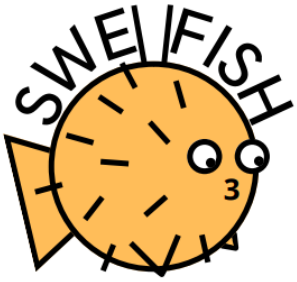


Piano di qualifica

swellfish14@gmail.com

Informazioni

Redattori	[Davide Porporati, Claudio Giaretta, Francesco Naletto]
Revisori	[Jude Vensil Braceros]
Responsabili	[Andrea Veronese]
Uso	[Esterno]

Descrizione

File contenente il piano di qualifica. Contiene le metriche e i criteri di accettazione dei prodotti.

Versione	Data	Redattore	Verificatore	Descrizione
2.0.0	21/09/2023	Claudio Giaretta	Francesco Naletto	Verifica generale e major update
1.0.4	14/09/2023	Claudio Giaretta	Francesco Naletto	Aggiornato Valutazione delle metriche
1.0.3	12/09/2023	Francesco Naletto	Davide Porporati	Aggiunta sezione test di unità
1.0.2	07/08/2023	Andrea Veronese	Claudio Giaretta, Davide Porporati	Aggiunte metriche da usare nei test di verifica
1.0.1	07/06/2023	Claudio Giaretta	Andrea Veronese, Davide Porporati	Creata struttura specifiche di test e aggiunti test di sistema
1.0.0	18/07/2023	Andrea Veronese	Claudio Giaretta, Davide Porporati	Aggiornato a versione 1.0.0
0.1.1	13/07/2023	Claudio Giaretta	Francesco Naletto	Aggiornamento dei grafici e aggiunta delle ultime considerazioni del gruppo
0.1.0	25/05/2023	Claudio Giaretta, Francesco Naletto	Francesco Naletto	Stesura introduzione, qualità di processo e qualità di prodotto

0.0.2	27/04/2023	Davide Porporati, Elena Marchioro, Francesco Naletto	Jude Vensil Braceros	Modificata la struttura del documento
0.0.1	25/04/2023	Andrea Veronese	Davide Porporati	Creata struttura di base del documento

Contents

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di definire le strategia di validazione e verifica addottate per garantire la qualità del prodotto. Per raggiungere questo obbiettivo viene applicato un sistema di verifica continua sui processi e sulle attività del gruppo, questo permette di ottenere un miglioramento continuo. Il documento non ha una funzione descrittiva, la definizione delle metriche indicate all'interno di questo documento, è presente nel documento "norme_di_progetto".

2 Qualità di processo

2.1 Processi primari

2.1.1 Fornitura

Metriche:

- MPC01: Actual Cost (AV)
  - **Calcolo della metrica**: Somma dei costi tracciati dal gruppo
  - **Valore ottimale**: $\leq BAC$
  - **Valore accettabile**: $\leq BAC$
- MPC02: Planned Value (PV)
  - **Calcolo della metrica**: Percentuale di completamento del progetto pianificata * BAC
  - **Valore ottimale**: $\leq BAC$
  - **Valore accettabile**: $\leq BAC$
- MPC03: Earned Value (EV)
  - **Calcolo della metrica**: Percentuale dell'effettivo stato di completamento del progetto * BAC
  - **Valore ottimale**: ≥ 0
  - **Valore accettabile**: $\leq BAC$

- MPC04: Cost Variance (CV)
  - **Calcolo della metrica**: EV - AC
  - **Valore ottimale**: $\geq 0\%$
  - **Valore accettabile**: $\geq -12\%$
- MPC05: Schedule Variance (SV)
  - **Calcolo della metrica**: EV - PV
  - **Valore ottimale**: $\geq 0\%$
  - **Valore accettabile**: $\geq -12\%$
- MPC06: Cost Performance Index (CPI)
  - **Calcolo della metrica**: EV / AC
  - **Valore ottimale**: ≥ 1
  - **Valore accettabile**: $\geq 0,9$
- MPC07: Estimated At Completition (EAC)
  - **Calcolo della metrica**: BAC / CPI
  - **Valore ottimale**: = BAC
  - **Valore accettabile**: $\geq BAC - 3\%; \leq BAC + 3\%$
- MPC08: Estimate To Completition (ETC)
  - **Calcolo della metrica**: (BAC - EV) / CPI
  - **Valore ottimale**: $\geq 0\%$
  - **Valore accettabile**: $\leq EAC$

2.2 Processi di supporto

2.2.1 Documentazione

Metriche:

- MPC09: Indice di Gulpease
  - **Calcolo della metrica**: $89 + \frac{300*(F)-10*(L)}{(P)}$

    - $*$ **L** = Numero di lettere nel testo
    - $*$ **P** = Numero di parole nel testo
    - $*$ **F** = Numero di frasi nel testo
  - **Valore ottimale**: 100 %
  - **Valore accettabile**: $\geq 60\%$

- MPC10: Errori ortografici
  - **Calcolo della metrica**: numero errori ortografici presenti nel testo
  - **Valore ottimale**: 0
  - **Valore accettabile**: 0

2.2.2 Verifica

Metriche:

- MPC11: Statement Coverage.
  La metrica si basa sullo statement coverage. Indica la percentuale di statement eseguiti almeno una volta dall'insieme dei testi di unità.
  I valori sono forniti dalla suite di testing Jest.

Prodotto	**Valore accettabile**	**Valore ottimale**
Software	$> 80\%$	$> 95\%$

- MPC12: Branch Coverage
  La metrica si basa sul branch coverage. Indica la percentuale di branch che vengono testati almeno una volta, con esito positivo.
  Valori fonriti dal report di Jest.

Prodotto	**Valore accettabile**	**Valore ottimale**
Software	$> 75\%$	$> 95\%$

- MPC13: Code Coverage
  La metrica si basa sul code coverage. Indica la percentuale di codice eseguito nella fase di testing.
  Valori fonriti dal report di Jest.

Prodotto	**Valore accettabile**	**Valore ottimale**
Software	$> 80\%$	$> 95\%$

- MPC14: Condition Coverage
  La metrica si basa sul condition coverage. Indica la percentuale di branch che risultano almeno una volta true e almeno una volta false nell'esecuzione di un test dedicato.
  Valori fonriti dal report di Jest.

Prodotto	**Valore accettabile**	**Valore ottimale**
Software	$> 70\%$	$> 80\%$

2.3 Processi organizzativi

3 Qualità di prodotto

3.1 Introduzione

Per assicurare la qualità del prodotto, abbiamo adottato lo standard ISO/IEC 9126 come punto di riferimento. In questa sezione, forniamo i valori ottimali e accettabili per le metriche selezionate dal gruppo SWEllFish.

