

ASSIGNMENT 3 - RISK ANALYSIS

Group Name: Group 9

Group Members:

Sanaa Msellek 902325

Christena Attia 894887

Arianna Silvestri 869617

Sezione 1

Per lo studio dell'analisi del rischio è stata scelta come applicazione TripTales, un'applicazione mobile completa per la gestione e documentazione dei viaggi. Il sistema offre un ecosistema integrato per viaggiatori che comprende: autenticazione utente tramite Firebase Auth con opzione login Google, creazione e gestione di diari di viaggio personalizzabili con date, destinazioni e immagini di copertina e un'organizzazione avanzata attraverso fragment dedicati per tappe con geolocalizzazione su mappe Google Maps, gestione budget e spese con tracker visivo, definizione obiettivi di viaggio con progress tracking, e checklist pre-partenza. L'app integra visualizzazioni geospaziali con mappe interattive che evidenziano i paesi visitati utilizzando GeoJSON, un calendario con marcatori visivi per date di viaggio, e un sistema di impostazioni completo con gestione profilo, modalità scura e multilingua. L'architettura MVVM supporta sincronizzazione dati in real-time via Firebase Realtime Database e persistenza locale tramite Room per accesso offline. Il progetto è stato sviluppato tramite Android Studio ed è stato usato GitHub come sistema di versionamento del codice.

Il team del progetto è formato da quattro sviluppatori, di cui una team leader, che si occupa dell'analisi e della progettazione dei requisiti, ed è seguito da una figura CEO, che ha potere decisivo sulle tempistiche e sul budget. Il sistema è in direzione di evoluzione; sono stati raccolti i nuovi requisiti da implementare tramite elicitazione, attraverso un questionario iniziale per gli utenti finali dell'app e attraverso un'intervista con la team leader del progetto. Uno degli obiettivi è di introdurre un modulo di prenotazioni di voli e alloggi con gestione tramite API terze (es. Skyscanner, Booking, Expedia, ecc...) e un modulo per i pagamenti tramite servizi esterni (come PayPal e Stripe), comunicando con questi tramite protocollo https. Si stima di finire ogni modulo in circa tre settimane, comprendendo anche i test.

L'analisi dei rischi è stata condotta dopo l'intervista con la team leader e prima dell'inizio della fase di implementazione. Il suo scopo è di identificare le aree di criticità del progetto, valutare probabilità e impatto dei possibili eventi negativi e definire le azioni di mitigazione e contingenza.

Alcuni rischi preliminari identificati includono i ritardi nella schedule, il fallimento della sincronizzazione e le difficoltà nell'integrazione di API esterne.

Sezione 2

Questa sezione riporta i risultati dell'attività di identificazione dei rischi condotta sul progetto TripTales.

L'identificazione è stata guidata da una checklist strutturata (checklist condivisa su e-learning: "*Sample risk identification checklistFile*"), che copre diverse aree critiche del progetto, tra cui requisiti, progettazione, sviluppo, integrazione, ambiente di sviluppo e vincoli di programma.

Per ciascun item della checklist è stato valutato se la condizione descritta fosse presente e se essa comportasse o meno un rischio. In caso di rischio identificato, viene fornita una motivazione che spiega le possibili conseguenze negative per il progetto; negli altri casi viene giustificato il motivo per cui il rischio non è stato ritenuto rilevante.

Part A – Product Engineering

ID	Tipo	Descrizione	Valutazione (Sì/No)	Comporta un rischio? (Sì/No/NA)	Motivazione
Requirements					
PE1.a	Stabilità	I requisiti cambiano anche mentre il prodotto viene sviluppato?	No	No	I requisiti raccolti riguardano l'implementazione di nuove funzionalità in aggiunta a quelle precedenti già progettate e introdotte. Per questo motivi i requisiti raccolti

					sono definiti come stabili e non portano a rischi legati a modifiche architetturali.
PE1.b	Completezza	Mancano requisiti o i requisiti non sono completamente specificati?	No	No	I requisiti sono stati stilati dopo il feedback iniziale da parte degli utenti e dopo l'intervista con la team leader del gruppo di sviluppo, per cui sono completi. Il fatto che siano completamente specificati permette di evitare rischi come rilavorazioni, ritardi nel rilascio e costi aggiuntivi.
PE1.c	Chiarezza	I requisiti non sono chiari o hanno bisogno di essere interpretati?	Sì	No	I requisiti sono stilati in maniera chiara e non ambigua nel documento, per cui non si verificano rischi legati allo sviluppo di implementazioni sbagliate e a insoddisfazione da parte degli utenti.
PE1.d	Validità	I requisiti portano al prodotto che i consumatori avevano in mente?	Sì	No	I requisiti sono stati stilati anche grazie al questionario iniziale per gli utenti, che ha permesso di capire gli interessi e le preferenze di design; quindi, portano al prodotto richiesto e non verificano rischi come insoddisfazione da parte degli utenti e necessità di rilavorazione.
PE1.e	Fattibilità	I requisiti non sono fattibili sotto un punto di vista analitico?	No	No	Come riportato dalla team leader del gruppo di sviluppo, i nuovi requisiti sono fattibili da un punto di vista analitico, grazie alla struttura modulare dell'architettura dell'app. Per questo motivo non si verificano rischi legati alla fattibilità dei requisiti.
PE1.f	Precedenza	I requisiti specificano qualcosa di nuovo o che l'azienda non ha mai fatto prima?	Sì	Sì	I requisiti specificano qualcosa di nuovo: richiedono di implementare una modalità per la gestione dei pagamenti, il cui sviluppo è nuovo per il team. L'inesperienza degli sviluppatori può portare al verificarsi di ritardi nel rilascio e costi aggiuntivi.
PE1.g	Scala	I requisiti specificano un prodotto più complesso di quello attuale?	Sì	Sì	I nuovi requisiti promuovono la complessità del prodotto, per via della nuova modalità di gestione dei pagamenti da implementare. Questa problematica può portare al verificarsi

					di costi aggiuntivi e modifiche architetturali.
Design					
PE2.a	Funzionalità	Esistono potenziali problemi nel soddisfare i requisiti funzionali?	Sì	Sì	I potenziali problemi nel soddisfare i requisiti funzionali riguardano l'integrazione e la validazione di API di terze parti per gestire le prenotazioni di alloggi e viaggi. Il fallimento della loro integrazione e il loro funzionamento non corretto possono portare al blocco del progetto.
PE2.b	Difficoltà	La progettazione o l'implementazione è difficile da ottenere?	No	No	L'architettura base del progetto è riutilizzata dalla fase di sviluppo precedente; i nuovi requisiti sono fattibili da un punto di vista analitico, per cui non verificano rischi relativi alla progettazione.
PE2.c	Interfacce	Le interfacce interne (hardware e software) sono ben definite e controllate?	Sì	No	Le interfacce interne sono ben definite e sotto controllo, quindi non si prevedono rischi di integrazione, costi aggiuntivi e ritardi nel rilascio.
PE2.d	Performance	I criteri di performance sono espressi chiaramente e sono realizzabili?	No	Sì	La team leader del gruppo degli sviluppatori non cita criteri su cui basare la performance del prodotto e nemmeno gli utenti si sono espressi a riguardo, per cui le prestazioni dell'app non vengono controllate con verifiche chiare.
Pe2.e	Testabilità	Il prodotto è difficile o impossibile da testare?	No	No	Il prodotto non è difficile né impossibile da testare: secondo la team leader la struttura modulare dell'architettura permette di facilitare i test e, infatti, la pianificazione per i test per l'app è programmata per la terza settimana dello sviluppo di un modulo. Per questo motivo, non si verificano rischi legati ai ritardi nel rilascio, né modifiche architetturali.
PE2.f	Vincoli Hardware	Esiste un vincolo stretto sull'hardware target?	No	No	Non vengono presi vincoli stringenti sull'hardware target, per cui l'architettura del software non presenta problematiche legate all'hardware.

