utente riesce a navigare in internet e l'indirizzo IP del raspberry è 10.20.4.50. | | |

| | | | |
|--------------------------|--|--------------|--------------------------------------|
| Test Case: | TC-012 | Nome: | Funzionamento dello script autostart |
| Riferimento: | REQ-007 | | |
| Descrizione: | Quando il raspberry si avvia viene eseguito lo script autostart che permette di rimuovere il cursore del mouse e aprire il browser chromium sulla pagina monitor . | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Avviare il raspberry. 2. Assicurare che il cursore del mouse scompaia. 3. Verificare che la pagina monitor appare a schermo intero sul televisore. | | |
| Risultati attesi: | Il cursore del mouse scompaia e appare a schermo intero la pagina schermo . | | |

| | | | |
|----------------------|--|--------------|----------------------------|
| Test Case: | TC-013 | Nome: | Funzionamento servizio ssh |
| Riferimento: | REQ-007 | | |
| Descrizione: | Il servizio ssh permette di collegarsi con utente da remoto sul raspberry. | | |
| Prerequisiti: | - | | |

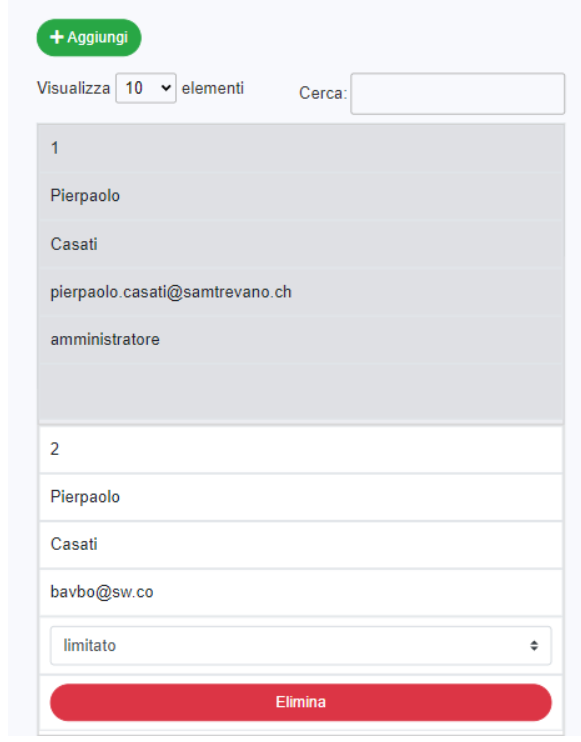
| | |
|--------------------------|---|
| Procedura: | <ol style="list-style-type: none"> 1. Avviare il raspberry. 2. Aprire il terminale e digitare il comando systemctl status ssh per potere verificare lo stato del servizio ssh. 3. Se il servizio non è attivo digitare il comando systemctl enable ssh a terminale. 4. Eseguire un ping all'indirizzo IP 10.20.4.50. 5. Assicurare che il ping funzioni. 6. Eseguire una connessione ssh. |
| Risultati attesi: | L'utente riesce a collegarsi sul raspberry da remoto. |

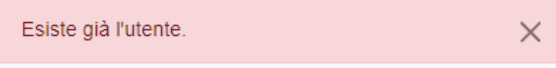
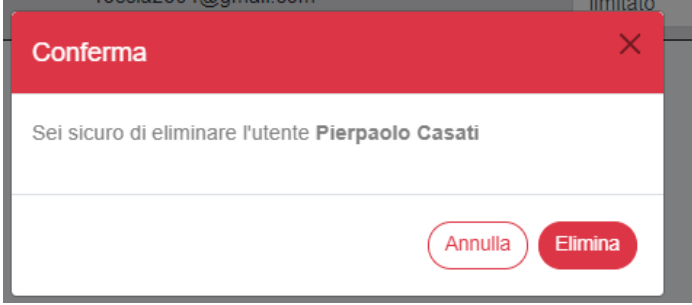
| | | | |
|--------------------------|---|--------------|-------------------------------|
| Test Case: | TC-014 | Nome: | Sicurezza: Prepared statement |
| Riferimento: | REQ-001 | | |
| Descrizione: | UN utente malevole non può eseguire delle SQL Injection.. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Eseguire la seguente SQL Injection: SELECT * FROM utente WHERE nome = " OR '1 2. Assicurare che l'utente non possa eseguirla. | | |
| Risultati attesi: | Se non ci sono le prepared statement e viene eseguita la SQL Injection scritta nella procedura, si potrebbe forzare la selezione di tutti i campi dati (*) di "tutti" gli utenti piuttosto che di un singolo nome come era inteso dal codice, ciò accade perché la valutazione di '1'='1' è sempre vera. Volendo un utente potrebbe scoprire lo username dell'utente. | | |