3.2 Affidabilità

Metriche:

- MPD01: Percentuale di difetti del prodotto.
  - Valore ottimale: 80%.
  - Valore accettabile: 60%.
  - Note: I valori possono essere modificati.

3.3 Efficienza

Metriche:

- MPD02: Tempo medio di risposta.
  - Metrica di misurazione: Secondi.
  - Valore ottimale: 5 secondi.
  - Valore accettabile: 7 secondi.

3.4 Funzionalità

Metriche:

- MPD03: Percentuale di copertura dei requisiti.
  - Valore ottimale: 100% dei requisiti obbligatori e 80% dei requisiti opzionali.
  - Valore accettabile: 100% dei requisiti obbligatori.

3.5 Manutenibilità

Metriche:

- MPD04: Percentuale di comprensibilità del codice.
  - Valore ottimale: 85% - 100%.
  - Valore accettabile: 65%.

3.6 Portabilità

Metriche:

- MPD05: Percentuale di compatibilità del prodotto.
  - Valore ottimale: 85% - 100%.
  - Valore accettabile: 60%.

3.7 Usabilità

Metriche:

- MPD06: Numero di errori compiuti dagli utenti durante l'utilizzo del prodotto.
  - Valore ottimale: Inferiore a 1 errore per utente.
  - Valore accettabile: Inferiore a 2 errori per utente.

4 Specifica di test

4.1 Test di accettazione

I test di accettazione sono essenziali per dimostrare la conformità del prodotto ai requisiti minimi concordati con il proponente. Questa fase comprende i test di sistema che vengono condotti durante il collaudo finale, coinvolgendo sia i membri del team di sviluppo che i rappresentanti dell'azienda proponente. L'intero processo è attentamente sorvegliato dal nostro gruppo.

4.2 Test di sistema

ID Test	Descrizione	Requisito	Stato
TS1	Verificare che l'utente riesca ad effettuare correttamente l'accesso a sistema	RF1	NI
TS2	Verificare che l'utente visualizzi correttamente lo stato del sistema	RF2	NI
TS3	Verificare che l'utente sia in grado di cambiare la luminosità correttamente	RF3	NI
TS4	Verificare che il sistema visualizzi correttamente il messaggio di errore nel caso in cui l'aumento della luminosità non fosse andato a buon fine	RF4	NI
TS5	Verificare che l'utente possa visualizzare correttamente la lista delle aree illuminate	RF5	NI

ID Test	Descrizione	Requisito	Stato
TS6	Verificare che l'utente possa vedere correttamente l'elenco delle zone	RF6	NI
TS7	Verificare che l'utente possa selezionare le zone correttamente	RF7	NI
TS8	Verificare che l'utente possa diminuire la luminosità di una zona correttamente	RF8	NI
TS9	Verificare che l'utente possa accedere correttamente alla dashboard	RF10	NI
TS10	Verificare che il sistema visualizzi correttamente il messaggio di errore nel caso la diminuzione della luminosità non fosse andata a buon fine	RF11	NI
TS11	Verificare che l'utente possa diminuire la luminosità correttamente	RF12	NI
TS12	Verificare che l'utente possa inserire una nuova area illuminata correttamente	RF13	NI
TS13	Verificare che l'utente possa rimuovere un area di illuminazione correttamente	RF14	NI
TS14	Verificare che l'utente possa accedere alla lista delle zone gestite correttamente	RF15	NI
TS15	Verificare che l'utente possa modificare le informazioni di un area illuminata	RF16	NI
TS16	Verificare che il sisteam mostri correttamente il messaggio di notifica una volta fatta la modifica all'area illuminata	RF17	NI
TS17	Verificare che l'utente possa inserire correttamente un sensore in un area illuminata	RF18	NI
TS18	Verificare che l'utente possa accedere correttamente all'area illuminata	RF19	NI
TS19	Verificare che l'utente possa rimuovere un sensore dall'area illuminata	RF20	NI

ID Test	Descrizione	Requisito	Stato
TS20	Verificare che l'utente possa fare il logout dal sistema correttamente	RF21	NI
TS21	Verificare che l'utente possa inserire un impianto nell'elenco dei guasti correttamente	RF22	NI
TS22	Verificare che l'utente possa rimuovere un impianto dall'elenco dei guasti correttamente	RF23	NI
TS23	Verificare che l'utente possa visualizzare i dettagli di una zona correttamente	RF24	NI
TS24	Verificare che l'utente possa selezionare un lampione correttamente	RF25	NI
TS25	Verificare che l'utente possa visualizzare i dettagli di un lampione correttamente	RF26	NI
TS26	Verificare che l'utente possa inserire un nuovo lampione all'interno di un'area illuminata correttamente	RF27	NI
TS27	Verificare che l'utente possa rimuovere un lampione all'interno di un'area illuminata correttamente	RF28	NI
TS28	Verificare che l'utente possa visualizzare l'elenco delle aree illuminate con dei malfunzionamenti correttamente	RF29	NI
TS29	Verificare che l'amministratore possa poter aprire una nuova segnalazione di un guasto tramite un ticket	RF30	NI
TS30	Verificare che l'amministratire possa poter chiudere il ticket dopo aver fatto la dovuta manutenzione correttamente	RF31	NI
TS31	Verificare che il manutentore possa visualizzare i dettagli aggiuntivi di un guasto forniti dal ticket correttamente	RF32	NI

ID Test	Descrizione	Requisito	Stato
TS32	Verificare che l'utente non amministratore possa riceve le credenziali di amministratore da un superamministratore	RF33	NI
TS33	Verificare che l'utente possa consultare il manuale Lumos Minima	RF34	NI
TS34	Verificare che le nuove aree illuminate appena inserite abbiano un setup standard	RF35	NI

4.3 Test di integrazione

ID Test	Descrizione	Stato
TI1	Verificare che il collegamento con backend avvenga correttamente.	S
TI2	Verificare che il collegamento con simulatori avvenga correttamente.	S

4.4 Test di unità

4.4.1 Test di unità - Frontend

ID Test	Descrizione	Stato
TU01	Verifica che il componente AggiungiAreaView venga renderizzato correttamente	S
TU02	Verifica che venga renderizzato un messaggio di errore quando il valore di submitIsError è true	S
TU03	Verifica che vengano cancellati gli errori correttamente	S
TU04	Verifica che il componente AggiungiGuastoView venga renderizzato correttamente	S
TU05	Verifica che venga renderizzato un messaggio di errore quando il valore di submitIsError è true	S
TU06	Verifica che venga renderizzato correttamente se areaDetails non è definita	S
TU07	Verifica che il componente AggiungiLampioneView venga renderizzato correttamente	S
TU08	Verifica che venga renderizzato un messaggio di errore quando il valore di submitIsError è true	S

