ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DIPARTIMENTO DI INFORMATICA - SCIENZA e INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Analisi, progettazione e distribuzione in cloud di applicativo per l'organizzazione di eventi condivisi

Relatore:
Chiar.mo Prof.
Michele Colajanni

Presentata da:
Giacomo Romanini

Sessione Luglio 2025

Anno Accademico 2025/2026

Abstract

Lo sviluppo di un applicativo multipiattaforma diretto all'organizzazione di eventi condivisi, caratterizzato in particolare dalla condivisione multimediale in tempo reale, richiede opportune capacità di scalabilità, atte a garantire una risposta efficace anche con alti volumi di richieste, offrendo prestazioni ottimali. Le tecnologie cloud, con la loro disponibilità pressoché illimitata di risorse e la completa e continua garanzia di manutenzione, offrono l'architettura ideale per il supporto di simili progetti, anche con fondi limitati.

Tuttavia, l'integrazione tra la logica applicativa ed i molteplici servizi cloud, insieme alla gestione delle loro interazioni reciproche, comporta sfide specifiche, in particolare legate all'ottimizzazione di tutte le risorse. L'individuazione e la selezione delle soluzioni tecnologiche più adatte per ogni obiettivo, così come l'adozione delle migliori pratiche progettuali, devono procedere parallelamente con lo sviluppo del codice, al fine di sfruttare efficacemente le potenzialità offerte.

In tale prospettiva, questa tesi illustra le scelte progettuali ed implementative adottate nello sviluppo dell'applicativo in questione, evidenziando l'impatto dell'integrazione delle risorse cloud sul risultato finale.

Indice

Introduzione

In un contesto sociale sempre più connesso, la crescente quantità di contatti, la rapidità delle comunicazioni e l'accesso universale alle informazioni rendono la ricerca, l'organizzazione e la partecipazione ad eventi estremamente facile, ma al contempo generano un ambiente frenetico e spesso dispersivo.

Risulta infatti difficile seguire tutte le opportunità a cui si potrebbe partecipare, considerando le numerose occasioni che si presentano quotidianamente. Basti pensare, ad esempio, alle riunioni di lavoro, alle serate con amici, agli appuntamenti informali per un caffè, ma anche a eventi più strutturati come fiere, convention aziendali, concerti, partite sportive o mostre di artisti che visitano occasionalmente la città.

Questi eventi possono sovrapporsi, causando dimenticanze o conflitti di pianificazione, con il rischio di delusione o frustrazione. Quando si è invitati a un evento, può capitare di essere già impegnati, o di trovarsi in attesa di una conferma da parte di altri contatti. In questi casi, la gestione degli impegni diventa complessa: spesso si conferma la partecipazione senza considerare possibili sovrapposizioni, o dimenticandosi, per poi dover scegliere e disdire all'ultimo momento.

D'altra parte, anche quando si desidera proporre un evento, la ricerca di un'attività interessante può diventare un compito arduo, con la necessità di consultare numerosi profili social di locali e attività, senza avere inoltre la certezza che gli altri siano disponibili. Tali problemi si acuiscono ulteriormente quando si tratta di organizzare eventi di gruppo, dove bisogna allineare gli impegni di più persone.

In questo contesto, emergono la necessità e l'opportunità di sviluppare uno strumento che semplifichi la proposta e la gestione degli eventi, separando il momento della proposta da quello della conferma di partecipazione. In tal modo, gli utenti possono valutare la disponibilità degli altri prima di impegnarsi definitivamente, facilitando in conteporanea sia l'invito sia la partecipazione.

In risposta a tali richieste è stata creata Wyd, un'applicazione che permette agli utenti di organizzare i propri impegni, siano essi confermati oppure proposti. Essa permette anche di rendere più intuitiva la ricerca di eventi attraverso la creazione di uno spazio virtuale centralizzato dove gli utenti possano pubblicare e consultare tutti gli eventi disponibili, diminuendo l'eventualità di perderne qualcuno. La funzionalità chiave di questo progetto si fonda sull'idea di affiancare alla tradizionale agenda degli impegni confermati un calendario separato, che mostri tutti gli eventi a cui si potrebbe partecipare.

