

Cliccasa

*La tua casa con un clic*

*Project plan*

Kenna Miriam Fatima *Matr. 1058218*

Longo Dandis Iana *Matr. 1065350*

Lorenzi Luca *Matr. 1068520*

*Progetto del corso Ingegneria del Software*

*Docente: Gargantini Angelo Michele*



Sommario

[**INTRODUZIONE** 5](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234311)

[**1.1 Lo sfondo e la storia del progetto5**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234312)

[**1.2 Obiettivi5**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234313)

[**1.3 Sintesi del progetto5**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234314)

[**2. MODELLO DI PROCESSO**6](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234315)

[**2.1 Definizione dei requisiti6**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234316)

[**3. Organizzazione del progetto:**7](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234317)

[**3.1 Struttura7**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234318)

[**3.2 Stile:8**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234319)

[**3.3 Organizzazione della squadra:8**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234320)

[**3.4 Ruoli 8**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234321)

[**Architetto: 8**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234322)

[**Programmatore: 8**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234323)

[**Testatore 9**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234324)

[**Project Manager 9**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234325)

[**4. Standard linee guida procedure**9](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234326)

[**4.1 Software e hardware9**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234327)

[**4.2 Incontri tra il personale9**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234328)

[**4.3 Documentazione9**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234329)

[**4.4 Qualità**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234330)

[**5. Attività di gestione**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234331)

[**6. Rischi**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234332)

[**6.1 Cosa sono i rischi? 10**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234333)

[**6.2 Rischi e come affrontarli**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234334)

**7.Personale**

[**8. METODI E TECNICHE**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234335)

**9. Garanzia di qualità**

[**10. PACCHETTI DI LAVORO**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234336)

[**11. Risorse**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234337)

[**11.1 Indicazioni generiche**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234338)

[**11.2 Hardware**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234339)

[**11.3 Software**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234340)

[**11.4 Personale**](file:///C:\Users\iana.dandis\Downloads\Project_plan%20(3).docx#_Toc90234341)

**12. Budget e programma**

**13. I cambiamenti**

**14. La consegna**



1

Introduzione

* 1. Lo sfondo e la storia del progetto

Il seguente progetto è stato realizzato da Miriam Kenna, Iana Longo Dandis e Luca Lorenzi come progetto universitario all’interno dell’Università degli Studi di Bergamo per comprendere al meglio i concetti teorici, nonché applicare le tecniche ed i metodi pratici spiegati durante il corso “Ingegneria del Software” tenuto dal prof. Angelo Michele Gargantini.

* 1. Obbiettivi
* la realizzazione di un software per la gestione interna (amministrazione) di una piattaforma online che permette la pubblicazione e la ricerca di annunci immobiliari;
* la realizzazione di un software di gestione della vendita degli immobili, facilmente utilizzabile da tutti, privati e agenzie, per produrre modifiche necessarie relativi agli annunci pubblicati e per la gestione di altri aspetti organizzativi che verranno ampiamente descritti successivamente;
* la realizzazione di un software di gestione dei pagamenti da parte dei privati e agenzie per la pubblicazione degli annunci sul portale.

1.3 Sintesi del progetto

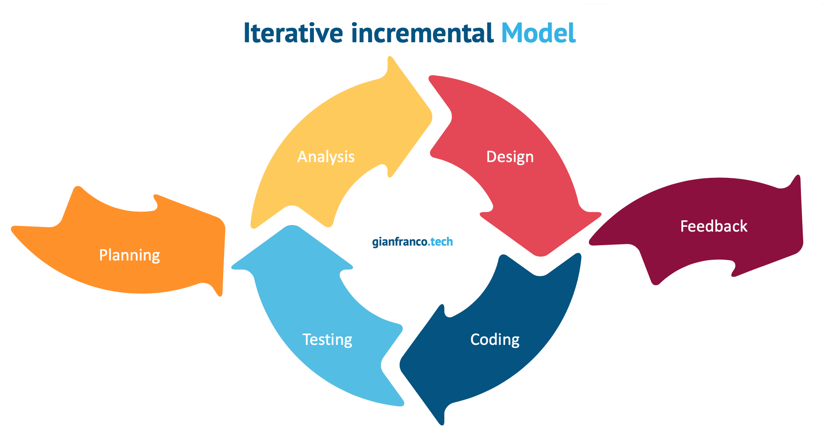
Il software ha come obiettivo finale quello di fornire ai clienti tutti i servizi necessari per facilitare il processo di compravendita di immobili attraverso la piattaforma online, in particolare i servizi possono essere forniti a diverse categorie di clienti: privati/ agenzie intenzionati a vendere immobili e privati interessati all’acquisto. In base alla categoria i clienti potranno accedere ed usufruire alle specifiche funzionalità del software, che verranno riprese e descritte ampiamente successivamente, inoltre il progetto software fornirà una parte di funzionalità destinati all’amministrazione e la gestione del portale nonché delle informazioni relative ai clienti e agli annunci per la vendita degli immobili.



