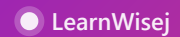


Esercizio di coding con l'IA

Costruisci il progetto FieldOps di questo modulo con ChatGPT o Claude partendo da un prompt già pronto che indirizza il modello alla specifica del modulo, poi rivedi, esegui ed estendi quello che ti restituisce.



Lo stesso modulo, costruito come costruirai davvero le cose.

Tutte le altre sezioni di questo modulo spiegano come funziona "API del dispositivo tramite il bridge ibrido". Questa ti chiede di produrlo: dirigendo un assistente con la specifica del modulo stesso, poi verificando che il risultato giri e faccia quello che descrive la traccia del lab. È il flusso di lavoro che userai sul lavoro vero, provato su qualcosa di cui conosci già la risposta giusta.

Che cosa stai chiedendo

Aggiungi le funzionalità del dispositivo a FieldOps: un'azione Take Photo che usa la fotocamera del dispositivo nello shell nativo e ripiega su un controllo Upload nel browser, un'acquisizione della firma caricata come immagine, un'azione Navigate che legge la posizione del dispositivo e apre l'app delle mappe, una notifica push quando viene assegnato un ordine di lavoro, e una verifica delle capacità che mostra o nasconde ogni azione a seconda dello shell.

Il modulo chiede questi deliverable, e il prompt qui sotto li chiede all'assistente esattamente così:

- ✓ Acquisizione dalla fotocamera con ripiego sull'upload nel browser
- ✓ Acquisizione e caricamento della firma
- ✓ Geolocalizzazione e passaggio all'app delle mappe
- ✓ Notifica push all'assegnazione
- ✓ Matrice delle verifiche delle capacità tra gli shell

Perché è fatto così

Un tecnico di FieldOps fotografa una valvola corrosa, raccoglie una firma sul vetro e guida fino all'indirizzo successivo — mentre il dispatcher che gli ha assegnato l'intervento osserva lo stesso ordine di lavoro in un browser desktop. Ogni controllo di FieldOps è un oggetto C# sul tuo server, che non può toccare la fotocamera, il chip GPS o l'area delle notifiche. Questo modulo è il giro di andata e ritorno che colma quel divario, e ciò che la schermata fa dove quel giro non esiste.

Questi sono i termini che l'assistente deve azzeccare. Se quello che ti restituisce li usa con leggerezza, non ha capito il modulo:

Termine	Che cosa significa qui
bridge del dispositivo	Il canale di messaggi che uno shell ibrido espone alla pagina che ospita: il tuo server chiede allo shell di eseguire un'operazione sul dispositivo, e lo shell risponde con un messaggio che l'app ritrasforma in un evento lato server.
host object	L'oggetto che una WebView inietta nella pagina ospitata così che il JavaScript lì presente possa chiamare codice nativo. Il C# lato server lo raggiunge solo tramite l'interop JavaScript dell'app.
descrittore delle capacità	Le funzionalità che uno shell dichiara all'avvio di una sessione — fotocamera, posizione, notifiche, biometria — messe in cache e consultate prima di mostrare qualsiasi azione sul dispositivo.
miglioramento progressivo	Progettare ogni schermata in modo che il percorso del semplice browser sia una funzionalità completa e il percorso nativo ne sia una versione migliore, mai l'unico che funziona.
autorizzazione just-in-time	Chiedere alla piattaforma l'accesso a fotocamera, posizione o notifiche nel momento in cui l'utente preme il pulsante che ne ha bisogno, subito dopo una frase che spiega perché.
evento di risultato del bridge	La risposta asincrona — byte della foto, coordinate o un rifiuto — che arriva su una richiesta successiva al clic che l'ha chiesta, e quindi ha bisogno di un proprio push.
archiviazione fuori dalla web root	Scrivere foto e firme acquisite dove nessun URL le espone, e poi servirle tramite un gestore che verifica se questa sessione può vedere quell'ordine di lavoro.

Costruiscilo — scegli il tuo assistente

Entrambi i pulsanti aprono una nuova scheda con lo **stesso prompt già pronto**. Il prompt indirizza il modello alla specifica di questo modulo — un unico file Markdown con la lezione, la traccia del lab e i deliverable — così lavora sul materiale vero invece di tirare a indovinare che cosa sia il Modulo 5. Puoi aprire tu stesso [module-05.md](#) per vedere esattamente che cosa leggerà.