ID Test	Descrizione	Stato
TU09	Verifica che il componente AggiungiSensoreView venga renderizzato correttamente	S
TU10	Verifica gestione dei submit con errore	S
TU11	Verifica gestione cambiamenti agli input del form	S
TU12	Verifica chiamata a clearError onFormFocus	S
TU13	Verifica se il componente AreaDetailsView viene renderizzato correttamente quando il caricamento è in corso. Il test cerca la presenza del testo "Loading..." nella vista.	S
TU14	Verifica se il componente AreaDetailsView viene renderizzato correttamente quando non ci sono errori. Il test cerca la presenza del testo "Dettagli area" nella vista.	S
TU15	Verifica se il pulsante "Accendi Lampioni Area" reagisce correttamente al click, ossia se chiama la funzione accendiLampioniArea del ViewModel.	S
TU16	Verifica se il pulsante "Aumenta Luminosità" reagisce correttamente al click, ossia se chiama la funzione aumentaLuminosità del ViewModel.	S
TU17	Verifica se il pulsante "Diminuisci Luminosità" reagisce correttamente al click, ossia se chiama la funzione diminuisciLuminosità del ViewModel.	S
TU18	Verifica se il pulsante "Spegni Lampioni Area" reagisce correttamente al click, ossia se chiama la funzione spegniLampioniArea del ViewModel.	S
TU19	Verifica se il pulsante "Cambia Modalità" reagisce correttamente al click, ossia se chiama la funzione cambiaModalità del ViewModel.	S
TU20	Verifica se il pulsante "Elimina Area" reagisce correttamente al click, ossia se chiama la funzione eliminaArea del ViewModel.	S
TU21	Verifica se il componente AreaDetailsView gestisce correttamente lo stato di errore, inclusa la visualizzazione del messaggio di errore.	S
TU22	Verifica se il componente AreaDetailsView mostra correttamente lo stato "Acceso" quando il ViewModel lo specifica.	S
TU23	Verifica se il componente AreaDetailsView mostra correttamente la luminosità in modalità automatica quando il ViewModel lo specifica.	S
TU24	Verifica come il componente AreaDetailsView gestisce gli errori di submit quando submitHasError è true.	S
TU25	Verifica come il componente AreaDetailsView gestisce il rendering quando i dati sono undefined.	S
TU26	viene verificato se il componente AreeView viene renderizzato correttamente.	S

ID Test	Descrizione	Stato
TU27	viene verificato se il componente AreeView gestisce correttamente lo stato di caricamento quando isLoading è impostato su true.	S
TU28	viene verificato se il componente GuastoDetailsView viene renderizzato correttamente.	S
TU29	verifica se il messaggio di caricamento "Loading..." è presente nella vista.	S
TU30	verifica se un messaggio di errore è presente nella vista.	S
TU31	verifica se la funzione chiudiGuasto del ViewModel è stata chiamata correttamente.	S
TU32	verifica che il componente non venga renderizzato quando stato è uguale a "1". In altre parole, si verifica che il componente non sia visibile nella vista.	S
TU33	verifica che il componente venga renderizzato correttamente quando stato è diverso da "1". In questo caso, si verifica la presenza del testo "Impostazioni Guasto" nella vista.	S
TU34	verifica che il componente sia ancora in grado di renderizzare correttamente anche quando i dati di guastoDetails sono undefined.	S
TU35	verificato se il componente HomeView viene renderizzato correttamente.	S
TU36	verifica se i messaggi di caricamento "Loading..." sono presenti nella vista.	S
TU37	viene verificato se il pulsante "Logout" è presente nella vista. Successivamente simula un clic su questo pulsante e verifica se la funzione logout del ViewModel è stata chiamata correttamente	S
TU38	verifica se gli elementi dei guasti sono stati renderizzati correttamente, confrontando i dati dei guasti presenti nel mock ViewModel con il testo previsto nella vista.	S
TU39	verificato se il componente ListaGuastiView viene renderizzato correttamente.	S
TU40	verifica se il messaggio di caricamento "Loading..." è presente nella vista.	S
TU41	verifica se il componente viene ancora renderizzato correttamente senza errori quando i dati sono mancanti.	S
TU42	viene verificato se il componente ListaLampioniView viene renderizzato correttamente.	S
TU43	verifica se il messaggio di caricamento "Loading..." è presente nella vista.	S
TU44	erifica se i dettagli dei lampioni sono visualizzati correttamente, compresi ID, IP, stato e tipo di interazione.	S

ID Test	Descrizione	Stato
TU45	Verifica se la funzione accendiLampione viene chiamata con l'ID corretto del lampione.	S
TU46	verifica se il componente viene ancora renderizzato correttamente senza errori quando i dati sono mancanti.	S
TU47	verifica se il componente viene renderizzato correttamente, inclusi i dettagli dei sensori (ID, IP, tipo di interazione, polling time e raggio di azione).	S
TU48	verifica se il messaggio di caricamento "Loading..." è presente nella vista.	S
TU49	verifica se il componente viene renderizzato correttamente e se il testo "Login" è presente nell'output.	S
TU50	verifica se il componente viene renderizzato correttamente e se i campi di input contengono i valori corrispondenti ai dettagli dell'area forniti dal ViewModel.	S
TU51	verifica se la funzione submit del ViewModel (mockViewModel.submit) è stata chiamata con i valori aggiornati del form. Ciò assicura che il form venga sottomesso correttamente con i dati modificati.	S
TU52	verifica se tutti i campi di input sono vuoti o nulli quando non ci sono dati dell'area disponibili. Questo copre il caso in cui l'utente voglia creare una nuova area e non ci siano dati preesistenti.	S
TU53	verifica se il componente viene renderizzato correttamente e se i campi di input contengono i valori corrispondenti ai dettagli del guasto forniti dal ViewModel.	S
TU54	verifica se la funzione submit del ViewModel (mockViewModel.submit) è stata chiamata con i valori aggiornati del form. Ciò assicura che il form venga sottomesso correttamente con i dati modificati.	S
TU55	verifica che il componente venga renderizzato correttamente anche quando lo stato del guasto è 1.	S
TU56	verifica se tutti i campi di input sono vuoti o nulli quando non ci sono dati del guasto disponibili. Questo copre il caso in cui l'utente voglia creare un nuovo guasto e non ci siano dati preesistenti.	S
TU57	verifica se il componente viene renderizzato correttamente e se i campi di input contengono i valori corrispondenti ai dettagli del lampione forniti dal ViewModel.	S
TU58	verifica se la funzione modificaLampione del ViewModel (mockViewModel.modificaLampione) è stata chiamata con i valori aggiornati del form. Ciò assicura che il form venga sottomesso correttamente con i dati modificati.	S
TU59	verifica se viene visualizzato un messaggio di errore appropriato quando si verifica un errore durante il submit del form.	S