2

Modello di processo

Per lo sviluppo di questo progetto software abbiamo deciso di utilizzare il modello incrementale (o modello iterativo), il quale si basa sulla successione dei seguenti step: pianificazione, analisi dei requisiti, progetto, implementazione, test e valutazione.



Ciò che ci ha portato a questa scelta è stata l’esigenza di ottenere uno “stadio” operativo del software ad ogni iterazione (prassi denominata Continuous Integration), fino alla soddisfazione dei requisiti richiesti dal cliente. Questo poiché durante la fase di analisi relative alla strutturazione del sistema abbiamo potuto distinguere delle macro parti: quella principale per la pubblicazione degli annunci, sulla quale si baserà poi la parte responsabile della modifica degli annunci stessi, alla quale poi sarà necessario aggiungere la parte relativa al pagamento. Ad ognuna di queste macro abbiamo potuto assegnare una priorità (di natura funzionale o architetturale), portando così, naturalmente, alla scelta del modello incrementale.

Pubblicazione annunci 🡪 priorità funzionale alta

Modifica annunci 🡪 priorità funzionale media

Possibilità di pagamento 🡪 priorità architetturale media

Essendo anche il primo progetto di sviluppo software sostenuto da noi in team, ci sembrava fondamentale l’integrazione dell’utente per la pianificazione del passaggio successivo, caratteristica fondamentale di questa tipologia di modello.



2.1 Definizione dei requisiti

Requisito 1: Caricamento degli annunci

Il software deve fornire agli utenti la possibilità di caricare i propri annunci sulla piattaforma online, rispettando le regole associate alle diverse tipologie di utente previste: privato o agenzia. In caso di privato, gli verrà data la possibilità di caricare il primo annuncio gratuitamente, per i successivi la tariffa sarà di 4 euro/mese, mentre per le agenzie sarà necessario sottoscrivere un abbonamento, che comporta la firma di un contratto online.

Requisito 2: Registrazione degli utenti sul portale

Questa distinzione tra la tipologia di utenti comporta anche la necessità di avere un diverso form di iscrizione a seconda della catalogazione.

I privati al momento dell’iscrizione dovranno inserire nome, cognome, indirizzo e-mail, password e numero di telefono; mentre le agenzie dovranno completare i campi: denominazione sociale dell’agenzia, partita iva, pec mail, referente dell’agenzia, telefono ed e-mail.

Requisito 3: Gestione dei pagamenti

Ipotizzando che successivamente al lancio, la piattaforma prenda piede, sopraggiunge la necessità di poter pubblicare più di un annuncio. Ciò comporta anche il dover sostenere maggiori costi di implementazione, gestione e manutenzione.

Da qui la decisione di inserire l’obbligatorietà di sottoscrivere un abbonamento per le agenzie, ed una tariffa mensile ad annuncio per i privati; con conseguente necessità di gestire i pagamenti.

Il pagamento avverrà tramite l’utilizzo di Stripe: esso propone infatti la propria infrastruttura tecnologica ed i propri software per accettare pagamenti e gestire la loro attività online. Sfruttando le sue potenzialità saremo in grado di gestire il pagamento degli abbonamenti e le fatture.

Inoltre è possibile sia per gli utenti privati che per le agenzie effettuare un upgrade dell’annuncio. Questo comporta una maggiore visibilità sul portale, portando l’annuncio in prima pagina della ricerca specifica degli utenti che ricercano un immobile. I privati possono effettuare upgrade di un solo annuncio pagando attraverso Stripe la cifra di 2,99 € al mese, mentre per le agenzie è previsto un abbonamento con la possibilità di effettuare l’upgrade di 5 annunci al prezzo di 6,99 € al mese.



Per poter identificare e accertare il raggiungimento dei requisiti, sia funzionali che qualitativi, del software viene utilizzata la fase di testing in concomitanza alla presentazione del software al cliente, così da poter identificare anche ulteriori esigenze del cliente e requisiti da soddisfare.

I test che abbiamo effettuato mirano a valutare l’effettivo funzionamento ed efficienza del nostro software. Per la gestione dei pagamenti viene utilizzata l’area di test messa a disposizione da Stripe stesso. Per quanto riguarda invece la macro di iscrizione viene testato il corretto inserimento della mail, un controllo sulla password con il doppio inserimento e un controllo del rispetto della password policy. Fondamentale è anche il controllo del corretto inserimento degli annunci, la loro visione e modifica. Per fare tutti questi controlli di varia natura, avremo due database; più avanti verrà esposta la motivazione di ciò.



3

Organizzazione del progetto

*Aspetti fondamentali della configurazione organizzativa interna al team*

3.1 **Struttura**: *Adattamento reciproco o adhocrazia*

* Il meccanismo di coordinamento è non gerarchico e si basa su un aggiustamento reciproco;
* Ciascun membro del team avrà le proprie attività da portare a termine, che verranno stabiliti in base alle priorità fissate;
* Si lavora in parallelo però con il focus su diversi aspetti dello stesso obiettivo e una volta raggiunto si procede con il prossimo;
* I componenti del team sono sempre aggiornati sull’avanzamento del lavoro degli altri ;
* La comunicazione è di tipo immediato, interpersonale è informale.