Una volta confermata la partecipazione a un evento, questo verrà spostato automaticamente nell'agenda personale dell'utente. Gli eventi creati potranno essere condivisi con persone o gruppi, permettendo di visualizzare le conferme di partecipazione. Considerando l'importanza della condivisione di contenuti multimediali, questo progetto prevede la possibilità di condividere foto e video con tutti i partecipanti all'evento, attraverso la generazione di link per applicazioni esterne o grazie all'ausilio di gruppi di profili. Al termine dell'evento, l'applicazione carica automaticamente le foto scattate durante l'evento, per allegarle a seguito della conferma dell'utente.

Figura 1: Il logo di Wyd

La realizzazione di un progetto come Wyd implica la risoluzione e la gestione di diverse problematiche tecniche. In primo luogo, la stabilità del programma deve essere garantita da un'infrastruttura affidabile e scalabile. La persistenza deve essere modellata per fornire alte prestazioni sia in lettura che in scrittura indipendentemente dalla quantità delle richieste, rimanendo però aggiornata e coerente. La funzionalità di condivisione degli eventi richiede inoltre l'aggiornamento in tempo reale verso tutti gli utenti coinvolti. Infine, il caricamento ed il salvataggio delle foto aggiungono la necessità di gestire richieste di archiviazione di dimensioni significative.

Organizzazione dei capitoli

Il seguente elaborato è suddiviso in cinque capitoli.
Nel primo capitolo si affronta la fase di analisi delle funzionalità, durante la quale, partendo dall'idea astratta iniziale, si definiscono i requisiti e le necessità del sistema, per poi creare la struttura generale ad alto livello dell'applicazione.
Nel secondo capitolo si affrontano le principali scelte architetturali e di sviluppo che hanno portato a definire la struttura centrale dell'applicazione.
Il terzo capitolo osserva lo studio effettuato per gestire la memoria, in quanto fattore che più incide sulle prestazioni. Particolare attenzione è stata dedicata, infatti, a determinare le tecnologie e i metodi che meglio corrispondono alle esigenze derivate dal salvataggio e dall'interazione logica degli elementi.
Il quarto capitolo si concentra sulle scelte implementative adottate per l'inserimento le funzionalità legate alla gestione delle immagini, che, oltre ad introdurre problematiche impattanti sia sulle dimensioni delle richieste sia sull'integrazione con la persistenza, richiedono l'automatizzazione del recupero delle immagini.
Infine, nel quinto capitolo, verranno analizzati e discussi i risultati ottenuti testando il sistema.

Capitolo 1

Risultati

Per poter definire l'efficienza e l'effettiva scalabilità del sistema creato è necessario misurare le sue prestazioni. A questo fine, vengono eseguiti dei test che effettuano un elevato numero di richieste, per valutare il suo comportamento in momenti di grande utilizzo e verificare così che sia garantita la qualità del servizio, anche in situazioni di stress.

Nell'ottica di ottenere dei risultati che descrivano fedelmente il comportamento dell'applicazione, evitando però di eseguire un test su ogni funzione implementata, sono state selezionate alcune funzionalità. La scelta di queste funzionalità deriva dalla capacità di poter ricondurre le loro prestazioni a tutto il resto del sistema, in base sia alla similitudine nel comportamento, che all'utilizzo dei vari servizi, che permette di misurarne la qualità della cooperazione.

