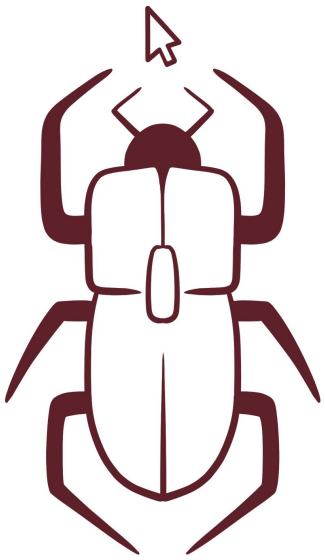

# Skarab Group

---

## Design Thinking

---

skarabswegroup@gmail.com

# Versionamento e changelog

| Data Modifica | Versione | Descrizione Modifica | Redattore | Verificatore |
|---|---|---|---|---|
| 2025-11-26 | 1.1.0 | Rimossa tabella Azioni e Responsabilità \| Correzione errori | Suar Alberto | Basso Kevin |
| 2025-11-25 | 1.0.0 | Prima stesura del documento | Suar Alberto | Basso Kevin |

# Indice

# 1 Informazioni generali

## 1.1 Apertura

La riunione, svolta nella sede di Var Group SpA, ha avuto inizio alle 14:05, con i referenti dell'azienda che hanno esposto i principali punti del giorno.

## 1.2 Presenti e assenti

| | |
|---|---|
| **Presenti:** | Suar Alberto, Zago Alice, Basso Kevin, Berengan Riccardo, Sandu Antonio, Sgreva Andrea |
| **Assenti:** | Martinello Riccardo |
| **Referenti Aziendali:** | Dindo Stefano, Massaro Michele, Pivetta Federico |

# 2 Ordine del giorno

1. Allineamento sullo scopo del progetto
2. Definizione delle Personas utilizzatrici del prodotto
3. Definizione delle principali funzionalità in relazione alle Personas
4. Scelta del modello di riferimento per la creazione degli agenti
5. Costruzione di un diagramma di flusso del prodotto

# 3 Svolgimento della Riunione

## 3.1 Punto 1 — Allineamento sullo scopo del progetto

### Sintesi

Prima dell'inizio effettivo della sessione di design thinking, l'azienda proponente ha chiesto di definire, in linea generale, lo scopo del progetto *CodeGuardian*, così da chiarire eventuali dubbi relativi alla presentazione del capitolato e strutturare una base di partenza.

### Decisione

Il team ha discusso e condiviso le proprie opinioni riguardo allo scopo del progetto *CodeGuardian*, giungendo alla conclusione che consista in **una piattaforma basata su un sistema ad agenti, capace di analizzare repository e fornire report dettagliati** relativi a controlli del codice — come analisi statica e copertura dei test — **controlli di sicurezza**, come la verifica del rispetto degli standard OWASP, e **controlli sulla documentazione**, verificando la presenza di documenti chiari e completi che descrivano accuratamente le funzionalità del prodotto analizzato.

## 3.2 Punto 2 — Definizione delle Personas utilizzatrici del prodotto

**Sintesi**

L'azienda ha introdotto come argomento di discussione la definizione delle Personas di *CodeGuardian*, ovvero le tipologie di utenti che il team ritiene essere maggiormente inclini a utilizzare il prodotto.

**Decisione**

A seguito di un'analisi inizialmente individuale e successivamente condivisa, sono emerse come principali Personas i seguenti ruoli:

- Sviluppatori Software
- Responsabili di Progetto
- Consulenti informatici

Per la prima fase del progetto, il team si concentrerà sulla **definizione di numerosi casi d'uso e requisiti** che rappresentino le necessità e le aspettative tipiche di queste categorie di utenti.

## 3.3 Punto 3 — Definizione delle principali funzionalità in relazione alle Personas

**Sintesi**

Sulla base delle Personas individuate come principali utilizzatrici del prodotto, l'azienda ha richiesto di focalizzarsi sulla definizione delle funzionalità che ciascuna figura professionale vorrebbe trovare in *CodeGuardian*.