**3.2 Stile:** *Di relazione*

* Le attività verranno vincolate ai singoli componenti del gruppo, tuttavia messe in discussione e approvate dall’intero gruppo;
* Le attività considerate più complesse possono essere svolte in coppia;
* Il lavoro è specializzato e ben definito, ma la collaborazione è fondamentale;
* Gli aspetti decisionali riguardano l’intero gruppo.

**3.3 Organizzazione della squadra**:*Agile*

* Il lavoro è coordinato da una pianificazione, tuttavia il team deve essere pronto a rispondere ai cambiamenti, si cerca sempre di migliorare è ottimizzare;
* Il lavoro è fortemente guidato dall’esigenza dei clienti che utilizzeranno il software;
* I pacchetti di lavoro sono sviluppati e poi rilasciati appena pronti anche come prototipi imperfetti ancora da testare e migliorare;
* Ciascun componente del gruppo gestisce ed organizza il proprio lavoro, tuttavia, sono prestabiliti delle revisioni e consegne periodiche.



**3.4 Ruoli**

* Dato che il meccanismo di coordinamento non è gerarchico i ruoli risultano essere interscambiabili. Per l’interscambio dei ruoli l’unico vincolo da rispettare è:
* il testing di un pacchetto di lavoro non può essere svolto dalla stessa persona che lo ha programmato ma necessariamente da una componente del team diversa;
* ciascun membro del team è sempre aggiornato sulle attività degli altri sia a livello organizzativo che concettuale
* ogni membro si concentra sulle proprie attività da svolgere ma ha anche le competenze e le conoscenze necessarie per intervenire su ogni pacchetto di lavoro in sviluppo
* inoltre si tiene un registro in cui ogni membro del team segna le attività e le descrizioni delle

In particolare, sono delineati i seguenti ruoli necessari allo sviluppo del software con le seguenti responsabilità:

**Architetto**:

* definisce le specifiche che i programmi devono avere
* realizza il disegno logico dei singoli componenti del software

**Programmatore**:

* definisce il linguaggio di programmazione
* scrive il codice sorgente del software

**Testatore:**

* + coordina tutte le attività di testing (pianificazione, progettazione e preparazione, esecuzione, monitoraggio e reporting) all’interno del progetto
  + Fornisce il feedback necessario al miglioramento del processo di testing

**Project Manager:**

* + Stabilisce le procedure all’interno del progetto
  + Acquisisce i requisiti del progetto
  + Definisce gli obiettivi del progetto
  + Sviluppa i piani di progetto
  + Controlla, monitora e realizza la documentazione di progetto



4

Standard, linee guida e procedure

*Ci siamo prefissati delle convenzioni per lo svolgimento di questo progetto.*

*Queste sono state redatte sulla base dei diversi componenti del progetto che spaziano dall’organico ai documenti di progetto.*

4.1 **Software e hardware**

Abbiamo deciso di usare software e hardware standard per ogni membro del gruppo.

In questa maniera preveniamo eventuali incompatibilità tra programmi nel caso in cui due o più membri del gruppo debbano lavorare sullo stesso file.

4.2 **Incontri tra il personale**

Per svolgere al meglio il lavoro è necessario che i membri del progetto si incontrino a cadenza fissa.

Pertanto, abbiamo deciso di programmare incontri a cadenza settimanale divisi in due tranche da 2 ore ciascuna.

Durante gli incontri si dovranno aggiornare gli altri membri del gruppo dei propri progressi ed eventualmente discutere di particolari problemi che potrebbero sorgere durante il proseguire del lavoro.

4.3 **Documentazione**

Un paragrafo a parte merita la documentazione.

Questa, infatti include ogni tipo di documento appartenente al progetto, pertanto è necessaria particolare cura per definire le linee guida sul suo utilizzo.

I programmi necessari per costruirla sono definita nella tabella sopra.

Per la condivisione con i membri del gruppo è essenziale GitHub.

Grazie ad esso è possibile condividere e aggiornare in tempo reale i membri del gruppo delle nostre modifiche sulla documentazione oltre che renderla immediatamente accessibile a tutti.

4.4 **Qualità**

La qualità del nostro lavoro deve essere garantita attraverso fasi di testing ed eventuali revisioni degli elaborati.

Simulare una situazione reale è lo standard a cui fare riferimento durante lo svolgersi dei test del programma.

Anche i vari documenti dovranno essere di qualità.

Infatti è necessario che siano facilmente comprensibili a tutti i membri del gruppo senza l’utilizzo di forme gergali o dialettali ma utilizzando esclusivamente la lingua italiana.



5

Attività di gestione

• l’implementazione del software viene suddivisa in pacchetti di lavoro descritti nel punto 10

• i pacchetti di lavoro sono rigorosamente definiti in base alle priorità stabilite al punto 2

• le attività da svolgere sono assegnate ai membri del team come loro diretta responsabilità da portare a termine

• per ciascuna attività è inoltre stabilità una scadenza, tuttavia ancora prima della scadenza utilizzando i metodi di condivisione descritti nel punto 4 i membri del team hanno sempre la possibilità di essere aggiornati sullo stato e sull'avanzamento del progetto

• La gestione della singola attività è lasciata alla persona che ne ha la diretta responsabilità ma il risultato finale deve essere approvato dall’intero team



6

Rischi

6.1 **Cosa sono i rischi?**

I rischi del progetto sono una cosa concreta e che non possono essere eliminati del tutto.