PE2.g	Software Non Sviluppato	Il software riutilizzato da usare nel progetto porta a problemi di mancanza di documentazione?	No	No	L'architettura di base è di riutilizzo di quella attuale, per cui il software è ben documentato e il lavoro degli sviluppatori è agevolato. Non si prevedono ritardi nel rilascio del prodotto.
Code and unit test					
PE3.a	Fattibilità	L'implementazione del progetto è difficile o impossibile?	No	No	Come spiega la team leader, i nuovi requisiti sono fattibili da integrare nel progetto, per cui non si prevedono rischi legati alla difficoltà del progetto.
PE3.b	Testing	Il tempo specificato per il testing unitario è adeguato?	No	Sì	La team leader del gruppo degli sviluppatori prevede di eseguire i test per ciascun modulo nella terza settimana dello sviluppo. Le possibili difficoltà nell'integrare le API di terze parti e l'inesperienza degli sviluppatori con l'implementazione dei nuovi requisiti rendono il tempo fissato per il testing inadeguato. Per questo motivo possono emergere difetti non individuati e possono verificarsi ritardi nel rilascio e costi aggiuntivi.
PE3.c	Coding / Implementation				
PE3.c.1	Detailed Design	Esistono problemi con la disponibilità del design dettagliato?	Sì	Sì	Nonostante l'architettura MVVM sia ben definita, manca un design dettagliato per il nuovo "Backend Bridge". Il Backend Bridge funge da strato di astrazione necessario per far comunicare TripTales con i servizi esterni (voli, alloggi e sistemi di pagamento). Senza schemi chiari delle API (ovvero la struttura esatta dei dati in entrata e in uscita, i codici di errore e gli endpoint), lo sviluppo del modulo dei voli, alloggi e pagamenti sarà caotico.
PE3.c.2	Constraints	Esistono problemi di timing, storage o memoria?	Sì	Sì	Dall'attività di "Elicitation" sono emerse tre criticità tecniche principali per TripTales: <ul style="list-style-type: none"> Saturazione Risorse: L'uso di mappe 3D e immagini ad alta risoluzione satura la RAM, causando crash su telefoni

					<p>economici (Android entry-level). Inoltre, l'eccessivo storage occupato dalle foto espone al rischio che Android possa cancellare i dati in cache per liberare memoria persistente, con rischio di perdita di informazioni nel database locale.</p> <ul style="list-style-type: none"> • Instabilità in assenza di rete: La scrittura offline tramite Room risulta attualmente instabile, mettendo a rischio la coerenza dei dati salvati in assenza di rete. • Latenza API: L'integrazione di servizi esterni (voli/hotel) può causare blocchi dell'interfaccia (App Not Responding) se le risposte dei server non vengono gestite con una programmazione asincrona rigorosa.
Integration and Test					
PE4.a	Environment	L'ambiente di integrazione e test è adeguato?	Sì	No	<p>L'ambiente di sviluppo Android Studio offre strumenti integrati avanzati, come l'Android Emulator per simulare le diverse API (es. Google Maps, API Google Login) e Logcat per il debugging in tempo reale. L'integrazione nativa con il framework JUnit consente di eseguire test di integrazione automatizzati in un ambiente controllato, garantendo che i nuovi componenti del Backend Bridge interagiscano correttamente con l'architettura MVVM prima del rilascio.</p>
PE4.b	Product				
PE4.b.1	Facilities	Le risorse sono facilmente ottenibili?	No	Sì	<p>Il team necessita di credenziali sviluppatore per diverse piattaforme (tra cui Booking, Skyscanner, ...). Eventuali ritardi nell'ottenimento di queste chiavi API potrebbero compromettere l'avanzamento dello sviluppo.</p>

PE4.b.2	Time	Le stime dei tempi sono coerenti?	No	Sì	Gli sviluppatori hanno stimato solo 3 settimane per consegnare un modulo funzionante. Questa tempistica è descritta come "minima", il che significa che non c'è margine per errori o bug imprevisti durante l'integrazione, rendendo il fattore della tempistica il rischio principale di questa sezione.
PE4.c	System				
PE4.c.1	Coordination	Esiste un coordinamento dell'integrazione del sistema?	Sì	No	La Team Leader coordina sia le attività di sviluppo che quelle di Integrazione e test del sistema. L'assenza di una figura separata riduce l'overhead organizzativo. La scelta è efficiente e coerente con le dimensioni del gruppo (≈ 5 persone).
Engineering Specialties					
PE5.a	Maintainability	L'implementazione è difficile da capire o mantenere?	No	No	L'adozione dell'architettura MVVM garantisce una netta separazione tra logica di business e UI. Inoltre, la pratica di code review incrociate su GitHub (menzionata dalla Team Leader durante l'intervista), assicura che il codice sia standardizzato, documentato e comprensibile a tutto il team, facilitando la manutenzione futura.
PE5.b	Reliability	I requisiti di affidabilità o disponibilità sono difficili da soddisfare?	Sì	Sì	Il rischio riguarda la possibile perdita di dati critici durante il processo di prenotazione. L'attuale instabilità della scrittura offline in Room, combinata con eventuali fallimenti del Backend Bridge, mette a rischio l'integrità delle transazioni. Se la sincronizzazione non fosse perfetta, l'utente potrebbe perdere la conferma di voli o hotel già pagati.

PE5.c	Safety	I requisiti di sicurezza sono fattibili e dimostrabili?	Sì	Sì	Poiché l'app integra mappe 3D e suggerisce itinerari, il rischio deriva da errori non intenzionali nel software: un bug nel sistema di navigazione o un crash improvviso dell'app (dovuto alla saturazione della RAM, già identificata) mentre l'utente si trova in zone isolate, può causare situazioni di pericolo fisico impedendo l'accesso a mappe, contatti di emergenza o informazioni sui trasporti.
PE5.d	Security	I requisiti di sicurezza sono più stringenti della pratica corrente o dell'esperienza del team?	Sì	Sì	La gestione di transazioni finanziarie e dati sensibili impone requisiti di sicurezza più elevati rispetto all'esperienza del team. HTTPS e crittografia non sono mai stati implementati, esponendo il sistema a possibili violazioni o perdita di dati.
PE5.e	Human Factors	Il sistema è difficile da usare a causa di una scarsa definizione dell'interfaccia?	No	No	L'app utilizza componenti standard (Material Design) e Fragment dedicati che rendono la navigazione familiare. L'integrazione di Google Login semplifica l'accesso, riducendo al minimo la barriera d'ingresso per l'utente.
PE5.f	Specifications	La documentazione è adeguata a progettare, implementare e testare il sistema?	No	Sì	Nonostante i requisiti di alto livello siano definiti, manca una specifica tecnica dettagliata per il nuovo Backend Bridge e per l'integrazione delle API esterne. Queste lacune rendono difficile per gli sviluppatori e per il tester coordinarsi efficacemente.

Part B - Development Environment

Development Process					
DE1.a	Suitability	Il modello di processo è adatto a soddisfare i requisiti del progetto?	Sì	No	Il team utilizza un approccio Scrum con rilasci rapidi (sprint di 3 settimane). Questo modello è ideale per un MVP (Minimum Viable Product) come il modulo prenotazioni, poiché permette di adattarsi velocemente ai feedback raccolti dagli utenti.

DE1.b	Process Control	I siti di sviluppo distribuiti sono coordinati?	Sì	No	Anche se i membri del gruppo lavorano da remoto, il coordinamento è garantito da meeting periodici e canali di comunicazione immediata, assicurando che tutti siano allineati sull'avanzamento dei singoli fragment.
DE1.c	Familiarity	Lo staff è a conoscenza del processo da seguire?	Sì	No	Il workflow (Git flow, architettura MVVM) è stato discusso e condiviso all'inizio. Ogni membro sa come gestire i branch e come integrare le proprie modifiche nel database Firebase/Room senza sovrascritture.
DE1.d	Product Control	Esistono meccanismi per il controllo delle modifiche?	Sì	No	L'uso di GitHub con un sistema di Pull Request e review incrociate (come citato dalla Team Leader durante l'intervista) permette il controllo su ogni riga di codice aggiunta, prevenendo l'introduzione di bug nel ramo principale (main).