**Decisione**

Per quanto riguarda le funzionalità considerate desiderabili da un **Software Developer**, sono emersi requisiti di questo tipo:

- I **tempi di risposta** devono essere **ragionevoli** e, qualora si verificassero problemi durante l'elaborazione, l'utente deve essere avvisato tempestivamente, possibilmente con suggerimenti su come procedere.
- Deve essere mostrato l'esito dei test eseguiti, evidenziando quelli eventualmente falliti.
- Devono essere forniti, tramite **dashboard e rappresentazioni grafiche**, indicatori che mostrino il miglioramento (o il peggioramento) rispetto alle versioni precedenti, **specificando** inoltre quali **best practices** non sono state rispettate nello sviluppo delle nuove funzionalità.

Per quanto riguarda le funzionalità desiderabili da un **Responsabile di Progetto**, sono stati evidenziati i seguenti requisiti:

- Il sistema deve verificare il rispetto degli **standard di sicurezza**, con particolare attenzione alle linee guida **OWASP**, segnalando eventuali vulnerabilità e suggerendo possibili azioni correttive.
- Devono essere disponibili confronti chiari tra versioni, mostrando in modo sintetico il **miglioramento o peggioramento** del progetto nel tempo, sia in termini di qualità del codice sia di sicurezza.
- Il sistema deve controllare che la **documentazione sia coerente con il comportamento effettivo del codice** e viceversa, identificando eventuali discrepanze.
- Devono essere fornite valutazioni di alto livello sull'**aderenza del team alle best practices**, con particolare attenzione a mantenibilità, qualità del design e conformità alle linee guida del progetto.

Per quanto riguarda le funzionalità desiderabili da un **Consulente Informatico**, sono state individuate esigenze aggiuntive legate all'analisi approfondita, alla sicurezza dei dati e alla valutazione complessiva del progetto:

- Il sistema deve offrire una **analisi storica dettagliata** del repository, permettendo di comprendere l'evoluzione del progetto, del codice e della documentazione nel tempo.
- Deve essere garantita la **protezione dei dati** forniti alla piattaforma, in particolare nel caso di utilizzo di modelli linguistici (LLM). Il sistema deve assicurare che le informazioni del repository siano trattate nel rispetto di criteri di sicurezza, privacy e riservatezza.
- Devono essere mostrati chiaramente gli **standard adottati** dal progetto (ad esempio sicurezza, formattazione, architettura), evidenziando le versioni obsolete o non più raccomandate.
- Il sistema deve indicare la presenza di **test mancanti o insufficienti**, segnalando l'impatto potenziale in termini di rischio.
- Devono essere forniti indicatori chiari sul **miglioramento o peggioramento** tra versioni diverse, includendo aspetti come sicurezza, qualità del codice e coerenza documentale.
- Deve essere possibile ottenere una valutazione complessiva sul livello di **compliance del progetto**, utile per supportare audit tecnici o consulenze strategiche.

## 3.4 Punto 4 — Scelta del modello di riferimento per la creazione degli agenti

**Sintesi**

Data la natura del progetto, fortemente basata sull'impiego di agenti AI, l'azienda ha introdotto i due principali modelli architetturali utilizzati per strutturare flussi di gestione delle richieste. Tali modelli definiscono il modo in cui gli agenti devono collaborare per fornire risposte chiare, coerenti e affidabili all'utente finale. I modelli presentati sono la **Rete Democratica di Agenti** e l'**Orchestratore Autocratico**.

A seguito di questa introduzione, il team è stato incaricato di analizzare entrambi i modelli, evidenziandone vantaggi e svantaggi, per poi scegliere quello più adeguato agli obiettivi del progetto.

**Decisione**

Il team ha deciso di adottare un approccio basato sull'**Orchestratore Autocratico**. In questo modello è presente un agente principale (leader) che coordina il lavoro degli altri agenti: definisce i compiti che ciascun agente deve svolgere, fornisce gli strumenti necessari alla loro esecuzione e raccoglie i risultati prodotti, decidendo di volta in volta quale agente attivare successivamente. A livello strutturale, si può visualizzarlo come un albero in cui ogni agente è orchestratore di uno o più sottoagenti specializzati in modo da poter utilizzare tool ad hoc, al fine di eseguire minuziosamente la richiesta.
Le motivazioni principali che hanno portato a preferire l'approccio dell'orchestratore rispetto alla rete democratica degli agenti sono le seguenti:

- **Maggiore controllo del flusso di esecuzione**: la presenza di un agente coordinatore consente di mantenere un flusso chiaro e prevedibile, fondamentale per un sistema che deve eseguire analisi complesse come test coverage, controlli di sicurezza e verifica della documentazione.
- **Riduzione dell'ambiguità nelle risposte**: in una rete democratica gli agenti possono fornire risposte divergenti, rendendo necessario un ulteriore livello di conciliazione. L'approccio autocratico garantisce invece una risposta coerente e univoca.
- **Facilità di estensione**: aggiungere nuovi agenti specializzati risulta più semplice, poiché basta inserirli nella pipeline coordinata dall'orchestratore senza modificare le logiche di negoziazione tipiche di una rete democratica.
- **Maggiore affidabilità nella gestione degli errori**: l'orchestratore può monitorare l'esecuzione dei singoli agenti e gestire eventuali problemi, ad esempio fornendo fallback o notificando tempestivamente l'utente.
- **Coerenza con gli obiettivi del progetto**: CodeGuardian richiede una struttura rigorosa e verificabile dei processi di analisi; la presenza di un coordinatore centralizzato contribuisce a mantenere un comportamento stabile e riproducibile.

## 3.5 Punto 5 — Costruzione di un diagramma di flusso del prodotto

**Sintesi**

Come ultimo aspetto analizzato durante questa prima sessione di design thinking, l'azienda ha richiesto la definizione di un diagramma di flusso che rappresentasse un'interazione completa con *CodeGuardian*, dall'inserimento di una nuova repository fino alla generazione dei risultati finali. Il team, attraverso l'uso di post-it colorati e una discussione collaborativa, ha quindi costruito una prima versione del flusso logico del prodotto.

**Decisione**

Dalla costruzione del diagramma di flusso è emerso un processo articolato, strutturato in più fasi e orchestrato da un agente centrale. Il flusso individuato può essere sintetizzato nei seguenti passaggi principali:

1. **Inserimento di una nuova repository** L'utente fornisce i riferimenti della repository da analizzare. In questa fase vengono inoltre raccolte informazioni preliminari, come eventuali vincoli, obiettivi o punti di attenzione richiesti dall'azienda.

2. **Richiesta dei punti da analizzare** L'utente specifica quali categorie di controlli desidera svolgere (analisi del codice, test, sicurezza, documentazione, ecc.).

3. **Attivazione dell'Orchestratore** L'orchestratore rappresenta il nodo centrale del flusso: riceve la richiesta, la scompone in attività distinte e definisce quali agenti coinvolgere e in quale ordine. Da questo punto si sviluppano due grandi rami paralleli:

   - il ramo relativo al **Codice**
   - il ramo relativo alla **Documentazione**

4. **Analisi del Codice** Il ramo del codice è suddiviso in più sotto-moduli:

   - **Analisi statica** (tramite tool esterni)
   - **Aggiornamento delle librerie** (verifica dipendenze, versioni obsolete, potenziali vulnerabilità)
   - **Copertura dei test** (interpretazione dei risultati, individuazione test mancanti)
   - **Controlli di sicurezza OWASP** attraverso strumenti specifici

   Ogni modulo produce un risultato parziale che viene inviato all'orchestratore.

5. **Analisi della Documentazione** Questo ramo include tre tipologie di analisi:

   - **Analisi sintattica**
   - **Analisi semantica**
   - **Analisi del rispetto delle best practices e standard aziendali** (coerenza tra ciò che la documentazione dichiara e ciò che il codice effettivamente realizza)

6. **Raccolta e consolidamento dei risultati** L'orchestratore raccoglie gli output prodotti dagli agenti, li sintetizza e li prepara per la fase successiva.

7. **Eventuale intervento del modulo di Remediation** Nel caso siano individuati errori, vulnerabilità o pratiche scorrette, l'agente di *Remediation* suggerisce possibili soluzioni, fix o azioni correttive.

8. **Salvataggio nel database e visualizzazione** I risultati finali vengono salvati nel database e resi disponibili all'utente tramite dashboard, grafici e report dettagliati.

Il diagramma costruito rappresenta quindi una prima versione dell'architettura operativa di *CodeGuardian*, sul quale il team potrà iterare nelle sessioni successive di progettazione.

# Chiusura

La riunione si è conclusa alle 18:00. In concordanza con l'azienda, è stato definito un ambiente Jira per la comunicazione asincrona in sostituzione alle email ed è stata stabilita una cadenza quindicinale per lo svolgimento delle riunioni di allineamento.

**Responsabile Team Skarab:**

Suar Alberto

**Referente Aziendale:**

Michele Moraro

Var Group S.p.A