Lo scopo della nostra gestione dei rischi è quella, infatti, di rendere altamente improbabile che questi rischi si realizzino.

6.2 **Rischi e come affrontarli**

Prendendo la lista elaborata da Boehm nel 1989 possiamo applicare una serie di contromisure:

|  |  |
| --- | --- |
| RISCHIO | COME AFFRONTARLO |
| Carenza di personale | Il personale selezionato deve coprire interamente il fabbisogno di organico.  Nel caso in cui per cause di forza maggiore venissero a mancare membri del personale la mansione, se critica per il proseguire del lavoro, verrà affidata al membro più qualificato. |
| Calendario/budget non realistici | La selezione dei tempi e del budget è fondamentale per evitare scontento sia da parte del cliente sia da parte del personale.  Pertanto verranno dati tempi e budget conformi alle necessità del lavoro. |
| Funzionalità sbagliata | Se le funzioni del sistema non dovessero corrispondere alle aspettative si dovrà rielaborare il tutto risolvendo il problema.  Poichè un situazione del genere potrebbe avere effetti decisamente negativi per il lavoro è assolutamente necessario prestare la massima attenzione alla definizione degli obbiettivi dello scopo del sistema collaborando attivamente in caso di dubbi sia tra i membri del personale sia comunicando col cliente. |



|  |  |
| --- | --- |
| Interfaccia utente sbagliata | L’interfaccia dovrà rispettare le richieste del cliente.  Un’interfaccia contorta o complessa rappresenta un rischio per il lavoro poiché non va incontro alle specifiche.  Anche in questo caso, come per la “Funzionalità sbagliata” la prevenzione è la cosa migliore. |
|  |  |
| Volatilità requisiti | Il cliente potrà modificare le sue richieste unicamente se le modifiche, dopo essere valutate dallo staff , saranno giudicate realizzabili rispettando tempi e budget. |
| Componenti esterni difettosi | Ogni componente difettoso riguardante materiale fisico dovrà essere riconsegnato al produttore in modo da ricevere altro hardware funzionante per proseguire col lavoro, nei casi ove ciò non sia possibile per fattori di tempo o simili l’hardware difettoso verrà sostituito con un omologo di un altro produttore.  Per quanto riguarda il software difettoso dovrà essere segnalato all’azienda fornitrice per trovare una soluzione rapida, in caso di esito negativo della segnalazione verrà sostituito con software analogo di altri enti. |
| Cattive attività esterne | Lacune nel materiale o nelle conoscenze fornite da terzi dovranno essere segnalate.  Si provvederà ad un nuovo addestramento del personale da una nuova azienda. |



|  |  |
| --- | --- |
| Carenze in tempo reale | Si procederà ad ulteriori fasi di test del sistema per individuare il punto debole.  Una volta trovato dovrà essere risolto prestando attenzione all’integrità complessiva del sistema. |
| Carenze di capacità | Prima di poter lavorare con nuove tecnologie o in ambiente diverso dal solito, per salvaguardare l’incolumità del personale, quest’ultimo dovrà essere addestrato al loro utilizzo/ dovrà sottoporsi ad un periodo di acclimatamento.  Questo periodo dovrà essere il più breve possibile per limitare i ritardi sulla tabella di marcia.  *ATTENZIONE!*  *La sicurezza del personale è al primo posto, assolutamente non fornire nuovi strumenti senza che il personale sia stato addestrato al loro utilizzo* |



7

Personale

Il progetto di “Cliccasa” è partito il 5 Dicembre 2021 con la prima call di organizzazione, avendo come step intermedio una prima consegna del project plan entro il 27 Dicembre 2021.

Il tempo a disposizione risulta essere di circa 50 giorni, con scadenza entro il 27 Gennaio 2022. Esso verrà suddiviso con una media di circa 10 ore settimanali, con aggiornamenti dati in una call di allineamento a cadenza settimanale.

Il tempo a disposizione è stato così suddiviso:

Il personale a disposizione per questo progetto risulta essere composto da un team di tre persone: Kenna Miriam Fatima, Lorenzi Luca e Longo Dandis Iana. I tre collaboratori risultano avere delle competenze omogenee dovute alla frequenza del medesimo corso di Ingegneria Informatica e competenze acquisite grazie alla mansione personale svolta di Identity and Access Manager e di Full Stack Developer.

Le competenze complessive riguardano: linguaggio C#, C, C++, Java, JavaScript, HTML 5 e CSS, PHP, SQL ed utilizzo dei database relazionali, gestione degli accessi e delle identità.



8

Metodi e tecniche

*I metodi e le tecniche utilizzate sono diverse per ogni fase affrontata: l'ingegneria dei requisiti, la progettazione, l'implementazione e il test.*

Durante l’ingegneria dei requisiti abbiamo scomposto la nostra idea di software in 3 macro categorie fondamentali al funzionamento dello stesso: pubblicazione degli annunci, modifica degli annunci e gestione dei pagamenti.

