Mettere in sicurezza una piccola web app con l'aiuto dell'AI

IkigaiApp — caso di studio su sicurezza e privacy

Salvatore Vivolo · Documento vivo, versione 0.4 · Ultimo aggiornamento 4 novembre 2026

> Questo documento descrive principi, scelte e risultati. Omette volutamente dettagli di configurazione, nomi interni e tutto ciò che potrebbe aiutare un attaccante.

1. In breve

IkigaiApp è una web app personale di autovalutazione, basata sul concetto giapponese di *ikigai*, con circa 170 iscritti. A ottobre 2026 un incidente di sicurezza presso il fornitore di hosting è diventato il punto di partenza per una revisione completa di sicurezza e privacy. In due settimane, lavorando con strumenti di AI sotto revisione umana, il progetto è passato da un "funziona" informale a una sicurezza documentata, testata e verificabile.

Ambito	Prima (ottobre 2026)	Ora (novembre 2026)
Segreti dopo l'incidente del fornitore	Potenzialmente esposti	Tutti sostituiti, compresa la chiave principale di cifratura
Controllo degli accessi	Mai verificato	Ogni route delle API controllata; tutti i problemi corretti e coperti da test
Account amministratore	Password o link via email	Autenticazione a due fattori obbligatoria (TOTP, RFC 6238)
Dati personali nei log	Indirizzi email, a volte segreti	Solo ID e codici con chiave (pseudonimi)
Cancellazione dell'account	Parziale	Completa, verificata da test automatici
Conservazione dei dati	Illimitata per diversi insiemi di dati	Scadenza automatica, come dichiarato nell'informativa
Dipendenze con vulnerabilità note	Decine di segnalazioni	Nessuna nota in produzione, verificato in automatico a ogni modifica
Test automatici superati	659	1119, tutti verdi e tutti bloccanti a ogni modifica
Aggiornamenti delle dipendenze	Manuali, occasionali	Automatici e settimanali, con test e attesa di sicurezza
Testi legali pubblici	In parte imprecisi	Riscritti per descrivere ciò che il codice fa davvero

2. Contesto

- **Prodotto:** un questionario che genera un "profilo Ikigai", una descrizione del risultato scritta con l'AI, esempi tratti da profili simili e strumenti personali (diario, to-do, flashcard).
- **Architettura:** un frontend web su una piattaforma edge gestita, un'API su una piattaforma container gestita nell'UE, un database documentale gestito nell'UE, un fornitore di email transazionali e un fornitore di AI.
- **Team:** una persona, più due assistenti AI: uno per analisi, specifiche e revisione, uno per l'implementazione.
- **Vincolo:** nessun budget per strumenti di sicurezza a pagamento. Tutto doveva essere ottenibile con piani gratuiti e buona ingegneria.

3. L'incidente da cui è partito tutto

A ottobre 2026 il fornitore di hosting ha comunicato che le variabili d'ambiente di alcuni clienti potevano essere state lette da terzi. Non era possibile stabilire se questo progetto fosse coinvolto, quindi la risposta ha ipotizzato il caso peggiore:

1. **Ogni segreto considerato esposto.** Credenziali del database, chiavi di firma delle sessioni, chiavi di firma di export e link, chiavi API, token di notifica: tutti rigenerati. La chiave principale che protegge le chiavi di cifratura del database è stata ruotata con una procedura dedicata e collaudata.
2. **Log e pannelli dei fornitori controllati** dall'inizio della finestra di esposizione: nessun segno di abuso.
3. **Autenticazione a più fattori attivata** su ogni account dei fornitori coinvolti (hosting, database, repository del codice, DNS, email, AI).
4. **Responsabilità GDPR:** una voce nel registro delle violazioni dei dati personali (art. 33.5 GDPR) documenta i fatti, la valutazione del rischio e i motivi per cui non era dovuta la notifica all'autorità di controllo.

La lezione: la cifratura dei campi è forte quanto il luogo in cui vivono le sue chiavi. Se quel luogo può essere compromesso, vanno ruotate anche le chiavi, non solo le password.

4. Sicurezza fin dalla progettazione

Dati cifrati campo per campo. I dati personali (email, nomi, nickname, fascia di reddito) e tutto ciò che gli utenti scrivono (diario, to-do, Piccole Ancore, flashcard) vengono cifrati dall'applicazione prima di arrivare al database, con AES-256-GCM. Le chiavi seguono uno schema *a busta* (envelope): le chiavi dei dati sono a loro volta cifrate da una chiave principale che non sta mai nel database. Ogni chiave ha una versione, così si possono ruotare senza interrompere il servizio. Una copia del database, da sola, non rivela questi campi.