ID Test	Descrizione	Stato
TU60	verifica se la funzione eliminaLampione del ViewModel (mockViewModel.eliminaLampione) è stata chiamata quando l'utente fa clic sul pulsante.	S
TU61	erifica se tutti i campi di input sono vuoti o nulli quando non ci sono dati del lampione disponibili. Questo copre il caso in cui l'utente voglia creare un nuovo lampione e non ci siano dati preesistenti.	S
TU62	verifica se il componente viene renderizzato correttamente e se i campi di input contengono i valori corrispondenti ai dettagli del sensore forniti dal ViewModel.	S
TU63	verifica se la funzione submit del ViewModel (mockViewModel.submit) è stata chiamata con i valori aggiornati del form. Ciò assicura che il form venga sottomesso correttamente con i dati modificati.	S
TU64	verifica se viene visualizzato un messaggio di errore appropriato quando si verifica un errore durante il submit del form.	S
TU65	verifica se la funzione eliminaSensore del ViewModel (mockViewModel.eliminaSensore) è stata chiamata quando l'utente fa clic sul pulsante.	S
TU66	verifica se tutti i campi di input sono vuoti o nulli quando non ci sono dati del sensore disponibili. Questo copre il caso in cui l'utente voglia creare un nuovo sensore e non ci siano dati preesistenti.	S
TU67	verifica se la funzione mutateAsync del aggiungiAreaMutation nell'oggetto areeStoreMock è stata chiamata.	S
TU68	verificato se la funzione submitIsError restituisce true quando submitError nell'oggetto areeStoreMock è impostato su un valore diverso da una stringa vuota.	S
TU69	verifica se la funzione setSubmitError nell'oggetto areeStoreMock è stata chiamata con il messaggio di errore restituito dalla chiamata a mutateAsync.	S
TU70	verifica che le proprietà e i metodi del ViewModel restituiscano valori definiti, confermando che il ViewModel sia stato creato correttamente e abbia le proprietà e i metodi attesi.	S
TU71	verifica se i metodi del store (mockStore) vengono chiamati correttamente quando vengono invocati i metodi del ViewModel.	S
TU72	verifica il comportamento del ViewModel quando si verifica un errore durante il submit.	S
TU73	verifica che le proprietà e i metodi del ViewModel restituiscano valori definiti, confermando che il ViewModel sia stato creato correttamente e abbia le proprietà e i metodi attesi.	S

ID Test	Descrizione	Stato
TU74	verifica se i metodi del store (mockStore) vengono chiamati correttamente quando vengono invocati i metodi del ViewModel.	S
TU75	verifica il comportamento del ViewModel quando si verifica un errore durante il submit.	S
TU76	verifica se la funzione getAreaDetails viene chiamata con l'ID corretto quando viene creato un'istanza di AreaDetailsViewModel.	S
TU77	verifica se la funzione isLoading restituisce true quando getAreaDetails restituisce un oggetto con isLoading impostato su true.	S
TU78	verifica se la funzione isError restituisce false quando getAreaDetails restituisce un oggetto con isError impostato su false.	S
TU79	verifica se la funzione aumentaLuminosità chiama la mutazione aumentaLuminositàMutation quando le condizioni sono soddisfatte.	S
TU80	verifica che la funzione aumentaLuminosità non chiami la mutazione aumentaLuminositàMutation quando la luminosità è già al massimo.	S
TU81	verifica se la funzione diminuisciLuminosità chiama la mutazione diminuisciLuminositàMutation quando le condizioni sono soddisfatte.	S
TU82	verifica che la funzione diminuisciLuminosità non chiami la mutazione diminuisciLuminositàMutation quando la luminosità è già al minimo.	S
TU83	verifica se la funzione eliminaArea chiama la mutazione eliminaAreaMutation.	S
TU84	verifica se la funzione cambiaModalità chiama la mutazione cambiaModalitaMutation.	S
TU85	verifica se la funzione accendiArea chiama la mutazione accendiAreaMutation.	S
TU86	verifica se la funzione accendiLampioniArea chiama la mutazione accendiLampioniAreaMutation.	S
TU87	verifica se la funzione spegniLampioniArea chiama la mutazione spegniLampioniAreaMutation.	S
TU88	verifica se la funzione aree restituisce un array contenente le aree simulate	S
TU89	verifica se la funzione isLoading restituisce false quando areeStoreMock ha isLoading impostato su false.	S
TU90	verifica se la funzione aumentaLuminositaCrepuscolo chiama la mutazione accendiAllAreeMutationMock.mutateAsync con gli argomenti previsti. In questo caso, viene verificato che sia chiamato senza argomenti (l'oggetto vuoto {}).	S

ID Test	Descrizione	Stato
TU91	verifica se la funzione diminuisciLuminositaCrepuscolo chiama la mutazione spegniAllAreeMutationMock.mutateAsync con gli argomenti previsti. In questo caso, viene verificato che sia chiamato senza argomenti (l'oggetto vuoto {}).	S
TU92	verifica se la funzione getGuastoDetails viene chiamata con l'ID corretto quando viene creato un'istanza di GuastoDetailsViewModel.	S
TU93	verifica se la funzione chiudiGuasto chiama la mutazione chiudiGuastoMutation e la funzione navigate con gli argomenti previsti quando è chiamata. Viene simulata una risposta di successo dalla mutazione.	S
TU94	verifica se la funzione numeroAree restituisce il numero corretto di aree, che è impostato come 10 nella simulazione areeStoreMock.	S
TU95	verifica se la funzione numeroAreeisLoading restituisce false, poiché numeroAree non è in stato di caricamento (isLoading è false).	S
TU96	verifica se la funzione areeLimit restituisce correttamente la lista delle aree simulate, che è stata impostata come aree nella simulazione areeStoreMock.	S
TU97	verifica se la funzione areeLimitisLoading restituisce true, poiché areeLimit è in stato di caricamento (isLoading è true).	S
TU98	verifica se la funzione guastiNumber restituisce il numero corretto di guasti, che è impostato come 5 nella simulazione guastiStoreMock.	S
TU99	verifica se la funzione guastiNumberisLoading restituisce false, poiché guastiNumber non è in stato di caricamento (isLoading è false).	S
TU100	verifica se la funzione guasti restituisce correttamente la lista dei guasti simulati, che è stata impostata come guasti nella simulazione guastiStoreMock.	S
TU101	verifica se la funzione guastiisLoading restituisce false, poiché la lista dei guasti non è in stato di caricamento (isLoading è false).	S
TU102	verifica se la funzione lampioniNumber restituisce il numero corretto di lampioni, che è impostato come 3 nella simulazione lampioniStoreMock.	S
TU103	verifica se la funzione lampioniisLoading restituisce false, poiché il numero di lampioni non è in stato di caricamento (isLoading è false).	S
TU104	verifica se la funzione sensoriNumber restituisce il numero corretto di sensori, che è impostato come 7 nella simulazione sensoriStoreMock.	S