Abbiamo poi associato ad ognuno di esse una priorità, funzionale o architetturale. La pubblicazione degli annunci possiede una priorità in primo luogo funzionale poiché risulta essere il cuore del nostro software e di conseguenza avrà una priorità alta.

La modifica degli annunci risulta poi essere necessaria in quanto trattandosi di immobili il prezzo dopo un periodo di tempo in vetrina può necessitare di una valutazione differente, in concordanza anche con l’andamento del mercato; inoltre è logico supporre uno scorretto inserimento di dati tecnici o l’esigenza di modificare le foto dell’immobile precedentemente pubblicate. Risulta così essere una priorità funzionale media.

La gestione dei pagamenti è una parte necessaria al funzionamento del nostro software a causa della metodologia di funzionamento con cui è stato pensato: si suppone infatti di sottoscrivere un abbonamento con rinnovo mensile o annuale. Necessiteremo quindi di utilizzare le funzionalità di una piattaforma di pagamento esterna, in quanto realizzare l’infrastruttura di pagamento aumenterebbe i costi di sviluppo, e soprattutto nel tempo. È stato quindi scelto l’utilizzo di Stripe. Questa macro risulta avere quindi una priorità architetturale, questa volta, media.



Per le categorie di pubblicazione e modifica è stato deciso di produrre una guida utente, con la spiegazione passo a passo per poter effettuare queste azioni, e una guida amministratore per poter risolvere eventuali problemi ed effettuare la verifica e convalida dell’annuncio e/o delle modifiche.

Per la gestione dei pagamenti è stata prodotta una guida amministrativa, contenente le spiegazioni tecniche e contabili per la gestione degli stessi.

A scopo di testing è stata presa la decisione di avere due database: un database di test ed un database di produzione. Sul database di test verranno implementate e testate tutte le funzioni del software; una volta accertata la loro corretta funzionalità, verranno portate in produzione e quindi aperta una connessione del software con il database di produzione.

Il database di test viene utilizzato anche per verificare il corretto salvataggio degli utenti e degli annunci all’interno di esso, senza perdita di informazioni essenziali o divulgazione di informazioni sensibili.

Per la gestione dei pagamenti viene utilizzato Stripe, il quale mette a disposizione delle API da integrare.

**Librerie API**

Stripe possiede librerie ufficiali per i diversi linguaggi di programmazione e le diverse piattaforme per dispositivi mobile, ed inoltre sono disponibili anche numerosi plugin e librerie di terze parti creati dalla community di Stripe.

*Nel link la documentazione completa:* [*https://stripe.com/docs/libraries#librerie-lato-client-e-interfaccia-utente*](https://stripe.com/docs/libraries#librerie-lato-client-e-interfaccia-utente)

*Oppure la documentazione su GitHub:* [*https://github.com/stripe/stripe-ruby*](https://github.com/stripe/stripe-ruby)

## **Librerie lato client e interfaccia utente**

Nel nostro caso utilizziamo Stripe sul Web, quindi saranno utilizzati gli Elements: un insieme di componenti preintegrati dell’interfaccia utente per raccogliere e convalidare i dati delle carte. Per creare i flussi di pagamento Elements utilizza Stripe.js, la libreria JavaScript necessaria.

*(*[*https://stripe.com/it-es/payments/elements*](https://stripe.com/it-es/payments/elements)*)*

Inoltre, Stripe offre un’area personale che permette la gestione dei pagamenti, quindi si ha la possibilità di visualizzare una dashboard con il riepilogo e accettare o rifiutare un pagamento.

*La configurazione di essa in Java è descritta qui:* [*https://stripe.com/docs/payments/accept-a-payment?platform=web&ui=checkout#testing-apple-pay-google-pay*](https://stripe.com/docs/payments/accept-a-payment?platform=web&ui=checkout#testing-apple-pay-google-pay)



Dopo la configurazione, Stripe offre un’area di testing, nella quale risulta sufficiente seguire le seguente istruzioni:

* Fare clic sul pulsante di pagamento;
* Compilare i dettagli di pagamento con i dati della carta di test:
  + Inserire 4242 4242 4242 4242 come numero della carta;
  + Inserire una data futura qualsiasi come scadenza della carta;
  + Inserire un numero di 3 cifre per il CVC;
  + Inserire un qualsiasi codice postale.
* Fare clic su Paga;
* Si verrà reindirizzati alla nuova pagina di conferma;
* Individuare il nuovo pagamento nella Dashboard Stripe. I pagamenti riusciti sono visualizzati nell’[elenco dei pagamenti](https://dashboard.stripe.com/payments) della Dashboard. Se si fa clic su un pagamento, si viene reindirizzato alla pagina dei dettagli del pagamento. La sezione “Riepilogo Checkout” contiene le informazioni di addebito e l’elenco degli articoli acquistati, che si può utilizzare per evadere l’ordine manualmente.