| | | | |
|--------------------------|--|--------------|-----------------------|
| Test Case: | TC-015 | Nome: | Sicurezza: test input |
| Riferimento: | REQ-001 | | |
| Descrizione: | Un utente malevole non può eseguire delle XSS scripting. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Provare ad inserire dello script malevolo nei campi di un form 2. Verificare nel database che il tag html è stato interpretato nella maniera corretta. 3. Rimuovere la funzione htmlspecialchars 4. Verificare che lo script venga eseguito. | | |
| Risultati attesi: | Se un utente inserisce dello script malevolo nei campi utilizzando dei tag HTML, l'applicativo web con l'utilizzo di htmlspecialchars permette di interpretare i tag HTML. | | |


| | | | |
|--------------------------|---|--------------|------------------|
| Test Case: | TC-016 | Nome: | Sicurezza: token |
| Riferimento: | REQ-001 | | |
| Descrizione: | Un utente non può modificare la password se non conosce il token. | | |
| Prerequisiti: | - | | |
| Procedura: | <ol style="list-style-type: none"> 1. Provare ad accedere https://raspberrypi/gestione_allarmi/login/changePassword/333. 2. Assicurare che si viene reindirizzato alla pagina di login. 3. Creare un nuovo utente. 4. Verificare che viene creato il token. 5. Accedere alla pagina di prima con il vero token. 6. Assicurare che si possa accedere alla pagina di modifica password. | | |
| Risultati attesi: | L'utente non può modificare la password se non conosce il codice del token. | | |

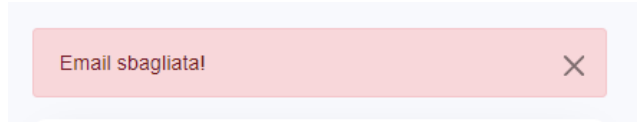
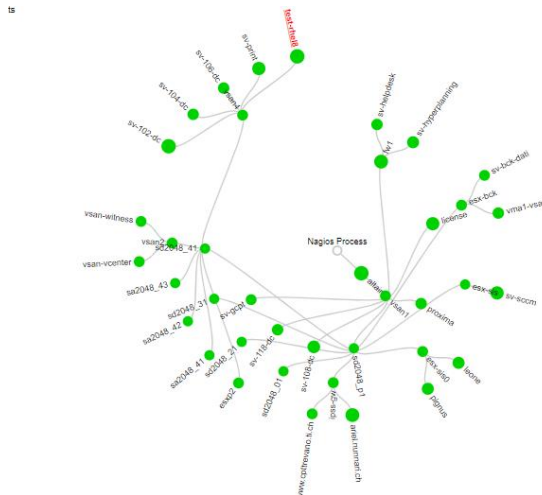
6.2 Risultati test

| Test Case | Funzionamento | Commento | Data |
|-----------|---------------|---|------------|
| TC-001 | PASSATO | Si riesce ad utilizzare l'applicativo sui diversi browser. | 20.05.2021 |
| TC-002 | PASSATO | La libreria bootstrap viene caricata e funziona correttamente. | 20.05.2021 |
| TC-003 | PASSATO | Nel sito sono presenti diversi elementi che utilizzano la libreria bootstrap. | 20.05.2021 |
| TC-004 | PASSATO | <p>Quando viene ridimensionato la pagina web gli elementi si adattano.</p> <p>Pagina Gestione utente versione mobile</p>  | 20.05.2021 |
| TC-005 | PASSATO | Quando clicco l'icona + Aggiungi compare il seguente form. | 21.05.2021 |