Usa ChatGPT

Apri [chatgpt.com](#) con il prompt di questo modulo già caricato. Funziona con GPT-5 o Codex; qualsiasi piano che possa navigare il web recupererà la specifica.



Usa Claude

Apri [claude.ai](#) con lo stesso prompt già caricato. Chi usa Claude Code può invece incollarlo in un terminale dentro una cartella vuota.



Mandalo così com'è e ottieni un progetto, non un frammento

Il prompt si chiude chiedendo l'intera soluzione in un unico `.zip` con i passi per compilarla ed eseguirla. Scaricalo, apri la soluzione in Visual Studio, premi F5 e FieldOps si apre su `localhost`. Ti servono l'SDK .NET e il pacchetto NuGet `Wisej-4`; tutto il resto il prompt chiede all'assistente di scriverlo da sé.

Sto seguendo il Modulo 5 - "API del dispositivo tramite il bridge ibrido" - del corso LearnWisej "Pacchetti ibridi e nativi", e voglio costruire con te il progetto del modulo. Rispondimi in italiano.

Comincia leggendo la specifica del modulo. È un unico file Markdown (in inglese) che contiene l'intera lezione, la traccia del lab e i deliverable:

<https://learnwisej.com/assets/courses/hybrid-native-packaging/specs/module-05.md>

Poi costruisci un progetto Wisej.NET 4 completo che io possa aprire in Visual Studio:

1. Progetto "FieldOps", C#, net8.0 o successivo, pacchetto NuGet Wisej-4 4.1.x, host Kestrel che chiama `app.UseWisej()`.
2. Costruisci quello che chiede il lab della specifica: Aggiungi le funzionalità del dispositivo a FieldOps: un'azione Take Photo che usa la fotocamera del dispositivo nello shell nativo e ripiega su un controllo Upload nel browser, un'acquisizione della firma caricata come immagine, un'azione Navigate che legge la posizione del dispositivo e apre l'app delle mappe, una notifica push quando viene assegnato un ordine di lavoro, e una verifica delle capacità che mostra o nasconde ogni azione a seconda dello shell.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - Acquisizione dalla fotocamera con ripiegio sull'upload nel browser
 - Acquisizione e caricamento della firma
 - Geolocalizzazione e passaggio all'app delle mappe
 - Notifica push all'assegnazione
 - Matrice delle verifiche delle capacità tra gli shell
4. Segui la specifica ovunque nomini una classe, una proprietà o un evento di Wisej.NET - usa esattamente quello. Dove lascia una scelta aperta, dimmi quale API hai scelto e perché.
5. Tieni l'autorità sul server: stato e validazione vivono in .NET, il browser disegna e riferisce. Non spostare mai una decisione in JavaScript solo perché lì è più comodo.
6. Rendi il risultato dimostrabile: devo poterlo eseguire e vedere il comportamento di cui parla questo modulo, compreso che cosa succede quando qualcosa va storto.

Prima di scrivere codice, riformula con parole tue che cosa costruirai e dove starà ogni pezzo, così posso correggere il progetto finché costa poco.

Quando il codice è pronto, impacchetta l'intera soluzione in un unico file .zip scaricabile e dammi i passi esatti per compilarla ed eseguirla.

Infine rispondi a queste domande della specifica, usando come prova la tua implementazione:

- Il risultato della fotocamera arriva quaranta secondi dopo il clic che l'ha chiesto. Spiega su quale richiesta arriva, quale stato di sessione deve trovare per sapere che cosa fare dei byte, e perché la picture box non compare nel browser senza un push esplicito.
- Tre settimane fa un tecnico ha negato in modo permanente l'autorizzazione alla posizione sul proprio telefono. Descrivi esattamente che cosa dovrebbe fare oggi la pressione di Navigate, e indica l'unica cosa che l'app non deve mai fare in risposta.
- Take Photo funziona nello shell mobile e il ripiegio su Upload funziona in Chrome desktop. Indica una cosa davvero diversa per l'utente tra questi due percorsi, e argomenta se quella differenza sia accettabile per un tecnico sul campo.