Stripe mette a disposizione diverse carte di test che puoi utilizzare per verificare che l’integrazione sia pronta per la produzione.

|  |  |
| --- | --- |
| **Numero carta** | **Descrizione** |
| 4000000000003220 | Affinché il pagamento abbia esito positivo, è necessario completare l’autenticazione 3D Secure 2. |
| 4242424242424242 | La transazione ha esito positivo e il pagamento viene elaborato immediatamente. |
| 4000000000009995 | La transazione ha sempre esito negativo con un codice di rifiuto insufficient\_funds. |

**Apple Pay e Google Pay**

Per abilitare Apple Pay o Google Pay in Stripe Checkout, non è necessario apportare modifiche alla configurazione o all’integrazione. Questi pagamenti vengono gestiti come tutti gli altri pagamenti con carta.



Per quanto riguarda invece l’iscrizione degli utenti vengono testati in primo luogo i campi inseriti e la loro correttezza:

• Corretto inserimento della mail: presenza di una singola chiocciola, assenza di caratteri speciali non conformi (es: /\*$%), esistenza della mail con conferma per l’attivazione del profilo.

• Password: viene richiesto un doppio inserimento della password per evitare errori di battitura, e rispetto della password policy (presenza di almeno una lettera maiuscola, di almeno una lettera minuscola, di almeno un numero e di almeno un carattere speciale).

Per gli annunci viene effettuato il medesimo controllo sia per la prima pubblicazione dell’annuncio che per le modifiche allo stesso. Si ha un controllo sull’avvenuto inserimento delle caratteristiche fondamentali, quali locazione, prezzo, numero di locali ed ampiezza dell’immobile. Ciò comporta anche il dover controllare il funzionamento dello strumento di ricerca degli annunci: particolare importanza è stata data alla presenza di caratteri speciali nella località, come ad esempio “Trezzo sull’Adda”, per verificare che essi non impedissero la corretta ricerca.

Un ultimo test fondamentale è il test di accettazione, ovvero il test sotto la supervisione dell’utente per poter individuare eventuali errori, incomprensioni o esigenze aggiuntive. Il cliente verrà convocato al termine di ogni step, gli verrà spiegato gli obbiettivi raggiunti ed il loro funzionamento e verranno successivamente testate insieme. Questa fase risulta essere fondamentale, proprio per la tipologia di modello di progettazione scelto, che include una forte partecipazione del cliente. Successivamente a questa fase, si continuerà con l’implementazione delle parti successive, o delle nuove necessità comunicate dal cliente.



9

Garanzia di qualità

Per garantire la qualità i piani di test sono progettati prima che venga sviluppato il codice software effettivo del progetto. In particolare, come descritto nelle attività di gestione, l’intero progetto è suddiviso in pacchetti di lavoro, i quali sono definiti in base alle priorità stabilite durante la definizione dei requisiti. Per ogni pacchetto di lavoro è quindi garantito un piano di test.

L’obiettivo finale del test è quello di determinare le condizioni che contrassegnano il successo o il fallimento del pacchetto di lavoro. Infatti, un pacchetto di lavoro è considerato concluso solo dopo aver svolto il testing del medesimo, con esito positivo.

Inoltre, come indicato nel punto 3, il testing di un pacchetto di lavoro non compete alla persona che ne è responsabile ma deve essere svolto necessariamente da una componente del team alternativa.

Il piano di test in questione riporta un elenco completo di procedure che devono essere testate per ciascun requisito individuato, con l’obbiettivo aggiuntivo di determinare quali difetti siano inaccettabili, al fine di garantire una maggiore qualità.

|  |  |  |
| --- | --- | --- |
| **Requisito** | **Procedure da testare** | **Difetto inaccettabile** |
| *Caricamento degli annunci* | * Inserimento dei dati: controllo dei campi obbligatori; * Caricamento delle foto: controllo dell’ordine di inserimento la prima foto principale, il resto foto secondarie; * Test del salvataggio degli annunci con i dati e le foto salvate correttamente. | * Salvataggio di un annuncio con i campi obbligatori non compilati; * Salvataggio di un annuncio senza la foto principale. |