| | | | |
|--------|---------|---|------------|
| | |  <p>Crea Utente</p> <p>Nome</p> <p>Cognome</p> <p>Email</p> <p>Amministratore</p> <p>Crea Esci</p> <p>Quando clicco il pulsante “Crea” compaiono i seguenti messaggi di errore (l'utente è obbligato compilare i campi)</p>  <p>Campo nome obbligatorio!</p> <p>Se inserisco un e-mail di un account che esiste già compare il seguente messaggio.</p>  <p>Esiste già l'utente.</p> <p>Invece se inserisco dei valori corretti viene creato il nuovo utente e la tabella utenti viene aggiornata.</p> | |
| TC-006 | PASSATO | <p>Quando clicco il pulsante “Elimina” compare il seguente popup di conferma.</p>  <p>Conferma</p> <p>Sei sicuro di eliminare l'utente Pierpaolo Casati</p> <p>Annulla Elimina</p> | 21.05.2021 |
| TC-007 | PASATO | <p>Quando clicco sul link password dimenticata mi compare il seguente form che mi chiede di inserire l'e-mail del proprio account.</p> | 21.05.2021 |

| | | |
|--|--|-----------------|
|  | SAMT – Sezione Informatica | Pagina 52 di 59 |
| | Gestione degli allarmi e informazioni visualizzate dall'applicativo Nagios su di un monitor tramite API | |

| | | | |
|--|--|--|--|
| | | <div data-bbox="582 291 1216 701"> <h3>Password dimenticata</h3> <div> <input type="text" value="Email"/> </div> <div> <input type="button" value="Richiedi nuova password"/> <input type="button" value="Esci"/> </div> </div> <p>Guardare che sia arrivata un e-mail che permette di modificare la password. Si può solamente email di destinazione gmail, altrimenti non arriva niente. Inoltre il codice è sbagliato nell'email.</p> <div data-bbox="628 898 1037 934"> Credenziali account Posta in arrivo x </div> <div data-bbox="560 972 616 1023">  </div> <div data-bbox="628 969 1003 1021"> Gestione allarmi <system@gestione-allarmi.ch> a me ▾ </div> <div data-bbox="628 1052 1090 1081"> Benvenuto Pierpaolo Casati su Gestione allarmi </div> <div data-bbox="628 1099 1160 1126"> Per modificare la password cliccare il seguente link: Gestione allarme </div> <p>Cliccare sul link e aspettare che viene caricato la seguente pagina che permette di modificare la password.</p> <div data-bbox="582 1321 1200 1852"> <h3>Cambia password</h3> <div> <input type="text" value="Password"/> </div> <div> <input type="text" value="Conferma password"/> </div> <div> <input type="button" value="Conferma"/> </div> </div> | |
|--|--|--|--|

| TC-008 | PASSATO | <p>Se inserisco delle credenziali sbagliate nel login mi viene mostrato un messaggio di errore</p>  | 21.05.2021 | | | | |
|------------|----------|---|------------|-------|------------|----------|------------|
| TC-009 | PASSATO | <p>Quando clicco sui check viene visualizzato nella tabella solamente i relativi campi.</p> <p>Ad esempio seleziono solamente il campo host e stato.</p> <table border="1"> <thead> <tr> <th>Host</th> <th>Stato</th> </tr> </thead> <tbody> <tr> <td>test-rhel8</td> <td>Critical</td> </tr> </tbody> </table> | Host | Stato | test-rhel8 | Critical | 22.05.2021 |
| Host | Stato | | | | | | |
| test-rhel8 | Critical | | | | | | |
| TC-010 | PASSATO | <p>Se non ci sono degli allarmi viene solamente visualizzata la mappa di rete.</p>  | 22.05.2021 | | | | |
| TC-011 | PASSATO | <p>Riesco ad eseguire un ping sul IP 10.20.4.50</p> | 22.05.2021 | | | | |
| TC-012 | PASSATO | <p>Quando il raspberry si accende, sul televisore compare un finestra del browser a schermo intero che mostra la pagina web dove vengono visualizzate le informazioni o i filmati/presentazioni.</p> | 22.05.2021 | | | | |
| TC-013 | PASSATO | <p>Riesco ad eseguire una connessione SSH e quindi posso accedere al raspberry da remoto.</p> | 25.05.2021 | | | | |
| TC-014 | PASSATO | <p>Un utente non può eseguire delle SQL Injection, perché ci sono le prepared statement.</p> | 25.05.2021 | | | | |
| TC-015 | PASSATO | <p>Se inserisco degli script malevoli nei campi, vengono interpretati come caratteri speciali.</p> | 25.05.2021 | | | | |
| TC-016 | | <p>Viene generato il token quando viene creato un nuovo utente e non si può accedere alla pagina di modifica se non si conosce il token.</p> | 25.05.2021 | | | | |

