

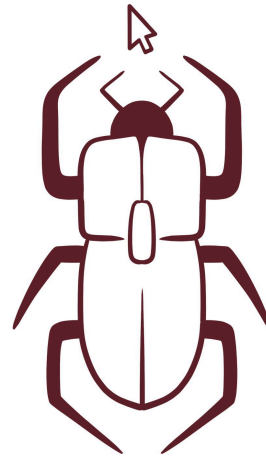
Presentazione RTB - Skarab Group

Ingegneria del Software - A.A. 2025/2026

Componenti del Team:

Alberto Suar
Kevin Basso
Riccardo Martinello
Alice Zago
Andrea Sgreva
Antonio Sandu
Riccardo Berengan

Contatto: skarabswegroup@gmail.com



CodeGuardian: Funzionalità del Prodotto



L'applicazione **CodeGuardian**, basata su **agenti autonomi intelligenti**, persegue i seguenti obiettivi operativi:

- **Audit Multilivello:** analisi automatizzata delle vulnerabilità di sicurezza, della coerenza del **codice sorgente** e della completezza della **documentazione** di progetto.
- **Reporting Centralizzato:** dashboard intuitiva per fornire una visibilità immediata e centralizzata sullo stato di salute degli asset software.
- **Orchestrazione Intelligente:** coordinamento di agenti specializzati per scalare l'analisi su repository complessi e molteplici pipeline di sviluppo.

Qualora gli obiettivi principali fossero rispettati, il team si impegna a includere come estensione del perimetro:

- **Integrazione CI/CD:** supporto proattivo al ciclo di vita del software tramite l'automazione dei processi su GitHub Actions. Il sistema verrebbe così reso disponibile come **strumento interno di sviluppo**, evolvendo da semplice applicazione a tool integrato nei processi di produzione.
- **Sistema di Remediation:** generazione automatizzata di suggerimenti risolutivi e Pull Request per correggere le vulnerabilità e le incoerenze rilevate, trasformando il tool in un assistente attivo alla manutenzione.

Tecnologie Adottate: NestJS e TypeScript



- **Infrastruttura Standardizzata**
- **Modularità e Scalabilità**
- **Ecosistema Integrato**
- **Documentazione e Supporto**
- **Integrazione Nativa TypeScript**

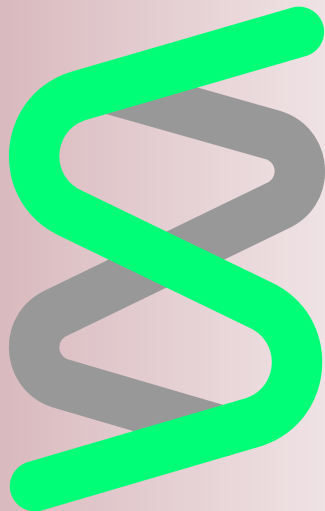
Confronto Tecnologie: NestJS vs ExpressJS



express

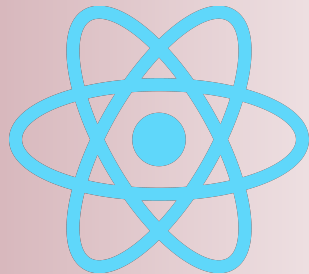
- **Framework vs Libreria**
- **Gestione della Complessità**
- **Sintassi e Decoratori**
- **Infrastruttura Coerente**
- **Risoluzione delle Dipendenze**

Tecnologie adottate: Strands SDK + Python



- Vantaggio Strands vs boto3
 - Gestione automatica dei cicli di reasoning e tool/sub-agent calling
 - Minore codice infrastrutturale e meno parsing delle risposte LLM
- Vantaggio Strands vs Langraph
 - SDK ufficiale AWS perfettamente integrato con i servizi AWS
 - Supporto maggiore per dynamic MCP

Tecnologie adottate: React + TypeScript



- Iterazioni rapide con componenti riutilizzabili
- Virtual DOM: aggiorna solo le parti cambiate
- Ecosistema ricco: recharts, socket.io-client, notifiche

Perché TypeScript con React



- Controllo statico: meno bug a runtime sui payload degli agenti
- Autocompletamento e refactor sicuri in IDE
- Tipi come documentazione implicita per il team

React vs Angular vs Vue



- **Angular**: troppo rigido e over-engineered per la tipologia di progetto
- **Vue**: valido, ma ecosistema più ristretto e meno esperienza nel team
- **React**: flessibile, componentizzato, ecosistema maturo, semplicità d'uso

Tecnologie adottate: MongoDB per persistenza dei dati



- Perché MongoDB:
 - MongoDB è un sistema di persistenza document-oriented
 - È ideale per dati che cambiano spesso o che cambiano forma
 - Permette scalabilità orizzontale

MongoDB vs SQL



- **SQL :**

- Usa strutture rigide e predefinite
- Permette un sistema molto complesso di relazioni
- Non consente alta eterogeneità del contenuto

- **MongoDB :**

- Usa strutture dinamiche che si adattano bene in base a richieste e risultati molto diversi tra loro
- Consente alte prestazioni tramite un semplice sistema di relazioni e indicizzazione