In particolare, le funzionalità di creazione e selezione degli eventi permettono di valutare direttamente le prestazioni in scrittura e lettura dei dati sul database, influenzando Azure Functions e Cosmos. L'aggiornamento di un dato di un evento, richiedendo una scrittura parziale, consente di misurare anche questa particolarità dei database non relazionali. Ancora più rilevanti però sono le conseguenze che scatena. Per garantire la consistenza la funzione aggiunge infatti un messaggio in coda al Service Bus, che verrà poi preso in carico da un'altra Azure Function. Infine, la richiesta di caricamento delle immagini ci permette di valutare il rapporto con Azure Storage Container, ultimo componente fondamentale dell'applicazione.

1.1 L'impostazione dei test

Per l'implementazione dei test è stato usato Azure Load Testing. Azure Load Testing(ALT) è un servizio che dà la possibilità di eseguire delle richieste secondo un carico impostabile, per poi monitorarne e mostrarne i risultati. Questo permette di testare le prestazioni dei servizi direttamente all'interno della piattaforma. Integrandosi con l'ambiente Azure consente inoltre di associare al test il monitoraggio di ulteriori risorse, allineando ai risultati complessivi le prestazioni riportate dai singoli componenti, permettendo analisi più comprensive e puntuali.

Azure Load Testing

Il carico viene definito tramite diversi parametri che possono essere impostati attraverso l'interfaccia o in un file di configurazione. Vengono così stabiliti la durata del test, il numero di istanze (ovvero server fisici su cui verrà eseguito il test) e il numero di thread per istanza. Per thread si intente un processo che, in parallelo con gli altri, esegue le richieste, fornendo un'idea generale sulla quantità delle richieste che verranno create. Se le istanze sono più di una, si può decidere di distribuirle su server in diverse posizioni geografiche, per ottenere informazioni relative alla qualità del servizio in diverse parti del globo.

Azure presenta tre modalità secondo le quali il carico può variare: lineare, in cui il carico varia stabilmente, scalare, nel quale il carico aumenta ogni intervallo di una quantità stabilita e picco, in cui il carico viene concentrato in un breve intervallo di tempo. Una volta stabilita la modalità, sarà necessario definire le caratteristiche che descrivono il suo comportamento. Ad esempio, la modalità lineare necessita di sapere in quanto tempo arrivare al carico massimo. In ogni caso è richiesta la durata del test.

I test possono essere eseguiti tramite url o usando un file di configurazione. Il primo caso necessita che la funzione da testare risponda a una richiesta HTTP. Il test prevede semplicemente la sua invocazione per il numero desiderato di volte. Permette di essere configurata completamente tramite interfaccia grafica, e risponde alla maggioranza delle necessità per testare prestazioni generiche. Nel caso in cui si voglia però misurare fun-

zionalità più complesse, che richiedono l'interazione con file esterni, la modifica in corso delle richieste o l'esecuzione di più richieste in successione, è necessario utilizzare un file apposito.

All'interno del file di configurazione bisogna indicare, tramite un apposito codice, il carico voluto e le richieste da eseguire. Oltre a fornire una maggiore precisione nella definizione del carico, la possibilità di impostare la modalità del test consente un'elevata libertà nel stabilire le proprietà delle richieste e le loro eventuali interazioni. Questo permette, ad esempio, di caricare e leggere file, che possono contenere informazioni utili per variare le richieste o possono essere usati per scambiare dati. Si può inoltre simulare un'interazione più complessa, in cui si aspetta una richiesta per ricevere il risultato, per poi usarlo per inviarne una seconda.

Il file può essere sviluppato usando Apache JMeter o Locust come codici di configurazione. I test sono stati implementati in JMeter, essendo quello utilizzato di default da Azure. L'interfaccia di Azure Load Testing per la creazione del test tramite del file di configurazione prevede il suo inserimento, assieme a tutti gli eventuali file allegati di cui necessita. Prevede inoltre l'impostazione del numero di istanze per il quale si vuole distribuire il test, nelle quali verrà duplicato.

Per ogni test di entrambe le modalità si possono infine specificare quali altre risorse del progetto monitorare, affiancando alle prestazioni generali del test le prestazioni specifiche dei singoli componenti.

