Gestione Apiario  
Documentazione

[1 Introduzione 3](#_Toc59120262)

[1.1 Informazioni sul progetto 3](#_Toc59120263)

[1.2 Abstract 3](#_Toc59120264)

[1.3 Scopo 3](#_Toc59120265)

[2 Analisi 4](#_Toc59120266)

[2.1 Analisi del dominio 4](#_Toc59120267)

[2.2 Analisi e specifica dei requisiti 4](#_Toc59120268)

[2.3 Use case 6](#_Toc59120269)

[2.4 Pianificazione 7](#_Toc59120270)

[2.4.1 Analisi: 8](#_Toc59120271)

[2.4.2 Progettazione 8](#_Toc59120272)

[2.4.3 Implementazione 8](#_Toc59120273)

[2.4.4 Test 9](#_Toc59120274)

[2.4.5 Documentazione 9](#_Toc59120275)

[2.5 Analisi dei mezzi 9](#_Toc59120276)

[2.5.1 Software 9](#_Toc59120277)

[2.5.2 Hardware 10](#_Toc59120278)

[3 Progettazione 11](#_Toc59120279)

[3.1 Design dell’architettura del sistema 11](#_Toc59120280)

[3.2 Design dei dati e database 12](#_Toc59120281)

[3.2.1 Descrizione delle tabelle del database 12](#_Toc59120282)

[3.2.2 Schema logico 13](#_Toc59120283)

[3.3 Design delle interfacce 14](#_Toc59120284)

[4 Implementazione 18](#_Toc59120285)

[4.1 Virtual Machine, preparazione Web Server 18](#_Toc59120286)

[4.2 Database 18](#_Toc59120287)

[4.3 Programmazione 18](#_Toc59120288)

[4.3.1 Implementazione sito 19](#_Toc59120289)

[5 Test 21](#_Toc59120290)

[5.1 Protocollo di test 21](#_Toc59120291)

[5.2 Risultati test 23](#_Toc59120292)

[5.3 Mancanze/limitazioni conosciute 23](#_Toc59120293)

[6 Gantt Consuntivo 24](#_Toc59120294)

[6.1 Differenze con il gantt preventivo 25](#_Toc59120295)

[7 Conclusioni 26](#_Toc59120296)

[7.1 Sviluppi futuri 26](#_Toc59120297)

[7.2 Considerazioni personali 26](#_Toc59120298)

[8 Bibliografia 27](#_Toc59120299)

[8.1 Sitografia 27](#_Toc59120300)

[9 Allegati 27](#_Toc59120301)

[10 Indice delle figure 27](#_Toc59120302)

# Introduzione

## Informazioni sul progetto

Allievi coinvolti nel progetto: Aris Previtali

Classe: informatica 3AC Scuola Arti e Mestieri Trevano

Docente responsabile: Geo Petrini

Data inizio:2020-09-03

Data consegna: 2020-12-17

## Abstract

Come apicoltore, tener conto di ogni apiario, note, meteo e trattamenti per essa può essere complicato e noioso. Quindi questo prodotto è ciò che fa al caso vostro, un gestore di arnie con tutte le funzionalità più comode. Si potranno gestire le proprie arnie aggiungendole, modificandole o eliminandole, si potranno aggiungere dei trattamenti, con data iniziale e finale con la notifica da parte del sistema all’utente una volta finito, delle note alle arnie che verranno visualizzate all’interno del calendario della pagina home utile anche per le informazioni meteo, del luogo in cui risiedono le arnie, che il sistema salva e mostra all’utente.

Gli elementi noiosi di gestire un apiario come salvare i trattamenti su di un foglio o le note saranno resi molto più facili e molto più ordinati rispetto a dei semplici fogli di carta, saranno infatti come detto primi visibili sul calendario nella pagina home dell’utente.

## Scopo

Lo scopo del progetto Gestione Apiario è quello di creare un sito nella quale è possibile creare un account per qualsiasi persona (apicoltore) che voglia gestire le proprie arnie, tra le varie opzioni troviamo la possibilità di aggiungere un’arnia, modificarla o eliminarla. Inoltre, è possibile aggiungere trattamenti con descrizione, data d’inizio e data di fine, note sia sulle arnie sia nei giorni del calendario, il quale sarà visibile nella home page del sito, sempre all’interno del calendario poter vedere i dati meteo del luogo di un’arnia della giornata e di quelle precedenti.  
Il sito è strutturato in modo da poter avere più utenti con accesso a tutte le possibilità che offre il sito.

# Analisi

## Analisi del dominio

Questo progetto serve per gestire un apiario. Potrà e verrà utilizzato da qualunque apicoltore voglia utilizzarlo. Questo progetto è web-based, ovvero verrà creata una pagina web dalla quale si può gestire il proprio apiario. Per questo motivo è infatti necessario, per me, lavorare con un Web Server con Apache, PHP e MariaDB (SQL) installati e andare a configurarli.

Il sito dev’essere appunto raggiungibile da qualsiasi browser.