ID Test	Descrizione	Stato
TU105	verifica se la funzione sensoriisLoading restituisce false, poiché il numero di sensori non è in stato di caricamento (isLoading è false).	S
TU106	verifica se la funzione guastiAperti chiama correttamente getGuastiAperti dallo store dei guasti.	S
TU107	verifica se la funzione guastiChiusi restituisce un array vuoto e chiama correttamente getGuastiChiusi dallo store dei guasti.	S
TU108	verifica se la funzione isLoading restituisce false.	S
TU109	verifica se la funzione areaDetails chiama correttamente getAreaDetails dallo store delle aree.	S
TU110	verifica se la funzione dettagliLampione chiama correttamente getdettagliLampioni dallo store dei lampioni.	S
TU111	verifica se la funzione listaLampioni chiama correttamente getlistaLampioni dallo store dei lampioni.	S
TU112	verifica se la funzione isLoading chiama correttamente getlistaLampioni dallo store dei lampioni.	S
TU113	verifica se la funzione accendiLampione chiama correttamente accendiLampioneMutation.mutateAsync dallo store dei lampioni con gli argomenti corretti.	S
TU114	verifica se la funzione spegniLampione chiama correttamente spegniLampioneMutation.mutateAsync dallo store dei lampioni con gli argomenti corretti.	S
TU115	verifica se la funzione listaSensori chiama correttamente getlistaSensori dallo store dei sensori.	S
TU116	verifica se la funzione isLoading chiama correttamente getlistaSensori dallo store dei sensori.	S
TU117	verifica se la funzione submit del LoginViewModel funziona correttamente.	S
TU118	verifica se il ViewModel restituisce tutti i metodi e le variabili attesi.	S
TU119	verifica se i metodi del negozio (getAreaDetails e modificaAreaMutation.mutateAsync) vengono chiamati correttamente quando i metodi del ViewModel (areaDetails, submit, ecc.) vengono invocati.	S
TU120	verifica se i metodi del negozio (getAreaDetails e modificaAreaMutation.mutateAsync) vengono chiamati correttamente quando i metodi del ViewModel (areaDetails, submit, ecc.) vengono invocati.	S
TU121	erifica se il ViewModel restituisce tutti i metodi e le variabili attesi.	S

ID Test	Descrizione	Stato
TU122	verifica se i metodi del negozio (getGuastoDetails e modificaGuastoMutation.mutateAsync) vengono chiamati correttamente quando i metodi del ViewModel (guastoDetails, submit, ecc.) vengono invocati.	S
TU123	verifica se il ViewModel gestisce correttamente gli errori durante l'invio del modulo.	S
TU124	erifica se il ViewModel restituisce tutti i metodi e le variabili attesi.	S
TU125	verifica se i metodi del negozio (getdettagliSensori, modificaSensoreMutation.mutateAsync, eliminaSensoreMutation.mutateAsync) vengono chiamati correttamente quando i metodi del ViewModel (sensoreDetails, submit, eliminaSensore, ecc.) vengono invocati.	S
TU126	verifica se il ViewModel gestisce correttamente gli errori durante l'invio del modulo.	S

4.4.2 Tracciamento test di unità - Frontend

ID Test	Metodo
TU01	LumosMinima\client\src__test__\View\AggiungiAreaView.test.tsx: test('renders AggiungiAreaView component correctly', () =>{
TU02	LumosMinima\client\src__test__\View\AggiungiAreaView.test.tsx: test('renders error message when submitIsError is true', () =>{
TU03	LumosMinima\client\src__test__\View\AggiungiAreaView.test.tsx: test('clears error onFocus', () =>{
TU04	LumosMinima\client\src__test__\View\AggiungiGuastoView.test.tsx: test('renders AggiungiGuastoView component correctly', () =>{
TU05	LumosMinima\client\src__test__\View\AggiungiGuastoView.test.tsx: test('renders error message when submitIsError is true', () =>{
TU06	LumosMinima\client\src__test__\View\AggiungiGuastoView.test.tsx: test('renders correctly if areaDetails data is undefined', () =>{
TU07	LumosMinima\client\src__test__\View\AggiungiLampione.test.tsx: test('renders AggiungiLampioneView component correctly', () =>{
TU08	LumosMinima\client\src__test__\View\AggiungiLampione.test.tsx: test('renders error message when submitIsError is true', () =>{
TU09	LumosMinima\client\src__test__\View\AggiungiSensoreView.test.tsx: test('renders AggiungiSensoreView component correctly', () =>{
TU10	LumosMinima\client\src__test__\View\AggiungiSensoreView.test.tsx: test('handles form submission with error', () =>{
TU11	LumosMinima\client\src__test__\View\AggiungiSensoreView.test.tsx: test('handles form input changes', () =>{
TU12	LumosMinima\client\src__test__\View\AggiungiSensoreView.test.tsx: test('calls clearError on form focus', () =>{
TU13	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('renders correctly when loading', () =>{
TU14	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('renders correctly without error', () =>{
TU15	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Accendi Lampioni Area" button click', () =>{

ID Test	Metodo
TU16	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Aumenta Luminosità" button click', () =>{
TU17	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Diminuisci Luminosità" button click', () =>{
TU18	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Spegni Lampioni Area" button click', () =>{
TU19	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Cambia Modalità" button click', () =>{
TU20	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles "Elimina Area" button click', () =>{
TU21	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles error state correctly', () =>{
TU22	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('stato acceso', () =>{
TU23	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('modalità automatica', () =>{
TU24	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('handles submit error correctly', () =>{
TU25	LumosMinima\client\src__test__\View\AreaDetailsView.test.tsx: it('renders correctly data undefined', () =>{
TU26	LumosMinima\client\src__test__\View\AreeView.test.tsx: test('renders AreeView component correctly', () =>{
TU27	LumosMinima\client\src__test__\View\AreeView.test.tsx: test('renders IsLoading', () =>{
TU28	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('renders GuastoDetailsView component correctly', () =>{
TU29	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('displays loading state', () =>{
TU30	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('displays error state', () =>{
TU31	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('clicking the "Chiudi Guasto" button calls chiudiGuasto function', () =>{
TU32	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('renders nothing when stato 1', () =>{
TU33	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('renders component when stato is not 1', () =>{
TU34	LumosMinima\client\src__test__\View\GuastoDetailsView.test.tsx: test('guastoDetails data undefined', () =>{
TU35	LumosMinima\client\src__test__\View\HomeView.test.tsx: test('renders HomeView component correctly', () =>{
TU36	LumosMinima\client\src__test__\View\HomeView.test.tsx: test('displays loading states when data is loading', () =>{
TU37	LumosMinima\client\src__test__\View\HomeView.test.tsx: test('displays "Logout" button and calls logout function on button click', async () =>{
TU38	LumosMinima\client\src__test__\View\HomeView.test.tsx: test('renders mapped guasti elements correctly', () =>{
TU39	LumosMinima\client\src__test__\View\ListaGuastiView.test.tsx: it('renders ListaGuastiView component correctly', () =>{
TU40	LumosMinima\client\src__test__\View\ListaGuastiView.test.tsx: it('renders isLoading', () =>{
TU41	LumosMinima\client\src__test__\View\ListaGuastiView.test.tsx: it('renders correctly guastiAperti undefined and guastiChiusi undefined', () =>{
TU42	LumosMinima\client\src__test__\View\ListaLampioniView.test.tsx: it('should render the component correctly', () =>{

