Zaretto Alessandro 11/06/2025

Sviluppo di un manuale d’uso e assistente virtuale per un sito web tramite Intelligenza Artificiale

Sommario

    - Botpress
    - Intercom
- Guida a una possibile implementazione
- Esempio di interazione (caso d’uso)
- Schema architetturale del sistema
- Aggiornamento e manutenzione della knowledge base
- Esempi reali di applicazioni AI per l’assistenza
- Analisi e previsione dei costi totali
- Feedback dei clienti (punti di forza e criticità)
- MVP vs sistemi scalabili
- Sitografia

Parte 1 – Generazione wiki da repository GitHub

Obiettivo della wiki

L'obiettivo è creare una documentazione tecnica chiara, accessibile e manutenibile per un progetto software, a partire direttamente dalla repository su GitHub. Questa documentazione fungerà da guida all’uso, panoramica tecnica e riferimento continuo, anche in assenza di un manuale formale.

Categorie di soluzioni esistenti

Le soluzioni per creare wiki da repository GitHub possono essere suddivise in quattro macro-categorie:

1. **Generatori automatici di documentazione dal codice**
2. **Generatori di siti statici da Markdown**
3. **Wiki modificabili manualmente**
4. **Soluzioni basate su Intelligenza Artificiale (AI)**

Generatori automatici di documentazione dal codice

Molti progetti software utilizzano generatori che analizzano i commenti nel codice e la struttura del progetto per produrre documentazione. I risultati sono pagine HTML, Markdown o PDF facilmente integrabili con GitHub Pages o CI/CD.

- **Doxygen** – supporta C, C++, Java, Python, PHP, C# ecc. Genera HTML, PDF e diagrammi. Ottimo per progetti multi-linguaggio.
    - Pro: ampia compatibilità, output ricco, gratuito
    - Contro: configurazione iniziale complessa
- **Javadoc / JSDoc** – tool standard per Java e JavaScript. Da commenti /** ... */ generano pagine HTML di API reference.
    - Pro: integrazione nativa con l'ecosistema, gratuito
    - Contro: meno flessibili fuori dai linguaggi target
- **Sphinx** – pensato per Python ma estendibile. Usa reStructuredText o Markdown. Output HTML, PDF, ePub. Si integra bene con ReadTheDocs.
    - Pro: potente e flessibile
    - Contro: output meno moderno, richiede tempo di apprendimento
- **DocFX** – per C#/.NET, converte commenti XML in documentazione HTML. Integrabile in pipeline .NET.
    - Pro: output professionale, gratuito
    - Contro: legato all'ecosistema Microsoft
- **Swagger/OpenAPI** – genera UI documentate da file OpenAPI. Ottimo per REST API.
    - Pro: interfacce chiare per API
    - Contro: richiede schemi separati, non legge direttamente il codice

---

Generatori di siti statici da Markdown

Questi strumenti trasformano file Markdown scritti a mano (o da script) in documentazione online. Spesso usati in combinazione con GitHub Pages.

- **Docusaurus** – creato da Facebook, basato su React. Supporta versioning, ricerca e plugin.
    - Pro: moderno, SEO-friendly
    - Contro: richiede conoscenza Node/React
- **MkDocs** – semplice e leggero, usa solo file Markdown. Plugin Material molto diffuso.
    - Pro: facile da avviare, ideale per progetti piccoli
    - Contro: meno estensibile di Docusaurus
- **Read the Docs** – servizio hosting per Sphinx/MkDocs, ricostruisce doc da GitHub ad ogni push.
    - Pro: versioning gratuito, automatizzato
    - Contro: setup iniziale necessario
- **Jekyll** – static site generator in Ruby. Supportato nativamente da GitHub Pages.
    - Pro: semplice, integrato con GitHub
    - Contro: meno orientato alla documentazione tecnica

- **GitBook** – piattaforma SaaS con editor visuale e sincronizzazione GitHub.
    - Pro: moderno e collaborativo
    - Contro: versione gratuita con pubblicità

Wiki modificabili manualmente

Consentono di creare e modificare contenuti liberamente tramite editor web o Markdown.