Development System

DE2.a	Capacity	Workstation, potenza, memoria e storage sono sufficienti?	No	Sì	Nonostante il team disponga dell'hardware necessario, l'uso combinato di Android Studio, emulatori e il Backend locale mette sotto stress i laptop personali. Questo rallenta drasticamente i tempi di build e test, riducendo la produttività.
DE2.b	Suitability	Il sistema di sviluppo supporta tutte le fasi, attività e funzioni?	Sì	No	L'ecosistema scelto (Android Studio, Firebase e GitHub) copre l'intero ciclo di vita del software. Dalla progettazione della UI alla gestione del database, il team dispone di un set di strumenti che non presenta lacune funzionali, facilitando il passaggio tra le diverse fasi del progetto.
DE2.c	Usability	Il sistema di sviluppo è facile da usare?	Sì	No	L'adozione di Android Studio garantisce un ambiente di lavoro omogeneo, altamente assistito da strumenti di analisi automatica e perfettamente compatibile con le tecnologie scelte (Firebase, Room), annullando i rischi legati a difficoltà tecniche dello strumento di sviluppo.
DE2.d	Familiarity	Il team di progetto ha familiarità con il sistema di sviluppo e con il cliente?	Sì	No	Il team dispone di una buona familiarità con il sistema grazie allo studio preliminare e al confronto diretto con gli sviluppatori. Il cliente è chiaramente identificato e coinvolto attivamente tramite interviste strutturate. La comunicazione

					continua con gli stakeholder riduce il rischio di incomprensioni e rende il fattore non critico.
DE2.e	Reliability	Il sistema soffre di bug software, tempi di inattività e backup integrato insufficiente?	No	No	Gli strumenti utilizzati (Android Studio, Git, Firebase) sono piattaforme mature e affidabili. Git garantisce versionamento e ripristino, mentre Firebase offre alta disponibilità e backup automatici. Il rischio di downtime o perdita dati è contenuto.
DE2.f	System Support	Esiste un supporto tempestivo da parte di esperti o fornitori per il sistema?	Sì	No	Tutti i principali strumenti utilizzati dispongono di un ampio ecosistema di supporto: documentazione ufficiale, community attive, forum tecnici, guide, aggiornamenti continui e assistenza da parte dei fornitori (Google per Android Studio e Firebase). Questo riduce fortemente il rischio di blocchi tecnici o ritardi dovuti a mancanza di competenze o assistenza.
DE2.g	Deliverability	Sono stati definiti i criteri di accettazione e i requisiti di consegna per il sistema da rilasciare al cliente?	Sì	No	I requisiti di accettazione dovrebbero emergere dall'intervista con la CEO in quanto è la figura chiave per la definizione degli obiettivi di business, delle priorità di progetto e dei vincoli organizzativi.
Management Process					
DE3.a	Planning	a) La pianificazione è tempestiva? b) Il piano include: • Approccio tecnico • Piano di emergenza	a) No b) No	Sì	La mancanza di una pianificazione tempestiva e di un piano tecnico e di emergenza aumenta il rischio di ritardi e di gestione inefficace degli imprevisti.
DE3.b	Project Organization	I ruoli e le relazioni gerarchiche sono chiari?	Sì	No	I ruoli degli stakeholder sono chiaramente definiti: il cliente/CEO ha responsabilità decisionali e strategiche, gli sviluppatori sono responsabili dell'architettura e dell'implementazione tecnica. Le relazioni tra gli stakeholder sono agevolate da meeting periodici.
DE3.c	Management Experience	I manager hanno esperienza in • Sviluppo software • Gestione software • Dominio applicativo • Processo di sviluppo • Programmi di grandi dimensioni	No	Sì	Il team non dispone di figure con esperienza manageriale consolidata nella gestione di grandi progetti software e nei processi di sviluppo strutturati; ciò può influire sulla capacità di pianificazione, coordinamento e controllo delle attività.
DE3.d	Program Interfaces	C'è una scarsa interazione/comunicazione con il	No	No	I canali di comunicazione con il cliente e con gli stakeholder principali sono chiari e diretti (tramite interviste strutturate e questionari di

		cliente, con altri appaltatori, con i responsabili senior e/o con i colleghi di pari livello?			feedback) riducendo il rischio di incomprensioni e problemi di coordinamento.
Management Methods					
DE4.a	Configuration Management	Le procedure di controllo delle versioni, inclusi i siti di installazione, sono adeguate?	Si	No	L'uso di Git garantisce un controllo di versione affidabile, tracciabilità delle modifiche e gestione sicura delle release, rendendo adeguate le procedure di configuration management.
Work Environment					
DE5.a	Communication	C'è scarsa consapevolezza della missione o degli obiettivi e una scarsa comunicazione delle informazioni tecniche tra colleghi e responsabili?	No	No	Il gruppo ha definito chiaramente la visione strategica del progetto: trasformare TripTales da un'applicazione di documentazione viaggi a un ecosistema end-to-end che integri pianificazione, prenotazione e documentazione. Gli obiettivi sono ben articolati e allineati con i feedback degli utenti e le priorità strategiche della CEO. Supponendo che si lavori con un processo agile non dovrebbero esserci problemi di comunicazione degli obiettivi.

Part-C - Program Constraints

Resources					
PC1.a	Schedule	La pianificazione è inadeguata o instabile?	Sì	Sì	La pianificazione del progetto risulta poco realistica e soggetta a variazioni frequenti. Questo può causare ritardi nelle attività e difficoltà nel rispettare le scadenze prefissate. Un calendario instabile rende complesso il coordinamento del team e aumenta il rischio di consegne tardive o incomplete.
PC1.b	Staff	Il personale manca di esperienza, conoscenza del dominio, competenze o è numericamente insufficiente?	Sì	Sì	Il team possiede buone competenze nello sviluppo Android e nell'architettura dell'applicazione, ma non ha esperienza diretta nella gestione di sistemi di pagamento, sicurezza delle transazioni e integrazione di provider finanziari. Questa mancanza di competenze specifiche può introdurre rischi tecnici, di sicurezza e di stima dei tempi, rendendo necessario supporto esterno o una fase di apprendimento aggiuntiva.
PC1.c	Hardware/Software	Le attività di valutazione e approvvigionamento di hardware e software causano rischi?	No	No	Il progetto utilizza strumenti software già disponibili e consolidati (Android Studio, Git, Firebase) e non richiede l'acquisto di hardware dedicato o licenze particolari. L'infrastruttura cloud è gestita dal fornitore, riducendo tempi, costi e rischi legati all'approvvigionamento.

PC1.d	Facilities	Le strutture sono adeguate a costruire e consegnare il prodotto?	Sì	No	Il team opera in modalità remota utilizzando strumenti collaborativi consolidati (repository condivisi, piattaforme di comunicazione online e ambienti di sviluppo locali). Non sono richieste strutture fisiche dedicate (uffici, laboratori o spazi comuni); le risorse disponibili sono adeguate allo sviluppo e alla consegna del prodotto.
Contract					
PC2.c	Dependencies	Il programma presenta dipendenze da prodotti o servizi esterni?	Sì	No	Il progetto Trip Tales presenta dipendenze da servizi e componenti esterni, come API di terze parti per la gestione di mappe, geolocalizzazione, contenuti turistici o servizi cloud per l'archiviazione dei dati. Il corretto funzionamento dell'applicazione dipende quindi dalla disponibilità, stabilità e continuità di tali servizi. Il progetto fa uso di diverse API Google che risultano essere stabili e bene mantenute.
Program Interfaces					
PC3.a	Customer	Esistono problemi con il cliente, come cicli di approvazione lunghi, scarsa comunicazione o competenze di dominio inadeguate?	No	No	Il cliente collabora in modo efficace e fornisce feedback chiari e tempestivi. I processi di approvazione sono ragionevoli e la comunicazione risulta adeguata. Di conseguenza, l'interazione con il cliente non introduce rischi significativi per il progetto.

Sezione 3

In questa sezione vengono presentate le stime di probabilità, perdita ed esposizione al rischio per ciascun rischio identificato nella sezione precedente.

Per ogni rischio sono riportati:

- una stima della probabilità ($P(UO) \rightarrow$ “*Probability of Unsatisfactory Outcome*”) di occorrenza,
- una stima della perdita ($L(UO) \rightarrow$ “*Loss to the parties affected if the Outcome is Unsatisfactory*”) associata all'evento negativo,
- il valore di Risk Exposure (RE), calcolato come prodotto di probabilità e perdita.