6.3 Mancanze/limitazioni conosciute

Per il progetto sono stati sviluppati tutti i requisiti che erano presenti nel quaderno dei compiti. L'unica limitazione è il sistema di email, ovvero il sistema che permette di mandare le email. Per realizzare il sistema di email ho utilizzato la libreria PHPMailer che mi permette di creare dei sistemi no-reply senza l'appoggio di un SMTP. PHPMailer utilizza le funzione PHP mail che ha sua volta utilizza il comando sendmail. In effetti ho dovuto installare il pacchetto sul raspberry. Il sistema di email non funziona se cerco di inviare delle email ad un account samtrevano o edu, dovuto da problemi di politica di sicurezza. Ma riesco comunque mandare delle email ha degli account gmail. Quando andrà in produzione per i sistemisti questo sistema di email verrà rimosso.

7 Consuntivo

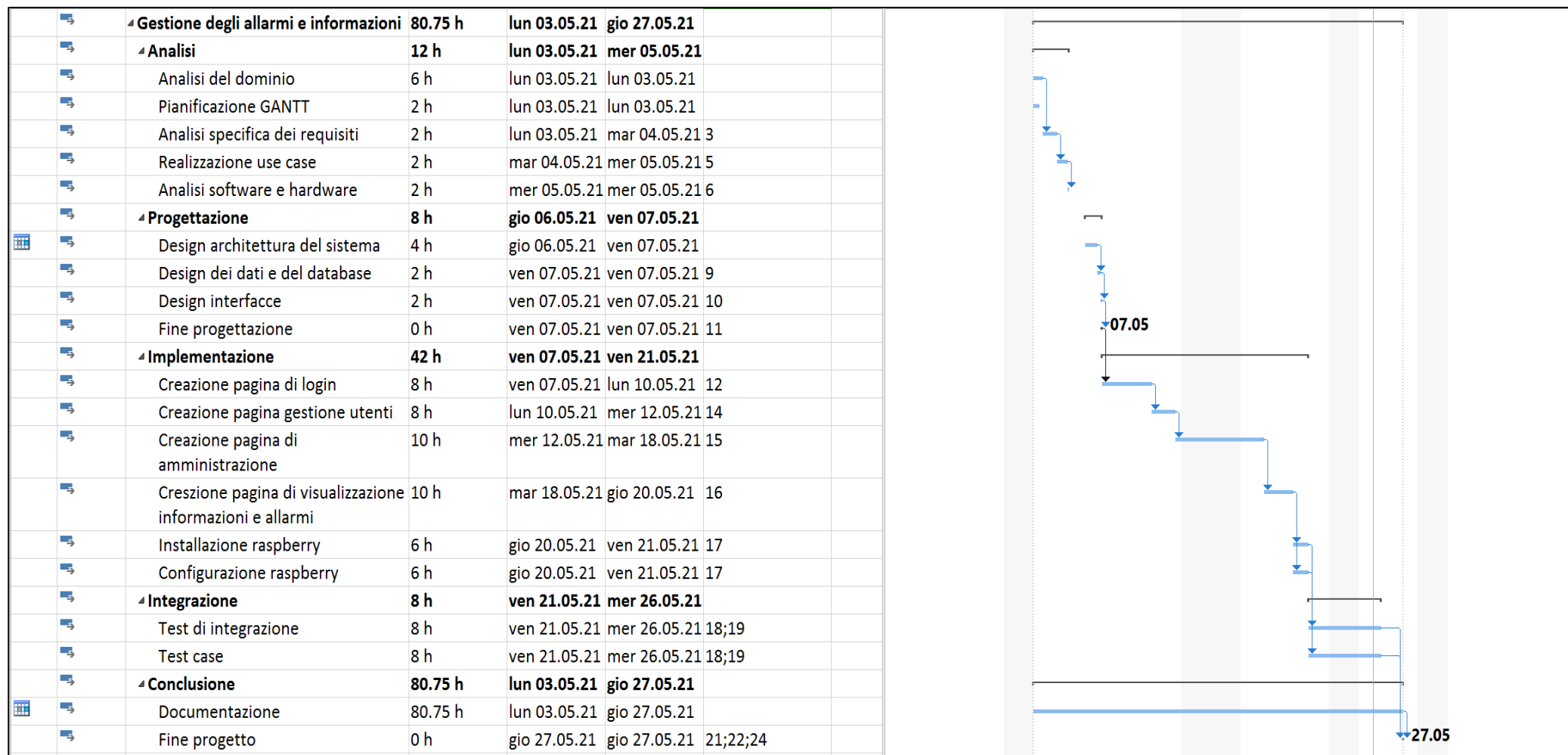


Figura 45: GANTT consuntivo

Durante la progettazione dell'applicativo web ho avuto un imprevisto, ovvero il mercoledì 05.05.2021 ho dovuto assentarmi per il reclutamento e quindi ho dovuto recuperare queste ore di lavoro al di fuori dall'orario previsto del LPI. Le ore totali della fase di **Analisi** non sono aumentate, però si può notare che ho spostato l'attività **Pianificazione GANTT** dalla progettazione all'analisi. Quest'ultima è stata spostata, perché mi è stato richiesto dal perito di realizzare già una pianificazione per il primo giorno di lavoro. Anche per quanto concerne le ore di della fase di **Progettazione** non sono aumentate e si può notare come spiegato già per l'analisi, l'attività **Pianificazione GANTT** è stata eliminata. Inoltre, a causa del reclutamento non ho potuto iniziare subito la progettazione e quindi rispetto alla pianificazione preventiva la data di inizio della fase è sfasata di un giorno. Per quanto riguarda la fase di **Implementazione** le ore di lavoro sono rimaste le stesse e le ore di ogni attività sono state stimate in modo corretto. In effetti la maggior parte delle attività le avevo già realizzate in precedenti progetti e quest'ultimo mi ha permesso di velocizzare i tempi di implementazione. Per quanto riguarda la fase di **Integrazione** ho realizzato tutti i miei test case necessari per potere verificare il giusto funzionamento dei requisiti. Ho anche realizzato dei test di integrazione che permettono di verificare il giusto funzionamento delle varie funzionalità. Per finire si può notare che per potere realizzare la documentazione ci ho messo di meno rispetto alla pianificazione preventiva. Le ore sono diminuite perché nei tempi ho aggiunto anche le pause.