1.2 Velocità in creazione

Il primo test si concentra sulla velocità di creazione di elementi. La creazione è un'operazione che richiede l'inizializzazione logica di oggetti sul server, per poi procedere a salvarli sul database. Per quanto l'interazione logica comporti un tempo limitato, la scrittura di dati sul database è un'azione che richiede normalmente tempi significativi, in quanto comporta la modifica diretta sul disco.

In particolare, si è scelto di testare la creazione di un nuovo evento, vista sia la sua centralità nell'applicazione che la sua complessità logica. La creazione di un nuovo evento infatti comporta, oltre alla creazione dell'elemento Event stesso, l'inizializzazione dei ProfileEvent ed EventProfile associati.

1.3 Velocità in lettura

perchè? La

1.4 Velocità degli aggiornamenti

locking elemento

1.5 Garanzia di propagazione delle informazioni

Oltre al tempo di propagazione delle modifiche, bisogna anche assicurarsi del totale successo dell'operazione.

1.6 Velocità di upload delle immagini

1.7 Velocità in lettura

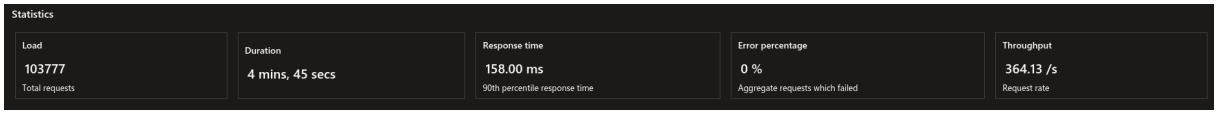

Figura 1.1: 158 ms su una media di 364 richieste al secondo

Figura 1.2: il tempo di risposta non varia in base al numero di richieste

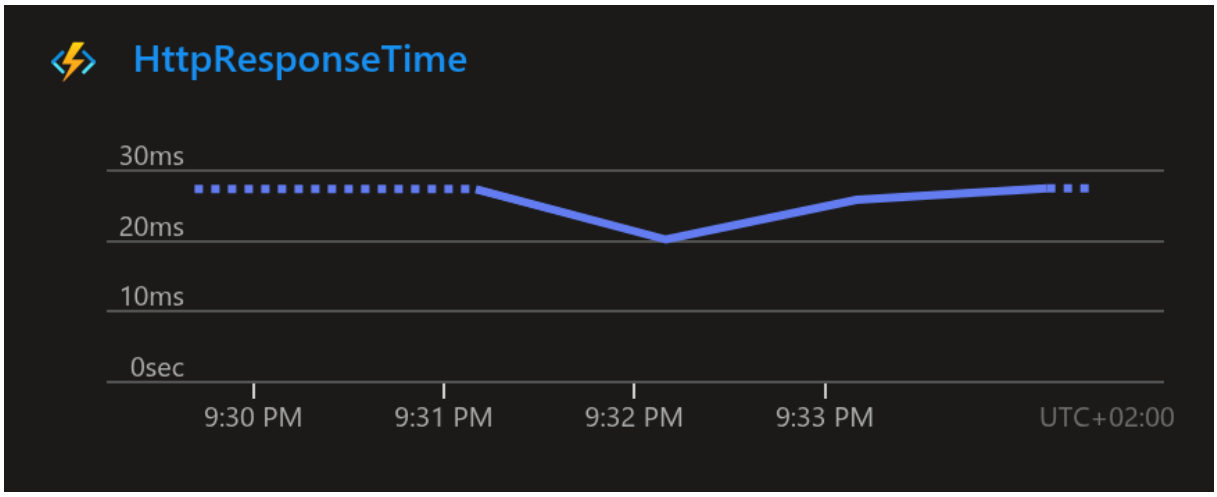

Figura 1.3: Dettaglio della velocità senza tempo di trasmissione

1.8 Caricamento di immagini concorrenti

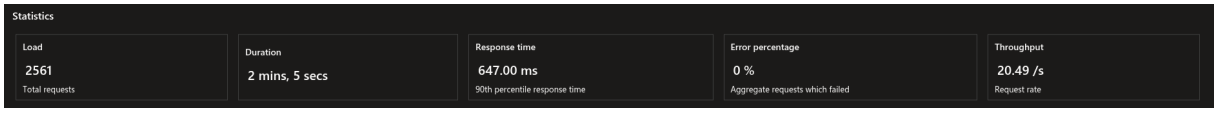

Figura 1.4: 2 MB di immagini in 600 ms 20 volte al secondo

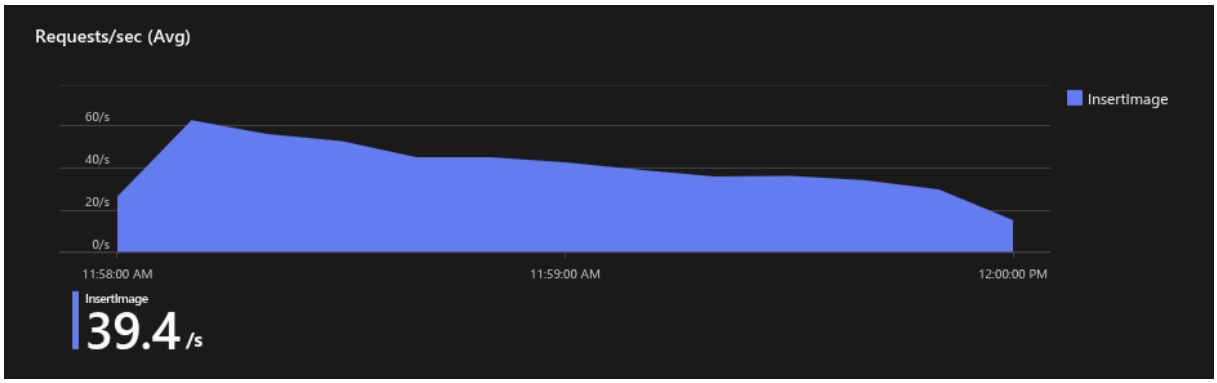

Figura 1.5: Andamento delle richieste

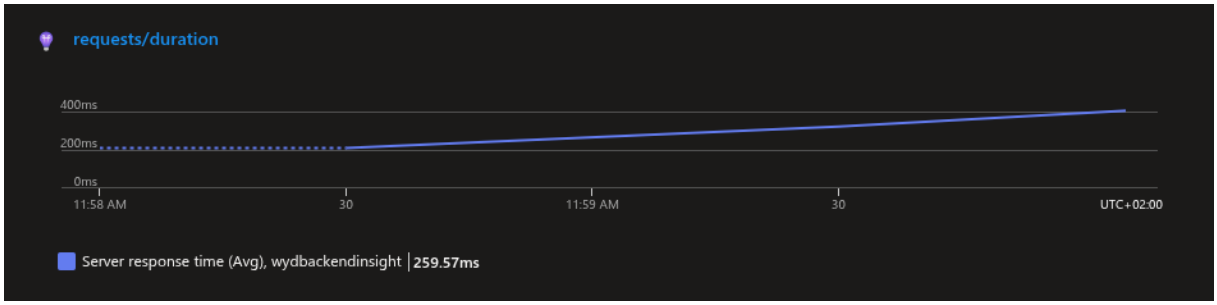

Figura 1.6: Dettaglio della velocità richiesta dal server

Conclusioni

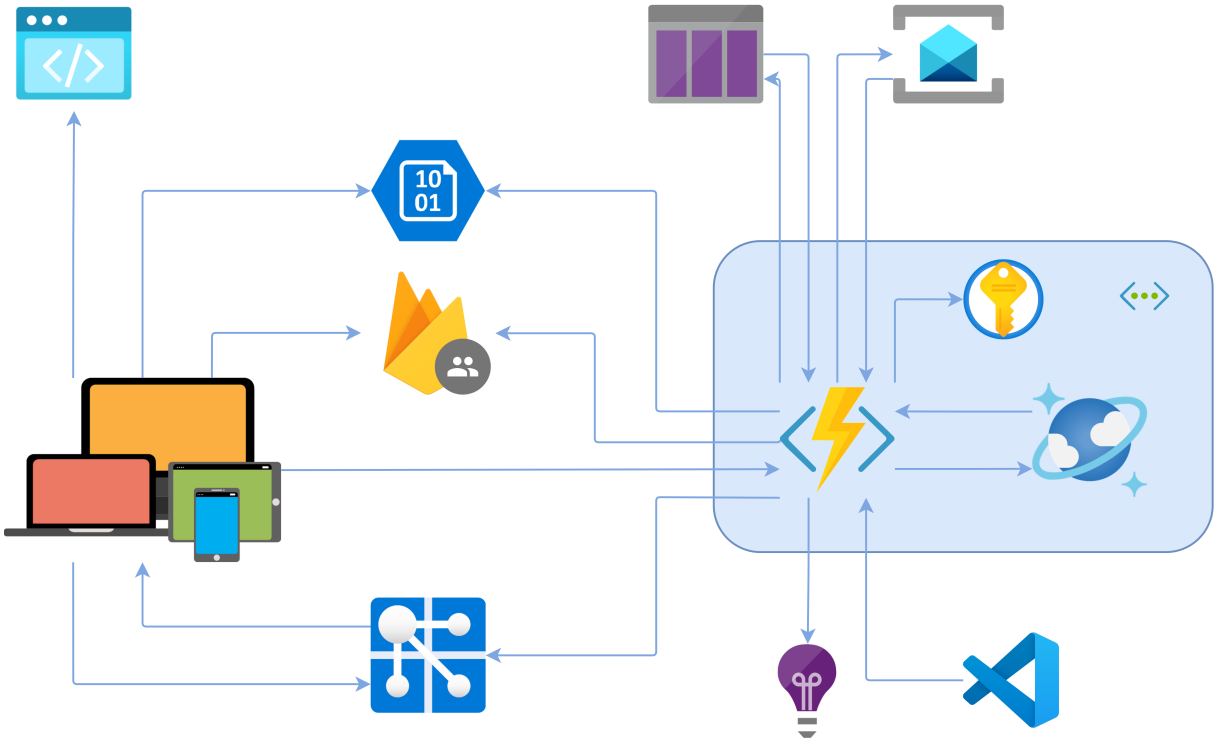

Figura 1.7: Grafico dell'architettura di WYD

1.9 Sviluppi futuri

Gli sviluppi futuri potranno comprendere, in base a decisioni di marketing:

- La visualizzazione degli impegni degli altri profili
- L'implementazione di una chat per ogni gruppo
- Sviluppo di strumenti utili all'organizzazione dei gruppi, quali:
  - form per combinare le disponibilità reciproche
  - appunti condivisi(liste della spesa o note su chi porta cosa)
  - calcolo delle spese compiute da ciascun componente
- La creazione di profili pubblici che possono essere seguiti
- La creazione di eventi pubblici
- Una funzionalità di ricerca degli eventi o dei profili pubblici
- Supporto alla gestione di prenotazione e organizzazione degli eventi, dalle liste di attesa alla vendita dei biglietti
- La possibilità per le aziende di gestire in locale il proprio server e i relativi dati

Fonti bibliografiche e sitografia

Object Management Group, OMG Unified Modelling Language Version 2.5.1, December 2017, https://www.omg.org/spec/UML/2.5.1/PDF

https://learn.microsoft.com/en-us/azure/reliability/reliability-cosmos-db-nosql https://learn.microsoft.gb/azure/cosmos-db/throughput-serverless https://learn.microsoft.com/en-gb/azure/cosmos-db/provision-throughput-autoscale