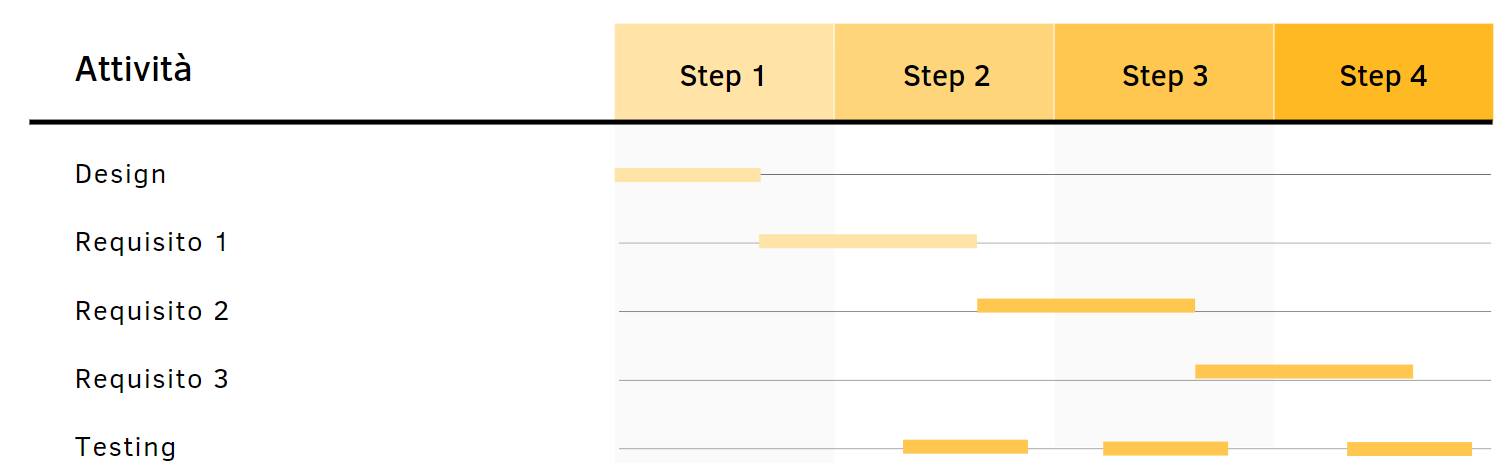


|  |  |  |
| --- | --- | --- |
| *Modifica dei dati personali* | Salvataggio corretto delle modifiche | Perdita delle modifiche |
| *Modifica dell’annuncio* | Salvataggio corretto delle modifiche | Perdita delle modifiche |
| *Upgrade dell’annuncio* | Testare che l’upgrade venga effettuato dopo l’avvenuto pagamento | Accettare un pagamento senza effettuare upgrade successivamente |
| *Registrazione dei privati/agenzie sul portale* | * Inserimento dei dati: controllo dei campi obbligatori; * Controllo della password: doppio inserimento e rispetto della password policy; * Corretto inserimento della mail, rispetto della mail policy; * Controllo se la mail inserita è già presente nel database. | * Perdita della registrazione dell’utente ; * Utente registrato due volte con la stessa mail. |
| *Pagamento* | Risorse di test offerte da Stripe e descritte ampiamente al punto 8. | |



10

Pacchetti di lavoro



• Ogni attività si identifica con un pacchetto di lavoro

• Lo svolgimento delle attività è sequenziale

• Le attività di testing vengono svolte alla fine dell’implementazione di ciascun pacchetto

• Se dopo aver eseguito il testing non si presentano problemi si passa alla prossima attività, altrimenti si risolve il problema e poi si procede

• Le attività come anche i tempi, gli step e le priorità sono solo indicativi e servono come punto di partenza per la realizzazione del software e quindi potrebbero essere modificati o definiti meglio successivamente

• I pacchetti di lavoro inoltre devono essere rivisti e modificati in seguito ad una richiesta di modifica, la gestione della quale è descritta al punto 13D



11

Risorse

## 11.1 **Indicazioni generiche**

Per evitare intoppi o rallentamenti durante lo svolgimento del lavoro i materiali usati devono essere di qualità e performanti.

Questo è riferito sia alla parte software sia alla parte hardware.

Le tabelle seguenti indicano i requisiti minimi della propria dotazione per poter lavorare al progetto limitando imprevisti.

## 11.2 **Hardware**

|  |  |  |
| --- | --- | --- |
| Hardware | | |
|  | **Minimo** | **Consigliato** |
| CPU | AMD FX 8320/ Intel I5 2nd | AMD Ryzen 5 3xxx/ Intel I7 10th |
| RAM | 4GB | 8GB |
| Scheda video | Intel HD graphics (6xx) | AMD R7 370/ Nvidia GTX 750 |
| Unità di memoria | SSD | SSD |

## 11.3 **Software**

|  |  |
| --- | --- |
| Software | |
| **Ambito** | **Consigliato** |
| Sistema operativo | Microsoft Windows (7 o superiore) |
| Stesura documenti di testo | Microsoft Word |
| Diagrammi | StarUML |
| Condivisione file | GitHub |

## 11.4 Personale

|  |  |  |
| --- | --- | --- |
| Personale | | |
| **Nome** | **Stadio progetto** | **Quantità** |
| Architetto | Iniziale |  |
| Programmatore | Centrale |  |
| Testatore | Finale |  |
| Project Manager | Iniziale |  |



11.4 Personale

|  |  |  |
| --- | --- | --- |
| Personale | | |
| **Nome** | **Stadio progetto** | **Quantità** |
| Architetto | Iniziale |  |
| Programmatore | Centrale |  |
| Testatore | Finale |  |
| Project Manager | Iniziale |  |



12

Budget e programma

È stato concordato all’interno del team che il budget per questo progetto sarà definito in termini di tempo, essendo questo un progetto realizzato in ambito universitario. Sarà così composto:

- Un’ ora a settimana per la riunione di allineamento, via teams, per aggiornamenti sull’andamento del lavoro;

- In aggiunta: il tempo che ciascun membro del team dedicherà allo svolgimento dei propri requisiti nonché allo studio/approfondimento delle conoscenze aggiuntive necessarie per la realizzazione del progetto [circa 10 h/settimana].