Le stime sono espresse tramite valori ordinali e sono motivate sulla base delle caratteristiche del progetto, dell'esperienza del team e delle informazioni raccolte durante l'analisi, con l'obiettivo di fornire una valutazione coerente e comparabile dei rischi individuati. Per l'assegnazione dei valori è stato studiato il caso del paper condiviso su e-learning (“*Software Risk Management: Principles and Practices*”), ma essendo il caso di un progetto differente dal nostro (“*SATELLITE EXPERIMENT SOFTWARE*”) in alcuni punti ci siamo riservati a fare una valutazione diversa.

ID	Categoria	P(UO)	L(UO)	RE	Motivazioni
Requirements					
PE1.f	Esperienza con i requisiti	5	7	35	L'esperienza degli sviluppatori è inadeguata ad affrontare lo sviluppo dei nuovi requisiti, come l'implementazione del modulo per la gestione dei pagamenti; l'evento negativo si ripercuote su costi aggiuntivi e ritardi nel rilascio.
PE1.g	Complessità del prodotto	6	7	42	I nuovi requisiti portano a una complessità maggiore nello sviluppo del progetto, rispetto al precedente, per cui la probabilità è stimata con valore medio-alto. L'evento negativo si ripercuote su costi aggiuntivi e ritardi.
Design					
PE2.a	Validità dei requisiti funzionali	6	9	54	La probabilità di riscontrare problematiche nella realizzazione dei requisiti funzionali dipende da fattori esterni, come ritardi o difficoltà nell'ottenimento di chiavi per l'integrazione di API di terze parti, per la gestione delle prenotazioni di alloggi e voli. Inoltre, una volta ottenute le chiavi bisogna stabilirne il corretto funzionamento. Per progetti che riportano le stesse condizioni, si stima questo valore come medio-alto. La perdita è alta perché l'evento negativo dato dal rischio porta al fallimento su una parte funzionale del progetto.
PE2.d	Criteri di performance	5	7	35	I criteri di performance non sono definiti chiaramente per questo progetto, per cui la probabilità è medio-bassa. L'evento negativo porta a implementazioni errate e rilavorazioni; la perdita dipende dalle capacità del team di sviluppo di saper gestire modifiche in fase avanzata del

					progetto, per cui il valore stimato è medio-alto.
Code and Unit Test					
PE3.b	Tempistiche per il testing	7	7	49	Le tempistiche per il testing sono programmate dalla team leader in corrispondenza della terza settimana dello sviluppo di ciascun modulo. I ritardi dipendono dall'esperienza del team con l'implementazione del modulo, per cui la probabilità è stimata come medio-alta. Il rischio impatta solo sulle tempistiche di rilascio e sulle modifiche architetturali, per cui la perdita è stimata come medio-alta.
PE3.c.1	Mancanza di design dettagliato	7	7	49	L'assenza di un design dettagliato per le nuove API rende lo sviluppo disorganizzato e suscettibile a modifiche iterative non pianificate. Poiché la fase di sviluppo è imminente, la probabilità che il rischio si concretizzi è medio-alta. L'impatto è discretamente alto perché richiede un riadattamento del lavoro con un conseguente rallentamento importante nell'avanzamento.
PE3.c.2	Saturazione delle risorse e Crash	5	9	45	Il rischio è già stato osservato attraverso crash su dispositivi con risorse limitate. La probabilità che si verifichi è medio-bassa e l'impatto è severo perché comporterebbe la perdita di utenti e la conseguente disinstallazione dell'applicativo.
Integration and Test					
PE4.b.1	Ritardo per le credenziali API	4	8	32	Il rilascio delle credenziali API dipende da terze parti (es. Booking, SkyScanner, ...) e costituisce un vincolo esterno al progetto. La probabilità è pressoché bassa e l'impatto è significativo, poiché eventuali ritardi comporterebbero un

					blocco diretto delle attività di sviluppo e test sul modulo interessato.
PE4.b.2	Ritardi nella Schedule	8	7	56	Durante l'intervista la Team Leader ha affermato che il primo modulo funzionante sarebbe stato realizzato in 3 settimane lavorative. La stima non tiene conto di margini di errore e dunque è chiaramente irrealistica. La probabilità di fallimento nel riuscire a consegnare il modulo delle prenotazioni in tempo è molto alta e l'impatto è rilevante, con ripercussioni sull'intero piano di sviluppo.
Engineering Specialties					
PE5.b	Fallimento Sincronizzazione	6	9	54	La funzionalità di scrittura offline non è al momento disponibile, come cita la Team Leader. La probabilità del rischio è elevata a causa dell'assenza di garanzie sul corretto allineamento dei dati in fase di sincronizzazione. L'impatto è grave, poiché la perdita di prenotazioni o di dati finanziari comprometterebbe direttamente la fiducia dell'utente e l'affidabilità del sistema.
PE5.c	Rischio incolumità dell'utente	5	7	35	Fornire indicazioni errate in zone isolate o mancare segnalazioni di emergenza compromette l'incolumità fisica dell'utente. La complessità di garantire dati affidabili in tempo reale è unita alla possibile indisponibilità dell'app per saturazione risorse (RAM). In conclusione, la probabilità è medio-bassa perché non è comune ritrovarsi in situazioni estreme o critiche, ma l'impatto, nel caso si verificasse, è medio-alto perché comprometterebbe la sicurezza fisica dell'utente.
PE5.d	Inesperienza nella gestione di dati sicuri	5	9	45	L'inesperienza del team nella gestione di transazioni finanziarie e protocolli HTTPS rappresenta un rischio critico per l'integrità dei dati. Nonostante la probabilità sia medio-bassa, l'impatto è categorizzato come molto alto poiché una vulnerabilità nei pagamenti o una fuga di dati sensibili comporterebbe gravi ripercussioni

					legali e la perdita totale di fiducia da parte degli utenti.
PE5.f	Mancanza Specifiche Tecniche	6	6	36	La mancanza di specifiche tecniche per il Backend Bridge costituisce un rischio legato all'integrazione delle API. Questo scenario comporta un aumento dei bug strutturali e della necessità di chiarimenti last minute. L'impatto è di natura organizzativa e tecnica, in quanto introduce ritardi negli sprint e può compromettere la stabilità del modulo di prenotazione.
Development Process					
Nella sezione "Development Process" non ci sono rischi					
Development System					
DE2.a	Lentezza Hardware/Build	6	4	24	L'utilizzo di hardware personale non adeguato agli strumenti professionali rallenta le attività di compilazione, test e debugging, generando un rallentamento significativo nel ciclo di sviluppo. Il fenomeno è già osservabile e non determina blocchi del progetto, ma riduce la produttività, con conseguente accumulo di micro-ritardi nei singoli sprint. L'impatto è pertanto basso: non interrompe l'avanzamento del lavoro, ma rende più complesso sostenere le scadenze, soprattutto nelle fasi più critiche.
DE3.a.a	Pianificazione non realistica	8	7	56	La stima dei tempi è molto approssimativa e non realistica perché si tende a partire con stime ottimistiche e senza analisi approfondita, per eccesso di fiducia o pressioni commerciali, portando quasi sempre a ritardi cumulativi. Questo porta ad avere un impatto medio-alto: ritardi a cascata, stress team, possibile superamento budget.
DE3.a.b	Imprevisti o Perdita improvvisa di risorse chiave	3	9	27	Gli imprevisti sono rari ma possibili; per esempio, se un membro chiave del team lascia il progetto o si ammala gravemente, non c'è un piano per sostituire immediatamente le sue conoscenze critiche. Questo porta ad avere un valore di impatto molto alto,

					data dalla perdita di conoscenze preziose, ritardi forti nel breve termine e possibile effetto domino. La Team Leader ha doppio ruolo: manager e sviluppatore, perderla può impattare gravemente il progetto.
DE3.a.c	Sovrapposizione dei ruoli	4	7	28	La combinazione del ruolo di manager e di sviluppatore nello stesso Team Leader genera un conflitto di priorità che riduce l'efficacia in entrambe le funzioni. Questa duplicità distrae dalla gestione strategica del progetto e, contemporaneamente, compromette la qualità e la concentrazione nel lavoro tecnico, aumentando il rischio di ritardi, errori e indebolimento della guida del team.
DE3.c	Inesperienza manageriale del Team Leader del progetto	6	6	36	La Team Leader si trova a ricoprire per la prima volta un ruolo di gestione, essendo stata promossa principalmente per il suo valore tecnico. Questa mancanza di esperienza gestionale diretta, combinata con la complessità intrinseca di un progetto basato su tecnologie innovative, rappresenta un rischio significativo. L'impatto è medio-alto perché l'inesperienza manageriale agisce come un moltiplicatore per altri rischi (DE3.a.a e PC1.a).
PC1.a	Consegne tardive	8	6	48	La probabilità è elevata poiché è molto probabile che il calendario pianificato dal team di sviluppo sia instabile e poco realistico, questo rende quasi certa la consegna tardiva o incompleta del prodotto. L'impatto risulta mediamente significativo in quanto tale instabilità di pianificazione si traduce sistematicamente in continue riprogrammazioni e ritardi cumulativi che provocano insoddisfazione nel cliente.
PC1.b	Gap nelle competenze (gestione pagamenti)	5	7	35	La probabilità di avere incompetenze in ambiti specialistici (es. integrazione sistemi di pagamento) non è raro, soprattutto se il team non ha maturato precedente esperienza diretta. Una valutazione delle skill in fase di pianificazione può ridurre la probabilità, ma non eliminarla completamente.