8 Conclusioni

8.1 Sviluppi futuri

Il mio progetto andrà in produzione per i sistemisti della CPT e quindi verranno sicuramente implementati degli sviluppi in futuro. Uno dei primi sviluppi è quello di rimuovere il sistema di email e creare un nuovo sistema per potere modificare la password provvisoria. Per realizzare questo sistema devo generare un password casuale dall'applicativo che l'amministratore dovrà fornire all'utente. L'ultimo sviluppo futuro è aggiustare i vari bug che compaiono sulla pagina di visualizzazione. Il primo bug è che ogni tanto viene ricaricata la pagina e quindi a sua volta viene ricaricata la tabella degli allarmi e la mappa di rete. Il secondo bug invece è che ogni tanto nella tabella allarme vengono creati le stesse allarmi. In effetti poi sul monitor viene visualizzata la stessa allarme e quindi sono dei dati ridondanti. L'ultimo problema non è proprio un bug, ma quando viene acceso il monitor e viene visualizzato a schermo intero la pagina di visualizzazione degli allarmi, l'utente deve autenticarsi ogni volta per potere accedere alla risorse di Nagios. In effetti se non si autentica, non viene visualizzata la mappa di rete.

8.2 Considerazioni personali

Ho trovato questo progetto molto interessante, perché ho potuto mettere in pratica tutte le mie conoscenze informatiche per lo sviluppo di applicazioni web e per l'installazione/configurazioni di sistemi informatici. Ritengo che questa applicazione web permetterà ai sistemisti di gestire meglio le apparecchiature della rete, perché tramite il monitor esterno verranno avvisati con un allarme. In oltre hanno anche la possibilità di localizzare molto velocemente l'apparecchiatura tramite la visualizzazione di una mappa di rete. Per finire trovo che sia molto giusto implementare gli sviluppi futuri, perché potrebbero migliorare diverse problematiche che possono accadere durante la visualizzazione. In effetti vorrei sistemare i diversi bug che si manifestano durante la visualizzazione, perché non sono molto belli da vedere e potrebbero confondere facilmente il sistemista.