Chiedimi qualsiasi cosa non ti sia chiara prima di cominciare.

Poi rivedi quello che ti è tornato indietro

Codice che compila non è ancora codice giusto. Le domande interessanti sono domande di progetto: è questo il controllo Wisej.NET che il compito richiede davvero, segue il ciclo di vita che il framework ti offre, lo stato autorevole sta sul server o è stato parcheggiato nel browser perché era comodo? Leggi il risultato da revisore, non da cliente.

Questo modulo ti dà la checklist. Rispondi a ciascuna domanda guardando il codice che hai appena ricevuto:

- ✓ Il risultato della fotocamera arriva quaranta secondi dopo il clic che l'ha chiesto. Spiega su quale richiesta arriva, quale stato di sessione deve trovare per sapere che cosa fare dei byte, e perché la picture box non compare nel browser senza un push esplicito.
- ✓ Tre settimane fa un tecnico ha negato in modo permanente l'autorizzazione alla posizione sul proprio telefono. Descrivi esattamente che cosa dovrebbe fare oggi la pressione di Navigate, e indica l'unica cosa che l'app non deve mai fare in risposta.
- ✓ Take Photo funziona nello shell mobile e il ripiego su Upload funziona in Chrome desktop. Indica una cosa davvero diversa per l'utente tra questi due percorsi, e argomenta se quella differenza sia accettabile per un tecnico sul campo.

Tutto ciò che hai dovuto correggere vale la pena di scriverlo. Una breve nota con le convenzioni e le decisioni a cui sei arrivato, incollata nel primo messaggio della sessione successiva, vale più di qualsiasi prompt ingegnoso.



Per questo corso non c'è ancora un esempio di riferimento — il tuo è il primo

Il [repository degli esempi LearnWisej](#) contiene build già fatte per quasi tutti i corsi, ma non ancora per questo. Significa che non c'è niente con cui confrontarsi e nessuna scorciatoia: la specifica e la tua revisione sono tutta la rete di sicurezza. Tieni quello che costruisci — e se viene bene, ci farebbe piacere vederlo.

Tre prompt per andare oltre

Punta il tuo assistente sul progetto che hai appena costruito e prova questi. Ognuno è un esercizio di revisione ed estensione, non una riscrittura.

- ✓ **Rivedilo come faresti con una pull request.** *"Accompagnami attraverso questo codice FieldOps come se stessi rivedendo la pull request di un collega e non l'avessi scritto tu. Dove vive lo stato, che cosa succede dall'inizio alla fine su una singola azione dell'utente, che cosa rimanderesti indietro per modifiche, e quali parti non ti sentiresti tranquillo a mettere davanti a un cliente? Sii specifico e cita le righe di cui parli — non riassumermi il progetto."*
- ✓ **Sostituisci un pezzo senza spostare le giunzioni.** *"Scegli la parte di questo progetto FieldOps che più evidentemente sta al posto di qualcosa di reale — una sorgente dati finta, un archivio in memoria, un servizio semplificato — e sostituiscila con un'implementazione di forma produttiva. Il contratto ai due lati non deve cambiare: stessi tipi, stesse chiamate, stesso comportamento in caso di errore. Dimmi che cosa l'implementazione reale ti ha costretto a gestire e che la versione provvisoria non gestiva."*

- ✓ **Fallo sopravvivere a una brutta giornata.** *"Esamina questo progetto FieldOps cercando le modalità di guasto che non gestisce ancora — input sbagliato o mancante, una dipendenza lenta o non disponibile, lavoro che arriva dopo che l'utente ha cambiato pagina, e due azioni che si accavallano. Per ciascuna dimmi prima se è un rischio reale in Wisej.NET 4 e perché, e solo dopo scrivi la correzione. Dove decidi che non serve gestire nulla, dillo e spiega il ragionamento invece di aggiungere codice difensivo."*



Portalo dentro il lab

Il lab pratico che chiude questo modulo chiede gli stessi deliverable che hai appena fatto produrre all'assistente. Usa quello che hai qui come forma della tua risposta e come prova a sostegno — non come la risposta in sé. Il lab viene valutato sul tuo ragionamento, e quella parte deve essere tua.