					L'assenza della competenza comporta ritardi per formazione, costi aggiuntivi imprevisti, potenziale degrado della qualità dell'implementazione, e possibile esposizione a rischi di sicurezza o conformità. L'impatto è medio-alto ma gestibile senza compromettere la fattibilità complessiva del progetto.
--	--	--	--	--	--

Sezione 4

Questa sezione elenca i 10 rischi più critici per TripTales, ordinati in base alla loro *Risk Exposure* ($RE = P \times L$).

L'obiettivo di questa classifica è identificare immediatamente le minacce su cui il team deve concentrare le proprie risorse limitate per evitare il fallimento del progetto.

Classifica	ID	Descrizione del rischio	P	L	RE
1°	PE4.b.2	Ritardi nella Schedule	8	7	56
2°	DE3.a.a	Pianificazione non realistica	8	7	56
3°	PE5.b	Fallimento Sincronizzazione	6	9	54
4°	PE2.a	Validità dei requisiti funzionali	6	9	54
5°	PE3.b	Tempistiche per il testing	7	7	49
6°	PE3.c.1	Mancanza di design dettagliato	7	7	49
7°	PC1.a	Consegne tardive	8	6	48
8°	PE5.d	Inesperienza nella gestione di dati sicuri	5	9	45
9°	PE3.c.2	Saturazione delle risorse e Crash	5	9	45
10°	PE1.g	Complessità del prodotto	6	7	42

Analisi dei Dati

Dalla classifica emergono tre chiare aree di criticità:

- **Dominanza del Fattore Temporale (Rischi 1, 2, 5, 7):**

I primi due posti della classifica sono occupati da rischi legati alla pianificazione e ai ritardi ($RE=56$). Questo evidenzia come il vincolo delle 3 settimane sia percepito come la minaccia più probabile e pericolosa. Le contromisure dovranno puntare drasticamente sullo *scope reduction* (ovvero il taglio delle funzionalità non essenziali) per proteggere la data di consegna.

- **Gravità dell'Impatto Tecnico (Rischi 3, 4, 8):**

Sebbene abbiano una probabilità inferiore rispetto ai ritardi, questi rischi presentano il valore di perdita massimo (L=9). Un fallimento nella sincronizzazione o una violazione dei dati finanziari non comporterebbe solo un ritardo, ma il fallimento qualitativo del prodotto. Per questi, il piano di gestione dovrà prevedere rigorosi protocolli di sicurezza e test di integrazione continui.

- **Correlazione tra Design e Qualità (Rischi 6, 9, 10):**

La mancanza di un design dettagliato e la complessità del prodotto alimentano direttamente il rischio di non ottenere un prodotto completo e affidabile.

Sezione 5

Questa sezione definisce i protocolli operativi per la gestione dei dieci rischi più importanti. L'obiettivo è integrare l'analisi dei rischi con azioni realizzabili che ne riducano l'incidenza durante lo sviluppo. Per ciascun rischio della "Top 10", è stata redatta una tabella comparativa che mette a confronto diverse strategie di mitigazione. Il processo decisionale si avvale del parametro Risk-Reduction Leverage (RRL), calcolato secondo la formula:

$$RRL = \frac{RE_{\{before\}} - RE_{\{after\}}}{Cost}$$

Dove la Risk Exposure (RE) è data dal prodotto tra Probabilità (P) e Impatto della Perdita (L).

L'utilizzo dell'RRL permette di identificare la contromisura più efficiente, bilanciando l'efficacia della riduzione del rischio rispetto ai costi e allo sforzo implementativo richiesti. Ogni tabella dettaglia obiettivi, responsabilità, approcci e risorse necessarie, garantendo che ogni scelta sia orientata al valore del prodotto finale.

Il piano di gestione risultante è articolato secondo due componenti complementari:

- **Management Strategy (Prevenzione):** comprende le azioni proattive da applicare fin da subito per ridurre la probabilità (P) che il rischio si presenti, intervenendo sulle cause originarie.
- **Contingency Strategy (Emergenza):** definisce le misure da attivare qualora il rischio si concretizzi, con l'obiettivo di contenerne l'impatto (L). Tale strategia viene avviata solo a seguito di un trigger specifico, che segnala la transizione del rischio da potenziale a reale.

Management Plans

1° – PE4.b.2 Ritardi nella schedule

I ritardi nella schedule compromettono il rilascio del prodotto alla scadenza stabilita.

Elemento	Contromisura 1 – Revisione iniziale	Contromisura 2 – Task in parallelo
Obiettivo	Ottenere una revisione iniziale della pianificazione del progetto grazie all'aiuto di figure professionali esperte	Eseguire in parallelo task che erano originariamente pianificati in sequenza per ridurre il tempo di implementazione del progetto
Risultati e traguardi	Alla fine della riunione con gli esperti, il risultato è un calendario che comprende la pianificazione iniziale del progetto coerente con il carico di lavoro e stime delle tempistiche e dei traguardi intermedi	Il risultato è un documento con la schedule e la suddivisione dei task in parallelo assegnati ai rispettivi sviluppatori
Responsabilità	CEO con un team qualificato di esperti	Team leader e sviluppatori

Approccio	Si programma una riunione iniziale prima della fase di implementazione, con esperti qualificati che valutano e correggono le stime iniziali per le tempistiche e stabiliscono scadenze per i traguardi futuri, in vista del rilascio dell'app	Si programma l'inizio dello sviluppo di un modulo quando un altro è ancora in corso di progettazione
Risorse	Tempo della CEO, costi aggiuntivi per assumere le figure professionali	Sforzo del team per svolgere un carico di lavoro più pesante
P	8 → 6 (la probabilità diminuisce perché si lavora sulla prevenzione dei ritardi)	8 → 6 (la probabilità diminuisce perché si lavora sulla prevenzione dei ritardi)
L	7 (l'impatto rimane uguale)	7 (l'impatto rimane uguale)
Nuovo RE	56 → 42	56 → 42
Costo	5 (il costo non è elevato perché si richiedono costi e tempi aggiuntivi per un periodo limitato)	8 (il costo è elevato perché la contromisura porta a un maggiore carico di lavoro per il team e a una maggiore coordinazione)
RRL	2.8	1.75

Scelta finale: la **contromisura 1** ha un RRL molto più alto, quindi è la strategia più efficiente.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):**
Si programma una riunione iniziale tra CEO ed esperti prima della fase di sviluppo. Si revisionano le tempistiche e si stilano delle nuove scadenze.
- **Contingency Strategy (Piano di Emergenza):**
Qualora si presentino ritardi nella schedule, la CEO del progetto coinvolgerà membri esperti per la revisione della pianificazione e delle attività da svolgere, per l'analisi di tutte le attività critiche e per l'identificazione dei possibili punti di ritardo. Verranno ridefiniti gli obiettivi e le priorità sulle funzionalità di sviluppo di base, bloccando ulteriori modifiche superflue, in modo da rilasciare un prodotto minimo funzionante nei tempi richiesti.

2° – DE3.a.a Pianificazione non realistica

La pianificazione non realistica compromette lo sviluppo del progetto e il rilascio alla scadenza stabilita, impattando su tempi e costi aggiunti.