- **GitHub Wiki** – nativo per ogni repo pubblica. Basato su Markdown, modificabile via Git.
    - Pro: facilissimo da usare
    - Contro: funzionalità limitate
- **Wiki.js** – moderno wiki self-hosted, supporta Markdown e WYSIWYG, sincronizzabile con Git.
    - Pro: flessibile, interfaccia moderna
    - Contro: richiede hosting proprio
- **BookStack** – wiki PHP semplice, con struttura gerarchica.
    - Pro: orientato a manuali interni
    - Contro: meno estendibile
- **MediaWiki/DokuWiki** – piattaforme classiche, molto personalizzabili.
    - Pro: potenti con estensioni
    - Contro: interfaccia datata
- **Confluence** – SaaS Atlassian, ricco di funzionalità collaborative e integrazione con Jira.
    - Pro: completo e professionale
    - Contro: a pagamento

Soluzioni basate su Intelligenza Artificiale

Recentemente sono emersi strumenti che utilizzano LLM (Large Language Models) per generare wiki automaticamente da una repository.

- **DeepWiki** – genera automaticamente una wiki da un repo GitHub pubblico, con grafici e spiegazioni.
    - Pro: documentazione visiva e immediata
    - Contro: non modificabile, solo online
- **OpenRepoWiki** – open-source, genera una wiki scaricabile da repo GitHub usando LLM (es. DeepSeek).
    - Pro: output editabile, gratuito
    - Contro: ancora sperimentale
- **Swimm** – SaaS che crea playbook interattivi sincronizzati con il codice. Supporta onboarding e gestione doc obsoleta.
    - Pro: aggiornamento continuo e live
    - Contro: piani a pagamento, dipende dal cloud

Sintesi comparativa delle soluzioni

Strumento	Tipo	Modificabile	Integrazione GitHub	Costo	Note principali
Doxygen	Estrazione commenti	Sì	Actions / CI	Gratuito	Multi-linguaggio, configurazione complessa
Docusaurus	Markdown statico	Sì	Pages / Actions	Gratuito	Moderno, React-based
GitBook	SaaS wiki	Sì	Git Sync	Freemium	Editor visuale, ads se free
Wiki.js	Wiki self-hosted	Sì	Git integrato	Gratuito	Richiede setup server
DeepWiki	AI auto-generation	No	URL dinamica	Gratuito	Visuale, ma non modificabile

OpenRepoWiki	AI auto-generation	Sì	Manuale o script	Gratuito	Scaricabile, output semplice
Swimm	AI + SaaS	Sì	GitHub Sync	$16+/mese	Alert documentazione obsoleta, onboarding dev

---

Parte 2 – Creazione di assistente IA per un sito (chatbot)

Obiettivo

L'obiettivo è creare un manuale d’uso per un sito. Si vuole realizzare un modello AI, cioè un repository di conoscenze dell'architettura e del funzionamento del programma AC, che possa essere interrogato da un frontend AI per creare un assistente virtuale del programma. Questa soluzione renderebbe l’assistenza al cliente precisa e automatizzata, disponibile in ogni momento. Permetterebbe inoltre di ridurre i tempi di attesa per le risposte e di alleggerire il carico di lavoro umano.

Analisi dell'obiettivo

Per costruire un assistente AI che risponda da un knowledge base specializzato si usano architetture RAG (Retrieval-Augmented Generation) e ricerca semantica su vector embedding. Con le architetture RAG, un modello linguistico di grandi dimensioni non genera le risposte basandosi solo sul suo training generale, ma anche su dati esterni estratti “al volo”.

Si indicizzano i documenti di interesse trasformandoli in vettori tramite un modello di embedding, li si inserisce in un database vettoriale e al momento della domanda si recuperano per costruire una risposta ben contestualizzata. Questa procedura migliora accuratezza e attualità.