Cercare senza decifrare. Per trovare un utente dall'email o verificare che un nickname sia unico senza decifrare tutta la tabella, l'app salva un *blind index*: un HMAC con chiave del valore normalizzato. Stesso valore, stesso indice. Senza la chiave l'indice non permette di risalire al valore, ma due valori uguali producono lo stesso indice: per questo indice e chiave vanno protetti insieme.

Dal server esce il minimo indispensabile. Un'unica funzione costruisce ogni vista di "un altro utente": un ID e un nome visualizzato, nient'altro. Nessun dato cifrato, email o ruolo arriva a chi non ne ha bisogno.

Sessioni e richieste. I token di sessione stanno in cookie HttpOnly, Secure e SameSite, irraggiungibili dagli script. Le richieste che modificano dati vengono controllate sull'origine. I link che agiscono senza login, come la disiscrizione, sono firmati e scadono. In produzione il server rifiuta di partire se manca un segreto obbligatorio: un'impostazione dimenticata non può ripiegare in silenzio su un valore debole.

Resistenza agli abusi. Limiti di richieste su autenticazione, verifica degli account, statistiche e funzioni AI. Le risposte sono uniformi e non rivelano se un indirizzo email ha un account.

5. Autenticazione

- Gli utenti accedono con un link via email o con la password.
- **Gli amministratori devono usare l'autenticazione a due fattori:** codici monouso a tempo secondo RFC 6238, lo standard aperto di Google Authenticator, Authy, Microsoft Authenticator, Apple Passwords e altre app. È implementata con la crittografia standard della piattaforma e verificata sugli esempi ufficiali della specifica. I codici non sono riutilizzabili; i dieci codici di recupero monouso sono salvati solo come hash lenti; il segreto è cifrato come gli altri dati personali.
- Esiste una **procedura d'emergenza** per il dispositivo smarrito. Può eseguirla solo il titolare dalla console di hosting, e ogni uso viene registrato nei log di audit.

6. Controllo degli accessi: una verifica route per route

Il passo più utile è stato un esame sistematico di ogni route delle API, 283 in tutto, con tre domande: chi può chiamarla, di chi sono i dati che tocca, cosa restituisce. La verifica ha trovato 14 problemi e 7 punti da approfondire. Si va da un controllo "è amministratore?" reso sempre vero da una coppia di parentesi mancanti, a dati di altri utenti raggiungibili cambiando un ID nella richiesta.

Sono stati corretti tutti, in quattro lotti. Ogni correzione è accompagnata da un test che **fallisce sul codice vecchio e passa su quello nuovo**: si dimostra che il buco c'era e che è chiuso.

7. Privacy e GDPR

La privacy è stata trattata come un vincolo di progetto, a volte a scapito della comodità:

- **Statistiche senza identità.** Le statistiche d'uso non registrano né l'ID utente né l'indirizzo IP: solo pagine, un identificativo casuale della visita e un contesto generico. Si è rinunciato di proposito alle statistiche per singolo utente. Resta un solo cookie di prima parte, con un valore casuale che dura 30 minuti; le statistiche sono usate solo in forma aggregata e non vengono incrociate né condivise. Valutata questa configurazione sulle linee guida del Garante, il cookie è trattato come tecnico, senza banner di consenso.
- **Conservazione automatica.** Le statistiche d'uso scadono dopo 13 mesi, i dati tecnici dopo 90 giorni, i registri dell'uso dell'AI dopo 12 mesi, i log di sicurezza tra 90 giorni e 12 mesi. Lo fa il database, non la buona volontà.