## Analisi e specifica dei requisiti

Quella che segue è la specifica dei requisiti:

|  |  |
| --- | --- |
| ID: REQ-001 | |
| Nome | Creare sito web |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-002 | |
| Nome | Poter mettere e dover gestire numero di arnie |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-003 | |
| Nome | Poter mettere e dover gestire luogo di arnie |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-004 | |
| Nome | Poter annotare ogni arnia |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-005 | |
| Nome | Data trattamenti sanitari e quanti giorni di durata |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-006 | |
| Nome | Durata di una determinata attività e notifica ad attività finita |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-007 | |
| Nome | Attività visibili in calendario, sia passate che future |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-008 | |
| Nome | Poter aggiungere anno di nascita dell’ape regina |
| Priorità | 2 |
| Versione | 1.0 |
| Note |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ID: REQ-009 | |  |  |
| Nome | Multiutente (login) |  | Poter aggiungere anno di nascita dell’ape regina |
| Priorità | 1 |  | 2 |
| Versione | 1.0 |  | 1.0 |
| Note |  |  |  |

|  |  |
| --- | --- |
| ID: REQ-010 | |
| Nome | Informazioni meteo |
| Priorità | 1 |
| Versione | 1.0 |
| Note |  |

|  |  |
| --- | --- |
| ID: REQ-011 | |
| Nome | Formato data: YYYY-MM-DD |
| Priorità | 3 |
| Versione | 1.0 |
| Note | Per tutta la lunghezza del progetto il formato della data utilizzato sarà YYYY-MM-DD |

## Use case



Figura - Schema use case

L’utente visitatore del sito potrà accedere al sito loggandosi o, se dovesse essere la prima volta, registrandosi. Essi ovviamente includono il controllo della password, dell’email e del nome utente in caso di login. Per la gestione del calendario da parte dell’utente ci dovrà essere un formato per la data: per l’intero progetto ho scelto che questo formato debba essere YYYY-MM-DD.

L’utente potrà inoltre aggiungere delle note sulle arnie e nel calendario.

Il sistema dovrà notificare all’utente, con appunto delle mail, la fine dei trattamenti aggiunti precedentemente dall’utente.

## Pianificazione

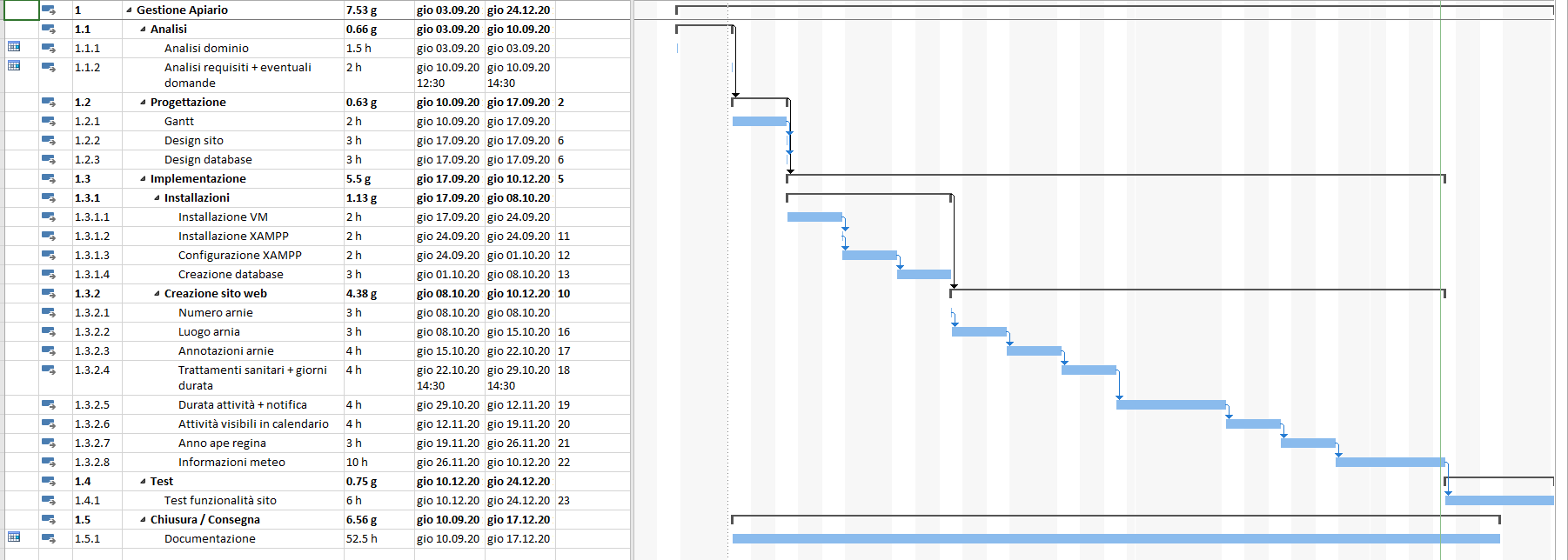


Figura - Diagramma di Gantt

## Analisi:

Ho suddiviso lo svolgimento in quattro grandi e principali attività, quest’attività consiste nel capire cosa mi venisse chiesto di realizzare e porre, se fosse il caso, alcune domande che mi venissero in mente.