I LLM non vanno riaddestrati da zero, richiamano semplicemente informazioni esterne prima di generare la risposta. La ricerca semantica su vettori consente inoltre di scoprire testi

pertinenti anche in mancanza di parole chiave esatte, permettendo ricerche per similarità semantica.

Piattaforme e framework consigliati

Chatbase

Servizio cloud rapido per creare chatbot RAG senza sviluppo, permettendo di caricare manualmente PDF, siti web e testi come knowledge base. Supporta canali multipli (sito web, Slack, WhatsApp) ed è pensata per utenti non tecnici. Limiti nella personalizzazione avanzata e assenza di live chat per passaggio a operatore umano.
Prezzi: piano gratuito limitato (20 messaggi/mese), Hobby $19/mese, Standard $99/mese (incluso GPT-4/Turbo), Illimitato $399/mese.
Pro: setup veloce, user-friendly.
Contro: costi crediti GPT-4, limitazioni funzionali.

LangChain (e LlamaIndex)

Libreria open-source Python per sviluppare pipeline RAG personalizzate. Offre componenti per caricare documenti, chunking, calcolo embedding e ricerca vettoriale. Massima flessibilità, ma richiede competenze di sviluppo e gestione infrastruttura.
Costo: gratuito, ma dipende da costi API LLM (es. OpenAI GPT-4).
Uso: prototipi, MVP, sistemi enterprise con tool di supporto come LangSmith.

Dify AI

Piattaforma open-source per applicazioni LLM basate su agenti e pipeline RAG. Offre workflow visuali, gestione modelli, hosting cloud o self-hosted. Interfaccia drag&drop, facile da usare anche senza codice.
Prezzi: piano sandbox gratuito (200 crediti, 50 documenti), Pro $59/mese, Team $159/mese.
Pro: facilità d’uso, open-source, buona scalabilità.
Contro: piattaforma giovane, ancora in sviluppo.

Botpress

Piattaforma conversazionale open-source e cloud PaaS, focalizzata su chatbot enterprise. Visual builder, moduli NLU, editor JavaScript, forti integrazioni. Supporta knowledge base con import di PDF, siti, tabelle e motore RAG.
Prezzi: gratuito pay-as-you-go, Plus $89/mese, Team $495/mese.
Pro: elevata flessibilità, controllo developer-friendly, buon UI.
Contro: curva di apprendimento, roadmap poco chiara.

Intercom (Fin)

Piattaforma SaaS di supporto clienti con chatbot e knowledge base AI. Automatizza risposte 24/7, integra articoli help center.
Prezzi: da $29/agente/mese + $0,99 per conversazione chiusa.
Pro: potente ambiente all-in-one.
Contro: costi elevati, complessità iniziale, meno focalizzato su RAG custom.

Guida a una possibile implementazione

1. **Raccolta e preparazione dei documenti**
   Ottenere tutta la documentazione rilevante (specifiche, manuali, FAQ, wiki, codice commentato). Convertire documenti in testo (OCR, parsing PDF/HTML). Pulire testo da elementi non pertinenti.

2. **Chunking del testo**
   Suddividere i documenti in “frammenti” di dimensione gestibile per rispettare i limiti di contesto LLM.

3. **Creazione di embeddings**
   Trasformare ogni chunk in un vettore usando un modello di embedding (costo orientativo: ~$0.0004 per 1.000 token con OpenAI).

4. **Indice vettoriale**
   Caricare embedding in un vector store (Faiss, Pinecone, Weaviate) per ricerca rapida per similarità.

5. **Motore di ricerca (retrieval)**
   Convertire la domanda utente in embedding, recuperare i chunk più simili.

6. **Generazione risposta con LLM**
   Usare LLM (es. GPT-4) per generare la risposta basata sui chunk recuperati.

7. **Frontend chat**
   Integrare API backend in interfaccia utente (web chat, widget) per invio query e ricezione risposte.

8. **Deploy e hosting**
   Ospitare su server cloud o on-premise, gestire scalabilità e ridondanza.

9. **Manutenzione e miglioramenti**
   Monitorare conversazioni, aggiornare knowledge base, migliorare risposte.