ID Test	Metodo
TU43	LumosMinima\client\src__test__\View\ListaLampioniView.test.tsx: it('should render loading message when isLoading is true', () =>{
TU44	LumosMinima\client\src__test__\View\ListaLampioniView.test.tsx: it('should handle spegniLampione function when "Spegni Lampione" button is clicked', () =>{
TU45	LumosMinima\client\src__test__\View\ListaLampioniView.test.tsx: it('should handle accendiLampione function when "Accendi Lampione" button is clicked', () =>{
TU46	LumosMinima\client\src__test__\View\ListaLampioniView.test.tsx: it('renders correctly listaLampioni undefined and Areadetails data undefined', () =>{
TU47	LumosMinima\client\src__test__\View\ListaSensoriView.test.tsx: test('renders ListaSensoriView component correctly', () =>{
TU48	LumosMinima\client\src__test__\View\ListaSensoriView.test.tsx: test('renders isLoading', () =>{
TU49	LumosMinima\client\src__test__\View\LoginView.test.tsx: test('renders LoginView component correctly', () =>{
TU50	LumosMinima\client\src__test__\View\ModificaAreaView.test.tsx: it('renders correctly', () =>{
TU51	LumosMinima\client\src__test__\View\ModificaAreaView.test.tsx: it('submits the form with updated values', () =>{
TU52	LumosMinima\client\src__test__\View\ModificaAreaView.test.tsx: it("no data in areaDetails", () =>{
TU53	LumosMinima\client\src__test__\View\ModificaGuastoView.test.tsx: it('renders correctly', () =>{
TU54	LumosMinima\client\src__test__\View\ModificaGuastoView.test.tsx: it('submits the form with updated values', () =>{
TU55	LumosMinima\client\src__test__\View\ModificaGuastoView.test.tsx: it('stato 1', () =>{
TU56	LumosMinima\client\src__test__\View\ModificaGuastoView.test.tsx: it("no data in guastoDetails", () =>{
TU57	LumosMinima\client\src__test__\View\ModificaLampioneView.test.tsx: it('renders correctly', () =>{
TU58	LumosMinima\client\src__test__\View\ModificaLampioneView.test.tsx: it('submits the form with updated values', () =>{
TU59	LumosMinima\client\src__test__\View\ModificaLampioneView.test.tsx: it('handles form submission error', () =>{
TU60	LumosMinima\client\src__test__\View\ModificaLampioneView.test.tsx: it('calls eliminaLampione when "Elimina Lampione" button is clicked', () =>{
TU61	LumosMinima\client\src__test__\View\ModificaLampioneView.test.tsx: it("no data in lampioneDetails", () =>{
TU62	LumosMinima\client\src__test__\View\ModificaSensoreView.test.tsx: it('renders correctly', () =>{
TU63	LumosMinima\client\src__test__\View\ModificaSensoreView.test.tsx: it('submits the form with updated values', () =>{
TU64	LumosMinima\client\src__test__\View\ModificaSensoreView.test.tsx: it('handles form submission error', () =>{
TU65	LumosMinima\client\src__test__\View\ModificaSensoreView.test.tsx: it('clicks the "Elimina Sensore" button', () =>{
TU66	LumosMinima\client\src__test__\View\ModificaSensoreView.test.tsx: it("no data in sensoreDetails", () =>{
TU67	LumosMinima\client\src__test__\ViewModel\AggiungiAreaViewModel.test.tsx: it('should call mutateAsync when submitting', async () =>{
TU68	LumosMinima\client\src__test__\ViewModel\AggiungiAreaViewModel.test.tsx: it('should set submitHasError to true when there is an error', () =>{
TU69	LumosMinima\client\src__test__\ViewModel\AggiungiAreaViewModel.test.tsx: it('should set submitError when there is an error', async () =>{

ID Test	Metodo
TU70	LumosMinima\client\src__test__\ViewModel\AggiungiGuastoViewModel.test.tsx: it('should return the expected ViewModel', () =>{
TU71	LumosMinima\client\src__test__\ViewModel\AggiungiGuastoViewModel.test.tsx: it('should call store methods when ViewModel functions are invoked', async () =>{
TU72	LumosMinima\client\src__test__\ViewModel\AggiungiGuastoViewModel.test.tsx: it("should setSubmitError and setSubmitHasError", async () =>{
TU73	LumosMinima\client\src__test__\ViewModel\AggiungiSensoreViewModel.test.tsx: it('should return the expected ViewModel', () =>{
TU74	LumosMinima\client\src__test__\ViewModel\AggiungiSensoreViewModel.test.tsx: it('should call store methods when ViewModel functions are invoked', async () =>{
TU75	LumosMinima\client\src__test__\ViewModel\AggiungiSensoreViewModel.test.tsx: it("should setSubmitError and setSubmitHasError", async () =>{
TU76	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call getAreaDetails with the correct ID', () =>{
TU77	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call getAreaDetails and isLoading true', () =>{
TU78	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call getAreaDetails and return isError false', () =>{
TU79	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call aumentaLuminositàMutation if conditions are met', () =>{
TU80	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should not call aumentaLuminositàMutation if luminosità is already at max', () =>{
TU81	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call diminuisciLuminosità if conditions are met', () =>{
TU82	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should not call aumentaLuminositàMutation if luminosità is already at min', () =>{
TU83	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call eliminaAreaMutation when eliminaArea is called', () =>{
TU84	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call cambiaModalitaMutation when cambiaModalità is called', () =>{
TU85	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call accendiAreaMutation when accendiArea is called', () =>{
TU86	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call accendiLampioniAreaMutation when accendiLampioniArea is called', () =>{
TU87	LumosMinima\client\src__test__\ViewModel\AreaDetailsViewModel.test.tsx: it('should call spegniLampioniAreaAreaMutation when spegniLampioniArea is called', () =>{
TU88	LumosMinima\client\src__test__\ViewModel\AreeViewModel.test.tsx: it('should return aree', () =>{
TU89	LumosMinima\client\src__test__\ViewModel\AreeViewModel.test.tsx: it('should return isLoading', () =>{
TU90	LumosMinima\client\src__test__\ViewModel\AreeViewModel.test.tsx: it('should call accendiAllAreeMutation.mutateAsync with expected arguments', () =>{
TU91	LumosMinima\client\src__test__\ViewModel\AreeViewModel.test.tsx: it('should call spegniAllAreeMutation.mutateAsync with expected arguments', () =>{
TU92	LumosMinima\client\src__test__\ViewModel\GuastoDetailsViewModel.test.tsx: it('should call getGuastoDetails with the correct ID', () =>{
TU93	LumosMinima\client\src__test__\ViewModel\GuastoDetailsViewModel.test.tsx: it('should call chiudiGuastoMutation and navigate when chiudiGuasto is called', async () =>{
TU94	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return numeroAree", () =>{
TU95	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return numeroAreeisLoading", () =>{
TU96	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return areeLimit", () =>{