Per quanto riguarda il programma [come specificato al punto 10] durante la definizione dei pacchetti di lavoro, si è deciso suddividere la realizzazione del progetto in diversi step. La durata di ciascuno step viene adattata e decisa in base alle scadenze della consegna.

Una rappresentazione approssimativa delle attività è rappresentata nella tabella sottostante:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Attività | Durata | Vincoli |
| Project Plan | * Definizione del software design; * Definizione della software architecture; * Specifica dei requisiti; * Definizione del software life cycle; * Specifica di tutti gli aspetti organizzativi del lavoro, delle risorse e comunicazione interna del team. | Prima versione:  05/12/2021  -  27/12/2021  Per le modifiche successive si terrà traccia su GitHub. | Consegna entro un mese prima dell’esame |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Attività | Durata | Vincoli |
| Project Plan | * Definizione del software design; * Definizione della software architecture; * Specifica dei requisiti; * Definizione del software life cycle; * Specifica di tutti gli aspetti organizzativi del lavoro, delle risorse e comunicazione interna del team. | Prima versione:  05/12/2021  -  27/12/2021  Per le modifiche successive si terrà traccia su GitHub. | Consegna entro un mese prima dell’esame |



|  |  |  |  |
| --- | --- | --- | --- |
| Modelling | * Use case diagram; * Class Diagram; * State machine diagram; * Activity diagram; * Implementazione del relativo codice in JAVA. | - | Rispettare i punti forniti nel documento su UML. |
| Software  Testing | Testing a fine di ogni pacchetto di lavoro, rispettando il piano di test descritto nel capitolo 9 | - | Il testing di un pacchetto di lavoro non può essere svolto dalla persona che ne è responsabile ma necessariamente da una componente del team diversa. |
| Software  Maintenance | Attività di refactoring ed ottimizzazione del codice | - | Da completare 5 giorni prima dell’esame orale. |



13

Cambiamenti

Un cambiamento può essere effettuato perché richiesto dall’utente tramite l’assistenza o perché portato come idea da una componente del team. In ogni caso prima di procedere si valuta l’effettiva esigenza di eseguire la modifica.

Se la modifica non porta a miglioramenti evidenti ma nello stesso tempo si valuta che potrebbe portare rischi difficili da gestire per quanto riguarda il codice ed il prodotto in generale, allora la modifica viene respinta.

Se invece si riconosce la necessità di effettuare la modifica come prima cosa si valuta la priorità.

Se la priorità è alta allora si decide di fissare una riunione durante la quale si stabilisce a chi verrà assegnato come task l’effettuazione della modifica, in generale viene assegnata alla persona responsabile della gestione della parte che interessa la modifica. Inoltre, durante la riunione si stabiliscono i tempi indicativi, si valutano i rischi e le eventuali difficoltà.

Se invece si decide che la priorità è bassa allora la richiesta della modifica viene posta in coda e verrà ripresa solo quando le altre attività a priorità più alta verranno completate.

Passi da seguire per eseguire una modifica:

* Il primo step è salvare una coppia della parte che si ha intenzione di modificare
* Si effettua la modifica sulla coppia, quindi si crea una versione del software diversa
* Dopo aver eseguito la modifica si procede con l’attività di testing
* Se non si presentano problemi in fase di test la nuova versione del software va a sostituire quella precedente



14

Consegna

Il progetto deve essere consegnato mediante una repository in GitHub, la quale deve essere condivisa con il professore Gargantini Angelo e l’esercitatrice Bonfanti Silvia. Per eventuali comunicazioni è necessario utilizzare gli issue ed assegnarli al professore, e per eventuali modifiche è necessario aggiornare i documenti tramite apposite funzioni di GitHub.

Il project plan deve essere consegnato entro il 27 Dicembre 2021 ed il progetto completo deve essere disponibile entro i 5 giorni precedenti il 27 Gennaio 2022.



Heading 2

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore et feugiat nulla facilisis.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore et feugiat nulla facilisis.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore et feugiat nulla facilisis.

Sed ut perspiciatis, unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam eaque ipsa, quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt, explicabo. nemo enim ipsam voluptatem, quia voluptas sit, aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos, qui ratione voluptatem sequi nesciunt, neque porro quisquam est, qui dolorem ipsum, quia dolor sit, amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt, ut labore et dolore magnam aliquam quaerat voluptatem. ut enim ad minima veniam.

1

Heading 1

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore et feugiat nulla facilisis.

Sed ut perspiciatis, unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam eaque ipsa, quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt, explicabo. nemo enim ipsam voluptatem, quia voluptas sit, aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos, qui ratione voluptatem sequi nesciunt, neque porro quisquam est, qui dolorem ipsum, quia dolor sit, amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt, ut labore et dolore magnam aliquam quaerat voluptatem. ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? quis autem vel eum iure reprehenderit, qui in ea voluptate velit esse, quam nihil molestiae consequatur, vel illum, qui dolorem eum fugiat, quo voluptas nulla pariatur?