Esempio di interazione (caso d’uso)

Per mostrare quale sarebbe il funzionamento pratico del sistema, si può simulare una tipica interazione tra utente e assistente AI. Questo esempio illustra il comportamento che dovrebbe avere il modello e il tipo di valore offerto nel contesto del supporto tecnico al software AC.

Esempio di domanda utente:
“Come posso impostare il backup automatico in AC?”

Esempio di risposta dell’assistente AI:
"Per impostare il backup automatico, accedi al pannello amministratore, seleziona la voce ‘Sicurezza’ e poi ‘Backup automatico’. Da qui puoi attivare la pianificazione, scegliere frequenza e destinazione. Per maggiori dettagli consulta il manuale tecnico alla sezione 4.2."

Questo tipo di risposta è possibile grazie al recupero semantico del contesto dai documenti e alla generazione contestualizzata tramite LLM. La presenza di riferimenti specifici ai documenti garantisce trasparenza e tracciabilità, riducendo eventuali rischi e migliorando le prestazioni.

Schema architetturale del sistema

La soluzione proposta si basa su un’architettura RAG, integrata con un frontend conversazionale. Di seguito una rappresentazione semplificata dei componenti principali:

1. **Raccolta dati**: documentazione tecnica, manuali, FAQ, codice commentato.
2. **Preprocessing**: suddivisione dei testi in chunk e pulizia dei contenuti.
3. **Embedding**: ogni chunk viene trasformato in un vettore tramite modello di embedding.
4. **Vector store**: i vettori vengono indicizzati in un database vettoriale.
5. **Retrieval**: a runtime, l'input utente è trasformato in embedding e confrontato con i vettori per recuperare i contenuti più simili.
6. **LLM + Prompting**: i chunk recuperati vengono uniti alla domanda per costruire un prompt, poi passato a un LLM.
7. **Output**: la risposta viene generata e restituita all’utente tramite il frontend conversazionale.

Aggiornamento e manutenzione della knowledge base

Per garantire l’affidabilità nel tempo del sistema e la pertinenza delle risposte, è fondamentale avere una strategia di aggiornamento regolare.

La knowledge base dovrà essere sincronizzata con l’evoluzione del programma AC, soprattutto in caso di modifiche all’interfaccia, alle funzionalità o alle normative di riferimento. Si può prevedere una revisione trimestrale dei contenuti indicizzati, con la possibilità di:

- aggiungere nuovi documenti se quelli vecchi non bastano;
- sostituire le versioni obsolete;
- monitorare le domande poste più frequentemente per individuare lacune informative e ottimizzare la copertura semantica.

In un contesto enterprise, può essere utile integrare strumenti di monitoraggio per visualizzare metriche d’uso, audit log e prestazioni in tempo reale.

Analisi e previsione dei costi totali

Piattaforma	Gratuito	Piani a pagamento	Costo LLM/embedding
Chatbase	20 messaggi/mese	Hobby $19/m, Standard $99/m (10k GPT-4 token), Illimitato $399/m	OpenAI GPT-4: ~$0.03/1k token input, $0.06/1k output
Dify AI	Sandbox free (200 crediti, 50 doc)	Pro $59/m, Team $159/m	Supporta vari modelli, OpenAI GPT-4 costi simili
Botpress	Pay-as-you-go base + $5/m AI credit	Plus $89/m, Team $495/m	Inclusi $5 crediti AI/m per GPT
LangChain	Open-source, gratuito	Dipende da LLM scelto (OpenAI come sopra)	Come sopra
Intercom	Nessun piano gratuito permanente	Da $29/agente/m + $0.99/conversazione chiusa	Incluso nel piano

Costi infrastrutturali aggiuntivi:

- Server/hosting (es. AWS t3.medium ~$20-40/mese)

- Vector DB (Pinecone/Weaviate cloud con costi, FAISS/Chroma on-prem free)
- Licenze LLM e costi di sviluppo/migrazione

Soluzioni no-code (Chatbase, Dify) riducono i costi iniziali di sviluppo ma hanno abbonamenti ricorrenti, mentre open-source (LangChain, Botpress) possono essere più economiche a lungo termine ma richiedono competenze e infrastruttura.