- **Cancellazione vera.** Eliminare l'account cancella tutto ciò che vi è collegato, in ogni collezione. Un test crea dati ovunque e verifica che non resti nulla. Gli account cancellati in passato sono stati ripuliti a posteriori. Restano solo i log di sicurezza fino alla scadenza e i backup cifrati fino alla loro sostituzione, entro 6 mesi, come dichiarato nell'informativa.
- **Esportazione.** Gli utenti possono scaricare in qualsiasi momento tutti i loro dati, in JSON o CSV.
- **Esempi pseudonimi, con possibilità di uscita.** La funzione "profili simili" usa copie dei risultati senza nomi né contatti, collegate all'account solo da un codice con chiave, così si possono cancellare su richiesta. Un unico interruttore permette di uscirne.
- **Niente interruttori finti.** Sono stati tolti i consensi che non facevano nulla. L'app non fa marketing, non vende dati e non addestra modelli.
- **Log con il minimo di dati personali.** I log di audit e degli eventi conservano ID e codici con chiave, cioè dati pseudonimi, con una scadenza; email e segreti vengono oscurati in scrittura e in lettura.
- **Documenti che corrispondono al codice.** Informativa privacy, cookie policy, pagina dei diritti, trasparenza AI e termini sono stati riscritti partendo dall'implementazione. Sono tenuti e versionati un registro interno dei trattamenti, l'elenco dei fornitori, il calendario di conservazione e la procedura per le richieste degli interessati.
- **Una promessa pubblica.** La pagina "La nostra promessa" riassume in sette impegni, in parole semplici, cosa l'app fa per proteggere i dati, con i link all'informativa completa.

8. AI, in modo responsabile

- L'AI serve a tre funzioni: il profilo narrativo, la generazione di flashcard e la traduzione dei messaggi dell'amministratore. Riceve solo quello che serve a ciascuna. Per il profilo sono categorie derivate, mai nomi, email, ID utente o risposte libere.
- Il fornitore non usa questi dati per addestrare i modelli; non c'è alcuna decisione automatica con effetti giuridici.
- Tutto ciò che è generato porta l'etichetta "Generato con AI", che resta anche quando un contenuto viene condiviso. Gli articoli del blog dicono quando testi o immagini sono stati realizzati con l'aiuto dell'AI.
- Secondo la valutazione fatta rispetto all'AI Act europeo, le funzioni non rientrano tra quelle ad alto rischio; le etichette rendono visibile l'uso dell'AI. Il calcolo dell'archetipo e la ricerca di profili simili sono deterministici e spiegabili, non machine learning.

9. A che punto è l'app rispetto alle minacce del 2026

Attacchi assistiti dall'AI. Oggi gli attaccanti usano l'AI per scandagliare in massa le applicazioni alla ricerca di controlli di accesso mancanti, per automatizzare i tentativi con credenziali rubate e per personalizzare il phishing. La risposta qui è togliere ciò che questi strumenti cercano:

- ogni route è stata verificata;
- le risposte non rivelano se un account esiste;
- i limiti di richieste rallentano l'automazione;
- l'account admin richiede un secondo fattore;
- i campi cifrati e i log minimi riducono ciò che una violazione esporrebbe.

Le funzioni AI generano solo testo e non possono agire sui dati, quindi la "prompt injection" non ha modo di fare danni.

Catena di fornitura del software. Le recenti ondate di pacchetti npm malevoli hanno mostrato che una sola dipendenza compromessa può rubare segreti già durante l'installazione. Le pratiche adottate sono:

- versioni esatte e lockfile nel repository;
- almeno sette giorni di attesa prima di adottare una nuova versione;
- script di installazione disattivati durante gli aggiornamenti;
- installazioni pulite e riproducibili in fase di deploy;
- integrazione continua: a ogni modifica girano i test bloccanti, la build e un controllo delle vulnerabilità nelle dipendenze di produzione, e una modifica che non passa non entra;
- aggiornamenti automatici settimanali, raggruppati e con sette giorni di attesa, mentre le correzioni di sicurezza arrivano subito; le versioni major restano decisioni manuali;
- azioni della CI fissate a versioni esatte (hash del commit);
- nessuna vulnerabilità nota nelle dipendenze di produzione.

Infrastruttura. L'incidente del fornitore ha mostrato che i segreti su una piattaforma di hosting sono un punto unico di rottura. Sono in atto:

- MFA su tutti gli account dei fornitori;
- segreti sostituiti e sigillati;
- configurazione che blocca l'avvio se manca un segreto;
- un endpoint di stato che non rivela nulla;
- pagine di debug eliminate;
- backup cifrati e verificati.

Limiti, detti onestamente:

- nessun test di penetrazione esterno, per ora;
- limiti di richieste gestiti per singola istanza del server;
- avvisi automatici sulle anomalie ancora basilari;
- chiavi dei dati ancora da ruotare e backup ancora manuali.