Elemento	Contromisura 1 – Revisione iniziale	Contromisura 2 – Monitoraggio periodico
Obiettivo	Ottenere una revisione iniziale della pianificazione del progetto grazie all'aiuto di figure professionali esperte	Monitorare periodicamente la realizzabilità degli obiettivi
Risultati e traguardi	Alla fine della riunione con gli esperti, il risultato è un calendario che comprende la pianificazione iniziale del progetto coerente con il carico di lavoro e stime delle tempistiche e dei traguardi intermedi	Il risultato è un calendario aggiornato con le tempistiche più coerenti

Responsabilità	CEO con un team qualificato di esperti	CEO e team degli sviluppatori
Approccio	Si programma una riunione iniziale prima della fase di implementazione, con esperti qualificati che valutano e correggono le stime iniziali per le tempistiche e stabiliscono scadenze per i traguardi futuri, in vista del rilascio dell'app	Nelle riunioni già programmate si aggiunge come attività la valutazione delle tempistiche future e della fattibilità dei prossimi obiettivi, in vista delle scadenze fissate
Risorse	Tempo della CEO, costi aggiuntivi per assumere le figure professionali	Non richiede tempo né costi aggiuntivi, oltre a quelli già sostenuti per le riunioni
P	8 → 6 (la probabilità diminuisce moderatamente perché si lavora sulla prevenzione di una pianificazione non adeguata solo all'inizio della fase di sviluppo)	8 → 5 (la probabilità diminuisce significativamente perché si lavora sulla prevenzione di una pianificazione non realistica a ogni controllo periodico)
L	7 (l'impatto rimane uguale)	7 (l'impatto rimane uguale)
Nuovo RE	56 → 42	56 → 35
Costo	5	5 (contenuto e distribuito nel tempo)
RRL	2.8	4.2

Scelta finale: la **contromisura 2** ha un RRL molto più alto, quindi è la strategia più efficiente.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):**

Nella riunione iniziale, prima della fase di progettazione, si valutano le tempistiche future e la realizzabilità degli obiettivi di progetto. Si programmano come di consueto le riunioni periodiche di confronto e organizzazione; in queste verranno monitorate di volta in volta la fattibilità degli obiettivi successivi e le loro relative scadenze.

- **Contingency Strategy (Piano di Emergenza):**

Qualora si presentino ritardi nella schedule, la CEO del progetto insieme al team di sviluppo analizzerà tutte le attività critiche e ridefinirà gli obiettivi e le priorità sulle funzionalità di sviluppo di base, bloccando ulteriori modifiche superflue, in modo da rilasciare un prodotto minimo funzionante nei tempi richiesti.

3°- PE5.b Fallimento sincronizzazione

Il fallimento della sincronizzazione può portare alla perdita di prenotazioni o di dati finanziari e può compromettere direttamente la fiducia dell'utente e l'affidabilità del sistema.

Elemento	Contromisura 1 - Sistema di gestione degli errori	Contromisura 2 – Correzione manuale
Obiettivo	Registrazione degli errori e notifiche automatiche agli sviluppatori quando si registra un errore	Procedura manuale di risincronizzazione dei dati critici da parte del team di sviluppo
Risultati e traguardi	Ogni volta che si verifica un fallimento nella sincronizzazione, il risultato è ottenere la registrazione dell'errore e dei dati critici e notifiche di allarme agli sviluppatori,	Ogni volta che si verifica un fallimento nella sincronizzazione, il risultato è ottenere un ripristino manuale dei dati critici da parte del team di sviluppo

	attraverso un sistema automatico di gestione dei fallimenti	
Responsabilità	Team di sviluppo	Team di sviluppo
Approccio	Si implementa un sistema automatico di gestione degli errori in grado di registrare gli eventi di fallimento nella sincronizzazione e di notificare l'incidente al team di sviluppo	Si implementa un database di backup; il team di sviluppo si occupa di ripristinare i dati critici attraverso il database in caso di incidente
Risorse	Costi e tempo aggiuntivi per l'implementazione e la manutenzione del sistema	Costi aggiuntivi per l'implementazione e la manutenzione del database di backup; sforzo e tempo da parte del team di sviluppo per ripristinare i dati critici
P	6 (la probabilità rimane uguale)	6 (la probabilità rimane uguale)
L	9 → 7 (si ha maggiore controllo sull'accadimento degli eventi)	9 → 7 (si ha maggiore controllo sul ripristino dei dati)
Nuovo RE	54 → 42	54 → 42
Costo	6	6
RRL	2	2

Scelta finale: le contromisure sono equivalenti dal punto di vista della riduzione del livello del rischio. Sono infatti complementari: prese insieme danno un'adeguata prospettiva di azione per ridurre il rischio.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):**

Si implementa un sistema di gestione degli errori che rileva fallimenti della sincronizzazione, registrando l'errore e i dati critici e notificando gli sviluppatori dell'accaduto. Si implementa inoltre in parallelo un database di backup.

- **Contingency Strategy (Piano di Emergenza):**

Qualora si verificasse un fallimento nella sincronizzazione dei dati, il team di sviluppo verrà allarmato dal sistema e provvederà a identificare i dati critici; disabiliterà temporaneamente le operazioni sui dati coinvolti; tramite il database di backup ripristinerà manualmente i dati coinvolti e infine avvertirà l'utente dell'incidente.

4° – PE2.a Validità dei requisiti funzionali

L'invalidità dei requisiti funzionali dipende anche da fattori esterni, tra cui l'integrazione e la validazione di API terze per la gestione delle prenotazioni. Il rischio può portare al fallimento di una parte funzionale del prodotto.

Elemento	Contromisura 1 – Definizione di requisiti alternativi	Contromisura 2 – Test per la validazione
Obiettivo	Definizione di requisiti alternativi in caso di indisponibilità delle API	Test per validare l'integrazione delle API nel progetto
Risultati e traguardi	In caso di indisponibilità delle API terze, il risultato è un progetto con funzionalità minime alternative e analoghe.	In caso di integrazione di API terze, a fine implementazione dei moduli appositi si

	Il risultato è la documentazione con nuovi requisiti funzionali.	stabilisce il loro corretto funzionamento, per evitare errori Il risultato è l'ambiente di produzione (sandbox) realizzato.
Responsabilità	Team leader del gruppo degli sviluppatori	Team di sviluppo
Approccio	La team leader definisce nuovi requisiti funzionali che possano sostituire i requisiti che richiedono l'integrazione di API esterne.	Gli sviluppatori realizzano un ambiente di produzione dedicato all'implementazione e all'esecuzione dei test sulle API (sandbox).
Risorse	Costi e tempo aggiuntivi per l'analisi e la progettazione da parte della team leader.	Costi e tempo aggiuntivi per l'implementazione e la manutenzione dalla sandbox.
P	6 (la probabilità rimane uguale)	6 (la probabilità rimane uguale)
L	9 → 7 (si riduce l'impatto del rischio per la progettazione di altri requisiti analoghi)	9 → 7 (si riduce l'impatto del rischio perché si validano le API)
Nuovo RE	54 → 42	54 → 42
Costo	5	7
RRL	2.4	1.7

Scelta finale: la **contromisura 1** ha un RRL molto più alto, quindi è la strategia più efficiente.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):**
Verificare in anticipo limiti, disponibilità, tempi di risposta per integrare le API; individuare e valutare provider secondari prima di iniziare lo sviluppo.
- **Contingency Strategy (Piano di Emergenza):**
Qualora i requisiti funzionali legati all'integrazione delle API esterne scelte non potessero essere soddisfatti, la team leader si occuperà di analizzare e progettare nuovi requisiti alternativi, che garantiscano funzionalità minime analoghe, ad esempio integrando API esterne secondarie.

5° - PE3.b Tempistiche per il testing

La mancanza di tempo per il testing porta a bug in produzione.