Feedback dei clienti (punti di forza e criticità)

- **Chatbase:** setup rapido e interfaccia user-friendly, ideale per MVP; limitata flessibilità e mancanza di live chat; costi GPT-4 possono crescere.
- **Dify AI:** facile da usare, builder drag&drop, open-source; piattaforma giovane con alcune limitazioni di maturità.
- **Botpress:** elevata flessibilità, interfaccia visuale, ottimo per sviluppatori; curva di apprendimento e roadmap non sempre chiara.
- **Intercom:** piattaforma potente all-in-one, efficace per supporto clienti; costosa e complessa, meno personalizzabile per RAG custom.

Esempi reali di applicazioni AI per l'assistenza

1. Chatbase – Fiverr Help Center Bot

Contesto
Fiverr è una piattaforma globale di freelance. Ha integrato un assistente AI tramite Chatbase per la gestione delle richieste e dei problemi più comuni (es. problemi di pagamento, consegne in ritardo, gestione account).

Funzionalità

- L'utente può fare domande in linguaggio naturale nella pagina "Help".
- L'assistente recupera le informazioni dalla knowledge base (FAQ e documentazione interna).
- La chatbot è disponibile 24/7 e supporta più lingue.

Punti di forza

- Velocità nel setup (circa 1 giorno).
- Risposte consistenti e precise.
- Riduzione dei ticket ripetitivi al supporto clienti.

Tecnologia
Chatbase con GPT-4 Turbo + PDF/documenti come base dati.

---

2. Dify AI – Notion AI Help Assistant

Contesto
Notion è un'app di produttività e workspace. Ha sperimentato internamente un bot tramite Dify AI per assistere gli utenti nel centro supporto e formazione.

Funzionalità

- Workflow visivo per creare risposte automatiche da articoli tecnici e tutorial.
- Integrazione API per domande specifiche sugli strumenti.

Punti di forza

- Integrazione diretta con database interni.
- Interfaccia drag-and-drop facilmente aggiornabile dal team di prodotto.
- Permette iterazioni rapide tramite test A/B.

Tecnologia
Dify AI open-source, deploy self-hosted + GPT-4.

---

3. Botpress – Service Canada Virtual Agent

Contesto
Service Canada, agenzia governativa canadese, utilizza Botpress per offrire un'assistenza AI automatica su servizi pubblici come pensioni, sussidi e imposte.

Funzionalità

- Assistenza via sito e chatbot WhatsApp.
- Domande complesse smistate automaticamente verso operatori umani.
- Utilizzo di moduli NLU per il riconoscimento delle intenzioni dell'utente.

Punti di forza

- Privacy e controllo totale dei dati (requisito critico in ambito pubblico).
- Flessibilità architetturale.
- Interfaccia visuale combinata con strumenti di sviluppo personalizzati.

Tecnologia
Botpress Cloud + knowledge base importata da documenti pubblici PDF + database SQL interno.

MVP vs sistemi scalabili

- **Prototipi rapidi (MVP):** Chatbase e Dify offrono setup guidato, bot funzionante in pochi minuti. LangChain con Python e API consente prototipi veloci con poco codice.
- **Sistemi scalabili e personalizzabili:**
  Botpress è pensato per aziende con multi-team, compliance e throughput elevati, mantenendo proprietà dati. Soluzioni custom con LangChain + hosting dedicato offrono massima personalizzazione ma con maggior impegno IT. Intercom è modulare ma costoso per grandi volumi.

Sitografia

- https://doxygen.nl
- https://overcast.blog
- https://dotnet.github.io/docfx
- https://getguru.com
- https://js.wiki
- https://gitbook.com
- https://deepwiki.com
- https://github.com/daeisbae/open-repo-wiki
- https://g2.com

- https://thectoclub.com
- https://docs.github.com
- https://coderefinery.github.io