9 Glossario

| | |
|---------------|---|
| AJAX | Asynchronous JavaScript and XML ed è un tecnica di sviluppo software per realizzare delle applicazioni web interattive, basandosi su uno scambio di dati in background tra browser e server, consentendo così l'aggiornamento dinamico di una pagina web. |
| Bootstrap | Bootstrap è toolkit open source front-end più popolare al mondo, con variabili Sass e mixin, sistema di griglie reattive, numerosi componenti predefiniti e potenti plug-in JavaScript. |
| CSS | Cascading Style Sheets è un linguaggio che gestisce il design e la presentazione delle pagine web. |
| GANTT | Il GANTT è uno strumento di supporto alla gestione dei progetti. |
| GET | GET svolge la funzione di richiedere una risorsa, come ad esempio un file HTML, a un web server. |
| HTML | HyperText Markup Language è un linguaggio di markup noto per la formattazione e impaginazione di documenti ipertestuali disponibili nel web. |
| HTTP | Hypertext Transfer Protocol è un protocollo usato per la trasmissione d'informazioni web. |
| HTTPS | Hypertext Transfer Protocol Secure è un protocollo per la comunicazione sicura attraverso una rete. |
| JavaScript | JavaScript è un linguaggio di programmazione orientato agli oggetti e agli eventi, utilizzato nella programmazione Web lato client (anche lato server) per la creazione di siti dinamici tramite funzioni di script. |
| JQuery | JQuery è una libreria di JavaScript leggera. Il suo scopo è quello di rendere molto più semplice l'utilizzo di JavaScript sul tuo sito web. |
| MVC | Model-View-Controller è un pattern per lo sviluppo di sistemi software nell'ambito della programmazione orientata ad oggetti e in applicazione web. |
| MySQL | MySQL è un sistema open source di gestione i database relazionali SQL sviluppato e supportato da Oracle. |
| PDO | PDO (PHP Data Object) è un estensione di PHP introdotta dalla versione 5.1 del linguaggio con lo scopo di unificare le API di accesso ai database. |
| PHP | PHP (Hypertext Preprocessor) è un linguaggio di scripting generico adatto allo sviluppo web. |
| POST | Le richieste POST vengono principalmente utilizzate in relazione ai moduli da compilare online |
| SQL | Structured Query Language è un linguaggio standardizzato per database basati su modello relazionale. |
| SQL Injection | Con SQL injection si intende lo sfruttamento di una vulnerabilità nei database relazionali, che utilizzano il linguaggio SQL per l'inserimento dei dati. |
| SSH | Secure Shell è un protocollo che permette di stabilire una sessione remota cifrata tramite interfaccia a riga di comando con un altro host di una rete informatica. |
| UML | L'Unified Modeling Language (UML) è un linguaggio di modellazione generale, evolutivo nel campo |

| | |
|-----|---|
| | dell'ingegneria del software che ha lo scopo di fornire un modo standard per visualizzare la progettazione di un sistema. |
| XSS | Cross-site scripting è una vulnerabilità informatica che affligge siti web dinamici che impiegano un insufficiente controllo dell'input nei form. |

10 Bibliografia

10.1 Sitografia

<https://draw.io>, Flowchart Marker and Online Diagram Software, 04.05.2021.
<https://www.apachefriends.org/index.html>, XAMP: PHP development environment, 03.05.2021.
<https://getbootstrap.com/>, Build fast, responsive sites with Bootstrap, 07.05.2021.
<https://fontawesome.com>, Font containing web-related icons, 07.05.2021.
<https://datatables.net>, Add advanced interaction controls to your HTML table, 07.05.2021.
<https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>, Setting up An Apache Web Server on a Raspberry PI, 20.05.2021.
<https://www.raspberrypi.org>, Raspberry PI website, 20.05.2021.
<https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/toc.html>, documentation Nagios, 12.05.2021.

11 Allegati

- Abstract
- Diari di lavoro
- QdC
- Manuali di utilizzo
- Codice sorgente presente su gitsam: http://gitsam.cpt.local/lavoro_finale_lpi_2021/gestione-allarmi