Elemento	Contromisura 1 - Test Driven Development (TDD)	Contromisura 2 - Testing Intensivo Pre-Consegna
Obiettivo	Ridurre l'accumulo di bug nelle fasi finali, distribuendo il testing lungo tutto lo sviluppo e non solo a partire dalla terza settimana dell'implementazione del modulo.	Individuare la maggior parte dei bug immediatamente prima della consegna.
Risultati e traguardi	Ogni funzionalità viene validata tramite Unit Test prima di essere considerata completa; I risultati sono una riduzione dei bug critici e minori correzioni d'emergenza. Il prodotto sono i test cases delle singole funzionalità.	Individuazione dei bug più visibili (crash, errori banali, ...). Non garantisce però copertura profonda né aumento della qualità strutturale del codice. Il prodotto

		sono i test cases delle singole funzionalità.
Responsabilità	Team di sviluppo, e revisione pull request da parte del Team Leader.	Tutto il team di sviluppo in modalità sincrona.
Approccio	L'attività di testing viene integrata nel processo di sviluppo, diventando parte strutturale della creazione del software. In questo modo, i difetti vengono intercettati nel momento esatto in cui emergono, evitando accumuli nelle fasi finali e riducendo la probabilità di imprevisti a ridosso della consegna.	Il testing esaustivo viene rimandato alla fase conclusiva del progetto con lo scopo di identificare le anomalie residue prima del rilascio. Questo metodo permette di individuare problemi evidenti e di usabilità, ma non elimina il rischio che difetti più profondi rimangano nascosti fino alla produzione.
Risorse	Richiede tempo aggiuntivo nelle prime iterazioni e competenze minime sui framework di test (es. JUnit). Non richiede presenza fisica simultanea.	Richiede tempo concentrato, disponibilità simultanea del team, dispositivi fisici e stress pre-consegna.
P	7 → 3 (poiché i bug vengono individuati pressoché subito evitando che il bug arrivi in produzione)	7 → 5 (si trovano solo alcuni bug: quelli più superficiali. Non c'è tempo per trovare quelli strutturali)
L	7 → 7 (rimane invariata perché se un bug passa comunque in produzione il danno è immutato)	7 → 7 (L'impatto è lo stesso. Come nella contromisura 1 (TDD), se un bug passa in produzione il danno rimane invariato)
Nuovo RE	3 x 7 = 21	5 x 7 = 35
Costo	5 (perché rallenta leggermente le fasi iniziali di scrittura del codice)	7 (alto sforzo + presenza sincrona + pressione temporale)
RRL	$(49 - 21) / 5 = 4$	$(49 - 35) / 7 = 2$

Scelta finale: la **contromisura 1** ha un RRL più alto; quindi, è la strategia più efficiente.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):**
Implementazione del TDD (Test Driven Development). Si stabilisce una modalità di sviluppo secondo la quale nessuna funzionalità può essere considerata completa se non accompagnata da un relativo Unit Test che ne verifichi il corretto funzionamento.
- **Contingency Strategy (Piano di Emergenza):**
Qualora il problema dovesse diventare reale il team dovrà sospendere i test per le componenti non critiche (es. animazioni UI, grafica, ...) per concentrare le risorse rimanenti esclusivamente su Integration Test per le funzioni più critiche (es. per il modulo "Prenotazioni": pagamento e conferma volo), accettando il rischio di bug estetici pur di garantire un core più stabile.

6° - PE3.c.1 Mancanza di design dettagliato

L'assenza di schemi chiari per il "Backend Bridge" rende lo sviluppo dei moduli voli, alloggi e pagamenti disorganizzato e caotico.

Elemento	Contromisura 1 - Definizione di uno Schema per le API	Contromisura 2 - Allineamento Continuo
Obiettivo	Eliminare ambiguità e disallineamenti sui dati condivisi tra Frontend e Backend definendo prima un contratto statico.	Risolvere i disallineamenti man mano che emergono tramite comunicazione costante e interazione diretta tra sviluppatori.
Risultati e traguardi	Il risultato è una struttura dati uniforme e riduzione dei casi di rework su oggetti comuni. Permette al Frontend di avanzare senza dipendere dallo sviluppo Backend. Il risultato è un file statico che definisce esattamente la struttura dei dati, evitando incongruenze tra i dati in arrivo delle API e i dati del Backend del progetto.	Vengono corretti i bug di integrazione.
Responsabilità	Collaborazione tra Sviluppatore Back-end e Sviluppatore Front-end con validazione finale del Team Lead per approvazione delle specifiche.	Sviluppatori Front-end e Back-end, ma richiede sincronizzazione e impegni simultanei.
Approccio	Prima di iniziare l'implementazione del Backend Bridge, si definisce un file statico (es. JSON Schema) che descriva esattamente gli endpoint e la struttura dei dati.	Si agisce durante l'integrazione, quando le modifiche sono più costose e impattano sulle tempistiche.
Risorse	Richiede tempo dedicato alla stesura di schema JSON, elenchi campi e codici errore. Nessuna necessità di presenza sincrona.	Richiede meeting lunghi, sessioni condivise, e blocca simultaneamente due risorse tecniche cruciali per molte ore
P	7 → 2 (poiché definire il "linguaggio" comune elimina le incomprensioni tra i moduli)	7 → 4 (riduce gli errori ma non previene i ripensamenti architetturali dell'ultimo minuto)
L	7 → 7 (rimane invariato poiché un errore di logica nel design avrebbe comunque lo stesso peso)	7 → 7
Nuovo RE	$2 \times 7 = 14$	$4 \times 7 = 28$
Costo	3 (richiede un impegno iniziale per la stesura del documento di specifiche)	7 (poiché blocca contemporaneamente due risorse chiave del team per molto tempo, richiedendo un forte sforzo e sottraendo ore allo sviluppo di altre feature)
RRL	$(49 - 14)/3 \approx 11,6$	$(49 - 28)/7 = 3$

Scelta finale: La **contromisura 1** è la più efficiente, con un valore della RRL decisamente più alto rispetto.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):** Definizione preventiva di uno schema obbligatorio per ogni endpoint Backend Bridge. Nessun endpoint del Backend Bridge può essere codificato senza una specifica che definisca tipi di dati, codici di errore e strutture JSON.
- **Contingency Strategy (Piano di Emergenza):** In caso di incongruenze nel JSON, si procederà all'identificazione immediata dei campi errati e alla loro sovrascrittura tramite una struttura dati locale corretta.

7° - PC1.a Consegne tardive

Il vincolo delle 3 settimane rende molto probabile una consegna incompleta del prodotto.

Elemento	Contromisura 1 - Taglio funzionalità	Contromisura 2 - Estensione orario di lavoro
Obiettivo	Garantire il rilascio del "core" del prodotto entro la scadenza prefissata	Completare tutte le feature previste nel backlog aumentando le ore di lavoro
Risultati e traguardi	Prodotto stabile con una consegna puntuale, pur con un set di funzionalità ridotto ma coerente col <i>MVP</i> (<i>Minimum Viable Product</i>)	Rilascio completo, ma con elevato rischio di instabilità e bug strutturali
Responsabilità	Product Owner (CEO/Professoressa che commissiona l'applicativo) e Team di sviluppo si riuniscono in un processo decisionale dove decidere quali feature non critiche bisogna rimuovere	Intero team di sviluppo
Approccio	Strategia basata su "prioritizzazione e sottrazione": si eliminano feature non essenziali per incentrarsi su risorse e componenti core ad alto valore.	Strategia basata sull'incremento dell'orario lavorativo, a costo di lavorare anche nel fine settimana. Si compensa la ridotta finestra temporale aumentando le ore produttive.
Risorse	Riunione di prioritizzazione e ridefinizione MVP	Ore straordinarie, turni serali e weekend, aumento stress e consumo di risorse mentali.
P	8 → 3 (avendo meno task da completare è molto più probabile rispettare la scadenza)	8 → 6 (lo stress e la stanchezza portano a nuovi bug, rallentando paradossalmente il lavoro)
L	6 → 4 (l'impatto diminuisce perché è meglio consegnare un prodotto con meno feature ma funzionante, piuttosto che uno completo ma rotto)	6 → 6
Nuovo RE	$3 \times 4 = 12$	$6 \times 6 = 36$
Costo	3 (richiede solo una riunione decisionale per ridefinire le priorità dello sviluppo)	9 (costo altissimo in termini di burnout del team e degrado della qualità del codice scritto sotto pressione)
RRL	$(48 - 12)/3 = 12$	$(48 - 36)/9 = 1.3$

Scelta finale: La **contromisura 1** è la più efficiente, risultando la scelta preferibile.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):** Riformulazione delle priorità: si classificano i task in "Must-Have" e "Nice-to-Have". Il monitoraggio del progresso avviene giornalmente per assicurarsi che i task vitali abbiano sempre la precedenza assoluta.
- **Contingency Strategy (Piano di Emergenza):** Se al termine della seconda settimana il trigger di ritardo è attivo, si procede al taglio formale delle funzionalità: tutte le funzionalità non essenziali vengono rimosse dal rilascio finale e le risorse vengono concentrate esclusivamente sulla stabilizzazione del core business di TripTales.