ID Test	Metodo
TU97	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return areeLimitisLoading", () =>{
TU98	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return guastiNumber", () =>{
TU99	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return guastiNumberisLoading", () =>{
TU100	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return guasti", () =>{
TU101	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return guastiisLoading", () =>{
TU102	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return lampioniNumber", () =>{
TU103	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return lampioniisLoading", () =>{
TU104	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return sensoriNumber", () =>{
TU105	LumosMinima\client\src__test__\ViewModel\HomeViewModel.test.tsx: it("should return sensoriisLoading", () =>{
TU106	LumosMinima\client\src__test__\ViewModel\ListaGuastiViewModel.test.tsx: it('should return guastiAperti', () =>{
TU107	LumosMinima\client\src__test__\ViewModel\ListaGuastiViewModel.test.tsx: it('should return guastiChiusi', () =>{
TU108	LumosMinima\client\src__test__\ViewModel\ListaGuastiViewModel.test.tsx: it('should return isLoading', () =>{
TU109	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should return areaDetails', () =>{
TU110	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should return dettagliLampione', () =>{
TU111	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should return listaLampioni', () =>{
TU112	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should return isLoading', () =>{
TU113	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should call accendiLampione with expected arguments', () =>{
TU114	LumosMinima\client\src__test__\ViewModel\ListaLampioniViewModel.test.tsx: it('should call spegniLampione with expected arguments', () =>{
TU115	LumosMinima\client\src__test__\ViewModel\ListaSensoriViewModel.test.tsx: it('should return listaSensori', () =>{
TU116	LumosMinima\client\src__test__\ViewModel\ListaSensoriViewModel.test.tsx: it('should return isLoading', () =>{
TU117	LumosMinima\client\src__test__\ViewModel\LoginViewModel.test.tsx: it("should submit login form", async () =>{
TU118	LumosMinima\client\src__test__\ViewModel\ModificaAreaViewModel.test.tsx: it('should return the expected ViewModel', () =>{
TU119	LumosMinima\client\src__test__\ViewModel\ModificaAreaViewModel.test.tsx: it('should call store methods when ViewModel functions are invoked', async () =>{
TU120	LumosMinima\client\src__test__\ViewModel\ModificaAreaViewModel.test.tsx: it("should setSubmitError and setSubmitHasError", async () =>{
TU121	LumosMinima\client\src__test__\ViewModel\ModificaGuastoViewModel.test.tsx: it('should return the expected ViewModel', () =>{
TU122	LumosMinima\client\src__test__\ViewModel\ModificaGuastoViewModel.test.tsx: it('should call store methods when ViewModel functions are invoked', async () =>{
TU123	LumosMinima\client\src__test__\ViewModel\ModificaGuastoViewModel.test.tsx: it("should setSubmitError and setSubmitHasError", async () =>{

ID Test	Metodo
TU124	LumosMinima\client\src__test__\ViewModel\ModificaSensoreViewModel.test.tsx: it('should return the expected ViewModel', () =>{
TU125	LumosMinima\client\src__test__\ViewModel\ModificaSensoreViewModel.test.tsx: it('should call store methods when ViewModel functions are invoked', async () =>{
TU126	LumosMinima\client\src__test__\ViewModel\ModificaSensoreViewModel.test.tsx: it("should setSubmitError and setSubmitHasError", async () =>{

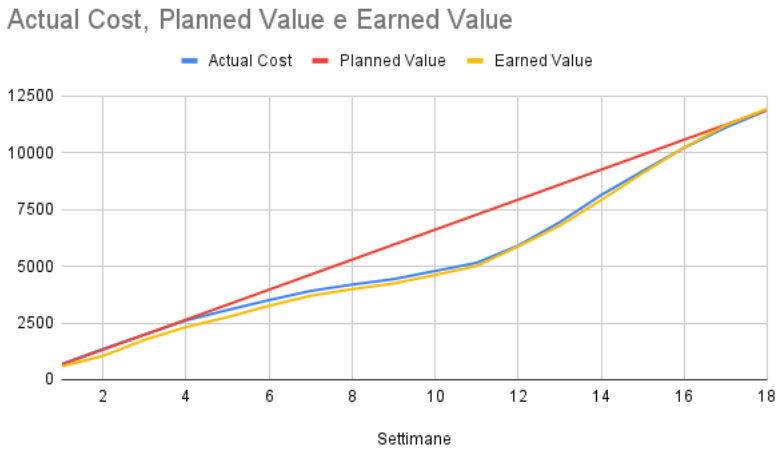
5 Applicazione e valutazione delle metriche

I grafici sono frutto di un foglio di calcolo creato dal gruppo che applica le formule per il calcolo delle metriche definite in questo documento.

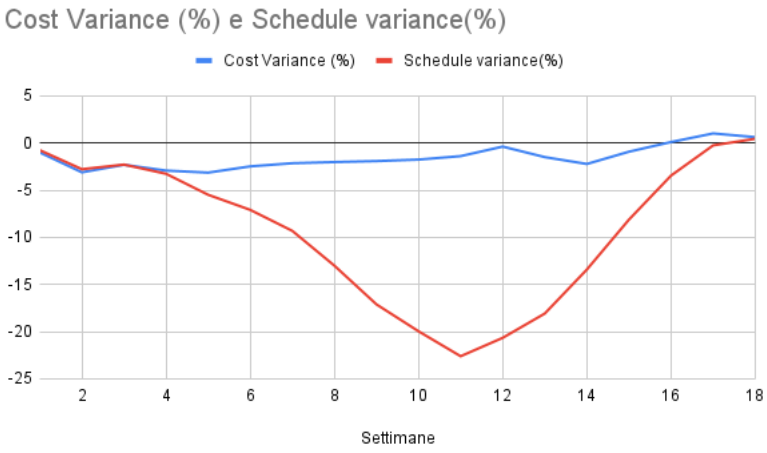
5.1 Valutazione d'insieme (Qualità di processo)

Il lavoro è proseguito secondo le aspettative, la mancanza di impegni nella fase di PB dei membri del gruppo ha permesso un aumento di ore dedicate al progetto. Ciò ha permesso , dopo una prima fase RTB rallentata dagli impegni personali del gruppo, di recuperare il tempo perso e rientrare all'interno delle soglie di accettabilità e raggiungere valori ottimali in quasi tutte le valutazioni. Questo è particolarmente evidente nei grafico "Cost Variance e Schedule Variance" e nel grafico "Planning Value, Actual Cost e Earned Value" dove successivamente al undicesimo sprint, che ha segnato la fine del RTB, il lavoro svolto ha subito un notevole incremento della qualità.