10. Come è stato fatto: con l'AI, sotto revisione umana

- **Ruoli.** Un assistente AI ha analizzato il codice, scritto le specifiche e rivisto ogni modifica; un secondo le ha implementate su branch separati. Il titolare ha preso ogni decisione, ha unito, pubblicato ed eseguito di persona ogni operazione in produzione.
- **Regole nel repository.** Un file di regole dice agli agenti cosa non devono mai fare: niente segreti nel codice o nei log, nessun indebolimento dei controlli di sicurezza, test per ogni cambio di comportamento, ogni modifica agli indici documentata, avanzamento riportato.
- **Prove, non rassicurazioni.** Ogni lotto si è chiuso con un report: test aggiunti, suite completa confrontata con il ramo principale **test per test** (non solo i totali), comandi da eseguire in produzione.

- **Modifiche in produzione in due tempi.** Ogni modifica ai dati è uno script che per default fa una prova e stampa solo conteggi, mai dati personali. L'esecuzione vera arriva dopo aver controllato i numeri, e dopo un backup cifrato nuovo.
- **La revisione trova quello che i test non vedono.** Un lotto superava tutti i suoi test, ma cancellava da tre collezioni usando nomi sbagliati, e i test usavano gli stessi nomi sbagliati. La revisione umana del codice l'ha scoperto prima della produzione, e la correzione ha fatto ricavare i nomi dai modelli dei dati. In un'altra occasione, la verifica di un backup ha rivelato che lo script mescolava un ritorno a capo nella chiave di cifratura; è stato corretto prima che servisse un ripristino.
- **La CI trova quello che il computer dello sviluppatore non vede.** Alla prima esecuzione su un server Linux, quattro file di test verdi sul Mac sono falliti: un percorso valido solo su macOS, dipendenze non installate, un test sensibile ai tempi. Nessun bug dell'app, ma esattamente il tipo di sorpresa che la CI serve a scoprire prima del merge.
- **Rimettere in piedi i test vecchi trova bug veri.** Portare a verde i 183 test che fallivano da tempo ha richiesto di distinguere, uno per uno, test superati da bug reali. Ne sono emersi alcuni: un questionario inviato poteva restare in bozza senza punteggi, uno stato di registrazione non si aggiornava dopo la verifica dell'email, la copia di un contenuto cifrato finiva nella risposta senza essere decifrata. Le correzioni che toccavano autenticazione e cifratura sono state fermate, documentate e fatte in un lotto separato, con revisione. Prima di renderli bloccanti, i test instabili sono stati resi deterministici e ripetuti venti volte su due sistemi operativi.

11. Stato e prossimi passi

Voce	Stato
Sostituzione dei segreti dopo l'incidente, MFA sui fornitori	Fatto
Verifica e correzione del controllo degli accessi	Fatto
Autenticazione a due fattori per gli amministratori	Fatto
Privacy: cancellazione, esportazione, conservazione, log minimi	Fatto
Documenti legali e di conformità allineati al codice	Fatto
Pagina pubblica "La nostra promessa"	Fatto
Integrazione continua con test bloccanti e aggiornamenti automatici delle dipendenze	Fatto
Tutta la suite di test verde e bloccante per i merge	Fatto
Rotazione delle chiavi di cifratura dei dati	Previsto
Backup automatici cifrati con prova di ripristino	Previsto
Avvisi automatici sulle anomalie (picchi di login falliti o di richieste bloccate, nuovi accessi admin, backup falliti)	Dopo i backup
Età minima delle versioni in npm, azioni della CI fissate a versioni esatte	Fatto
Verifica delle firme dei pacchetti, scansione di codice e segreti	Prossimo
Revisione degli header di sicurezza, prove periodiche di ripristino, procedura scritta per gli incidenti	Prossimo
Passkey per gli amministratori, due fattori facoltativi per gli utenti	Più avanti
Test di penetrazione esterno o scansione web automatica	Più avanti

12. Lezioni per qualsiasi nuovo progetto

1. Verificare il controllo degli accessi route per route prima di aggiungere funzioni: è lì che si nascondono i bug veri.
2. Cifrare i campi sensibili nell'applicazione, e prevedere la rotazione delle chiavi fin dal primo giorno.
3. Raccogliere meno dati. Ciò che non conservi non può trapelare, non richiede regole di conservazione né consenso.
4. Far sì che l'informativa descriva il codice, e mantenerli allineati.
5. Usare l'AI per la velocità e le persone per il giudizio. Chiedere prove (test che falliscono prima, confronti prima/dopo), non rassicurazioni.
6. Ogni modifica ai dati in produzione: backup, prova, revisione, esecuzione, verifica.

Documento vivo: viene aggiornato a ogni passo completato.