8° – PE5.d Inesperienza nella gestione di dati sicuri

L'inesperienza porta a rischio perché aumenta la probabilità di errori umani (come configurazioni sbagliate o esposizione involontaria dei dati).

Elemento	Contromisura 1: Formazione mirata e certificazione del team	Contromisura 2: Assunzione di figure esperte
Obiettivo	Acquisire competenze adeguate nella gestione di dati sensibili, transazioni finanziarie e protocolli HTTPS.	L'assunzione di personale esperto si fa carico dello sviluppo dell'implementazione e scelte progettuali critiche nell'ambito della gestione delle transazioni finanziarie e dei protocolli HTTPS.
Risultati e traguardi	Team formato e certificato, riduzione degli errori operativi, aggiornamento continuo sulle minacce.	Decisioni progettuali valide, riduzione di vulnerabilità critiche, controlli periodici. Il prodotto è un modulo di gestione dei pagamenti sicuro e affidabile.
Responsabilità	Team di sviluppo, responsabile della formazione, Team leader.	Consulenti esterni.
Approccio	Corsi certificati, formazione specifica, aggiornamenti periodici.	Impiego di consulenti esterni, che si occupano interamente della progettazione del modulo critico.
Risorse	Budget per corsi, tempo del team, materiale formativo.	I consulenti esterni diventano parte integrante del team per cui devono essere remunerati in base alla loro qualifica.
P	5 → 3	5 → 1
L	9 → 7 (dalla formazione potrebbero comunque persistere delle lacune)	9 → 4
Nuovo RE	21	4
Costo	6	8
RRL	4	5.1

Scelta finale: la **contromisura 2** ha un RRL superiore, risultando la scelta preferibile in termini di cost-effectiveness, garantendo anche un'elevata riduzione del rischio.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):** Si adotta la contromisura 2 assumendo personale esperto riducendo di molto la probabilità di avere personale inesperto.
- **Contingency Strategy (Piano di Emergenza):** Prevede il blocco immediato delle funzionalità compromesse, la notifica agli stakeholder e gli utenti coinvolti, il conseguente avvio delle procedure di risposta all'incidente. Seguiranno un'analisi per individuarne le cause e una revisione delle procedure, garantendo una gestione della crisi rapida, ordinata e conforme agli obblighi normativi.

9° – PE3.c.2 Saturazione delle risorse e Crash

La saturazione della RAM e di storage su dispositivi entry-level a causa di mappe 3D e immagini ad alta risoluzione, con crash dell'app e portano a rischio. L'utente potrebbe rimanere insoddisfatto della performance e disinstallare l'applicazione.

Elemento	Contromisura 1: Ottimizzazione risorse	Contromisura 2: Evitare blocchi da latenza API
Obiettivo	Ridurre il consumo di RAM e storage su dispositivi tramite compressione immagini, caching intelligente e pulizia periodica della cache	Evitare blocchi e crash dovuti a latenza API, garantendo che l'app rimanga reattiva anche quando i server esterni tardano a rispondere.
Risultati e traguardi	Implementazione di compressione immagini, cache funzionante, gestione dati persistenti senza perdita di informazioni	Tutte le chiamate API sono gestite in modo asincrono con retry (meccanismo che riprova a fare la chiamata all'API) e fallback (utilizzo dei dati memorizzati in cache locale). Questo approccio permette di ottenere la riduzione dei crash.
Responsabilità	Team di sviluppo	Team di sviluppo e sviluppatore backend (per gestione API)
Approccio	Analizzare le risorse consumate, adattare dinamicamente caricamento immagini e mappe, implementare cache con limiti e pulizia automatica	Rifattorizzare codice introducendo l'esecuzione di chiamate per thread separati in modo tale da evitare il blocco dell'applicazione. Ogni richiesta viene eseguita su un thread separato, consentendo all'interfaccia di rimanere sempre attiva. È introdotta una logica di timeout per determinare se la richiesta fallisce. In caso di timeout, il sistema prevede un meccanismo di retry che riesegua la chiamata.
Risorse	Tempo sviluppatori, eventuali librerie di compressione e caching	compressione e caching Tempo sviluppatori, monitoraggio memoria
P	5 → 3 (diminuisce perché le azioni preventive riducono la possibilità del crash)	5 → 3
L	9 → 6	9 → 6 (diminuisce perché le contromisure limitano i danni e mantengono l'app reattiva.)
Nuovo RE	18	18
Costo	6	6

RRL	4.5	4.5
-----	-----	-----

Scelta finale: le contromisure sono equivalenti dal punto di vista della riduzione del livello del rischio. Sono infatti complementari: prese insieme danno un'adeguata prospettiva di azione per ridurre il rischio.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):** come misure preventive è giusto utilizzare sia l'ottimizzazione delle risorse sia la gestione manuale delle chiamate API, perché entrambe vanno a ridurre la probabilità di crash dell'applicazione (anche per dispositivi con risorse limitate) e i ritardi nelle risposte.
- **Contingency Strategy (Piano di Emergenza):** In caso di crash ripetuti o blocchi dell'app, viene temporaneamente disabilitato il caricamento delle mappe 3D e dei contenuti ad alta risoluzione. Le API più lente vengono sospese o sostituite. L'utente viene informato che, per garantire stabilità e prestazioni, alcune funzionalità avanzate sono temporaneamente limitate.

10° – PE1.g Complessità del Prodotto

Aumenta esponenzialmente i punti di fallimento e riduce la visibilità complessiva, rendendo difficile identificare vulnerabilità, prevedere comportamenti anomali e garantire una sicurezza coerente in tutte le componenti interconnesse.

Elemento	Contromisura 1: Sviluppo incrementale (MVP e iterazioni)	Contromisura 2: Documentazione tecnica e linee guida di sviluppo
Obiettivo	Realizzare il prodotto per passi successivi partendo da una versione minima funzionante (MVP), estendendolo gradualmente per limitare la complessità introdotta.	Definire regole chiare e documentazione condivisa per rendere il codice più comprensibile e manutenibile, facilitando il lavoro su un sistema complesso.
Risultati e traguardi	Prima versione con funzionalità di base; nuove funzionalità aggiunte gradualmente; ogni iterazione validata prima di procedere. Versioni incrementali del progetto.	Documentazione dei moduli e delle loro responsabilità; linee guida per naming, struttura del codice ed eventuali pattern architetturali.
Responsabilità	Team di sviluppo, team leader	Team di sviluppo, team leader
Approccio	Approccio iterativo e incrementale basato su MVP, con pianificazione delle release e validazione continua.	Creazione e manutenzione di documentazione tecnica aggiornata e adozione di linee guida di sviluppo vincolanti.
Risorse	Team di sviluppo, ambienti di test, strumenti di pianificazione (es. Jira)	Tempo del team per documentare, strumenti di documentazione, sessioni di allineamento.
P	6 → 3	6 → 4
L	7 → 4	7 → 5
Nuovo RE	12	20
Costo	4	4

RRL	7.5	5.5
-----	-----	-----

Scelta finale: la **contromisura 1** presenta il valore di RRL più alto, essa risulta essere la soluzione più efficace in rapporto al rischio ridotto per unità di costo.

Piano di Gestione del Rischio

- **Management Strategy (Prevenzione):** Sviluppo incrementale del prodotto a partire da un MVP con funzionalità di base. Le nuove funzionalità vengono aggiunte gradualmente e validate ad ogni iterazione prima del rilascio successivo.
- **Contingency Strategy (Piano di Emergenza):** Il piano di contingenza viene attivato quando vengono rilevati moduli eccessivamente complessi o code smell critici che generano bug ricorrenti e problemi di integrazione. Prevede l'eliminazione dei componenti inutilizzati o ridondanti e l'analisi approfondita delle classi più complesse. Infine, viene eseguito un refactoring mirato delle parti critiche per ridurre accoppiamento e complessità, migliorando la manutenibilità del prodotto.