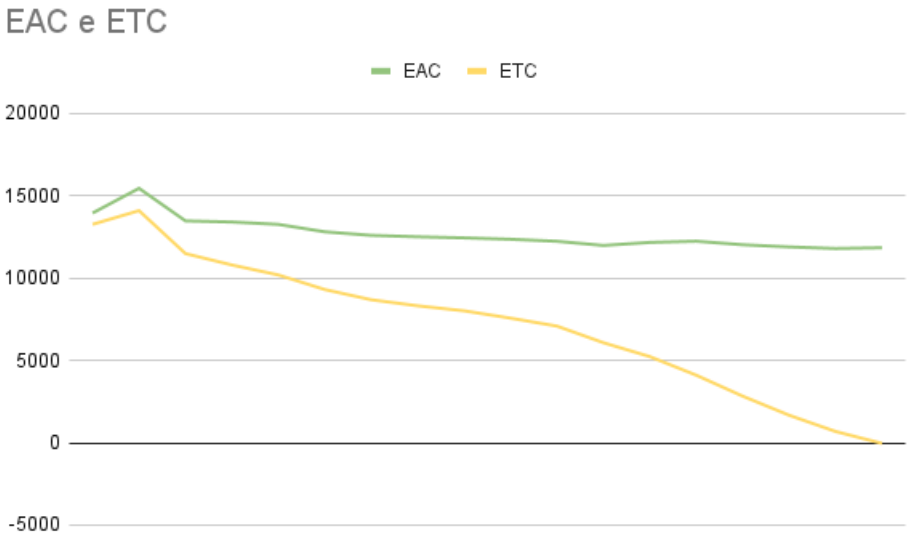
5.2 Planning Value, Actual Cost e Earned Value



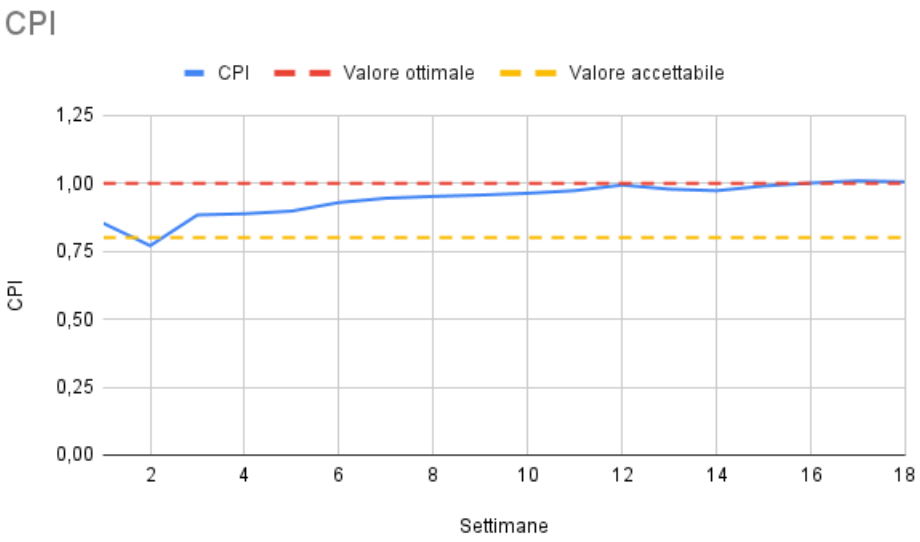
5.3 Cost Variance e Schedule Variance



5.4 Eastimate at completition e Estimate to Complete



5.5 Cost Performance Index

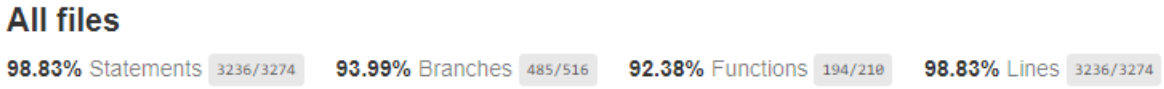


5.6 Indice di Gulpease

Valutazione indice di Gulpease				
Documenti	**Numero di righe**	**Numero di parole**	**Numero lettere**	**Indice di Gulpease**
analisi dei requisiti	1441	6077	26723	100
manuale utente	333	1553	5731	100
norme di progetto	1143	6556	27520	99
piano di progetto	3707	8798	29084	100
specifica tecnica	601	1602	7605	100
piano di qualifica	1924	6350	35510	100
glossario	168	714	3979	100

5.7 Verifica del Codice

5.7.1 Front-end



L'immagine sovrastante riporta rispettivamente la Statement, Branch, Function e Code coverage **complessiva** ottenuta sulla parte di front-end del prodotto

File		Statements		Branches		Functions		Lines	
View/Model		97.82%	675/690	93.61%	132/141	94%	94/100	97.82%	675/690
pages		100%	224/224	100%	16/16	100%	0/0	100%	224/224
stores		97.37%	817/839	97.47%	116/119	87.8%	72/82	97.37%	817/839
view		99.93%	1520/1521	92.08%	221/240	100%	28/28	99.93%	1520/1521

L'immagine sovrastante riporta rispettivamente la Statement, Branch, Function e Code coverage ottenuta sui singoli file della parte di front-end del prodotto

5.7.2 Back-end

All files

86.36% Statements `1920/2223` **80%** Branches `232/290` **93.54%** Functions `116/124` **86.36%** Lines `1920/2223`

L'immagine sovrastante riporta rispettivamente la Statement, Branch, Function e Code coverage **complessiva** ottenuta sulla parte di back-end del prodotto

File		Statements		Branches		Functions		Lines	
server		100%	38/38	100%	5/5	100%	3/3	100%	38/38
server/controllers		91.72%	743/810	76.5%	127/166	93.47%	43/46	91.72%	743/810
server/models		100%	238/238	100%	17/17	100%	11/11	100%	238/238
server/routes		100%	70/70	100%	10/10	100%	5/5	100%	70/70
server/services		77.88%	831/1067	79.34%	73/92	91.52%	54/59	77.88%	831/1067

L'immagine sovrastante riporta rispettivamente la Statement, Branch, Function e Code coverage ottenuta sui singoli file della parte di back-end del prodotto