Inoltre, in questa fase ho realizzato la tabella dei requisiti.

Oltre a capire in cosa consistesse il progetto ho guardato e mi sono documentato sui migliori strumenti da utilizzare di cui avessi bisogno.

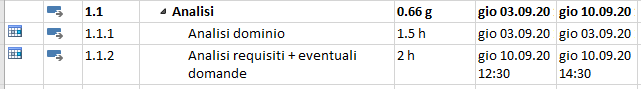


Figura - Analisi, Gantt

## Progettazione

Per la progettazione ho ideato il diagramma di Gantt sopra descritto inoltre ho progettato i due design che mi servivano per la realizzazione del sito, ovvero il design del sito stesso con la struttura principale delle pagine e il design del database il quale serve per salvare le arnie e tutto ciò che concerne un utente.

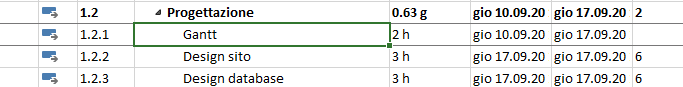


Figura - Progettazione, Gantt

## Implementazione

Nell’implementazione ho aggiunto tutte quelle attività che avevo bisogno di fare per creare il sito web, nell’implementazione c’è, inoltre, le installazioni, macchina virtuale, XAMPP, realizzazione del database e la preparazione della macchina sulla quale ho lavorato.

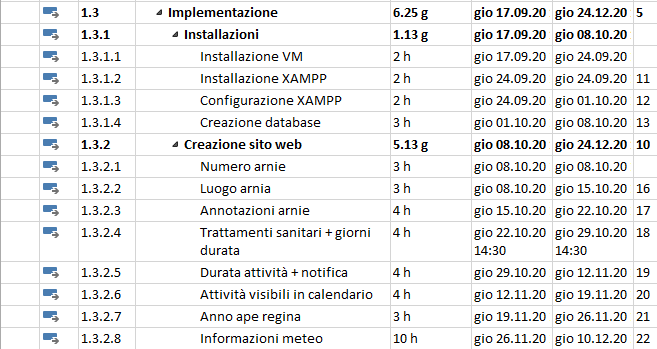


Figura - Implementazione, Gantt

## Test

Nell’ultima principale attività prima della chiusura ho aggiunto i test da fare al sito.



Figura - Test, Gantt

## Documentazione

Per la documentazione ho calcolato di scriverla per tutta la durata del progetto poco alla volta quando avevo dello spazio libero.

## Analisi dei mezzi

### Software

I software che ho utilizzato sono:

* Mozilla Firefox Version: 62.0
* PHP Version:7.4.9
* Apache Version: 2.4.46
* MariaDB Version:10.4.14
* Workbench Version:8.0
* Microsoft VS code Version: 1.49.1
* VMWare Workstation 14 Pro, Version: 14.1.2
* Microsoft Project 2016
* Microsoft Word 2016
* Draw.io (<https://draw.io>)
* Mockflow (<https://www.mockflow.com>)

### Hardware

Per l’hardware ho utilizzato una macchina virtuale avente come sistema operativo Windows 10.

# Progettazione

## Design dell’architettura del sistema

Un qualsiasi utente che voglia visitare il sito deve avere necessariamente accesso ad internet, è possibile accedere al sito tramite un qualsiasi dispositivo, che sia un computer, un tablet o iPad o un telefono.

Oltre al sito è ovviamente necessaria una registrazione con nome, e-mail e password.

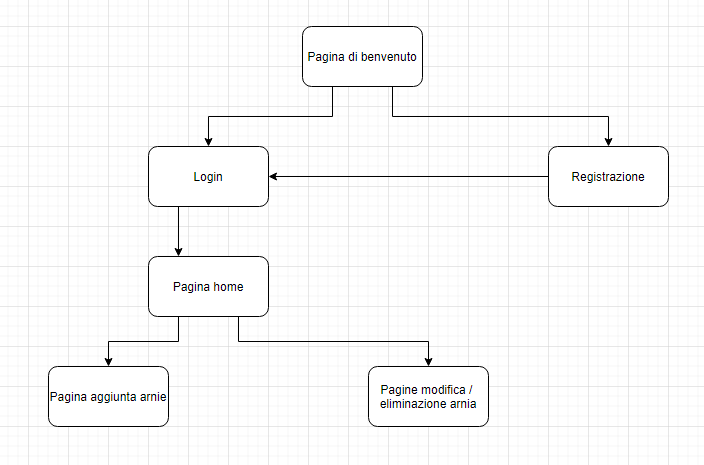
Sitemap:

Figura - Sitemap, Gestione Apiario

L’utente una volta loggato si ritroverà nella pagina home dove vedrà le proprie arnie, “il calendario con la meteo e le note”. Potrà, con dei bottoni, portarsi nella pagina di aggiunta arnie dove appunto potrà aggiungere un’arnia o alla pagina di modifica / eliminazione di un’arnia dove potrà modificare un’arnia, aggiungere una nota o un trattamento ad una di esse o eliminarne una.

## Design dei dati e database

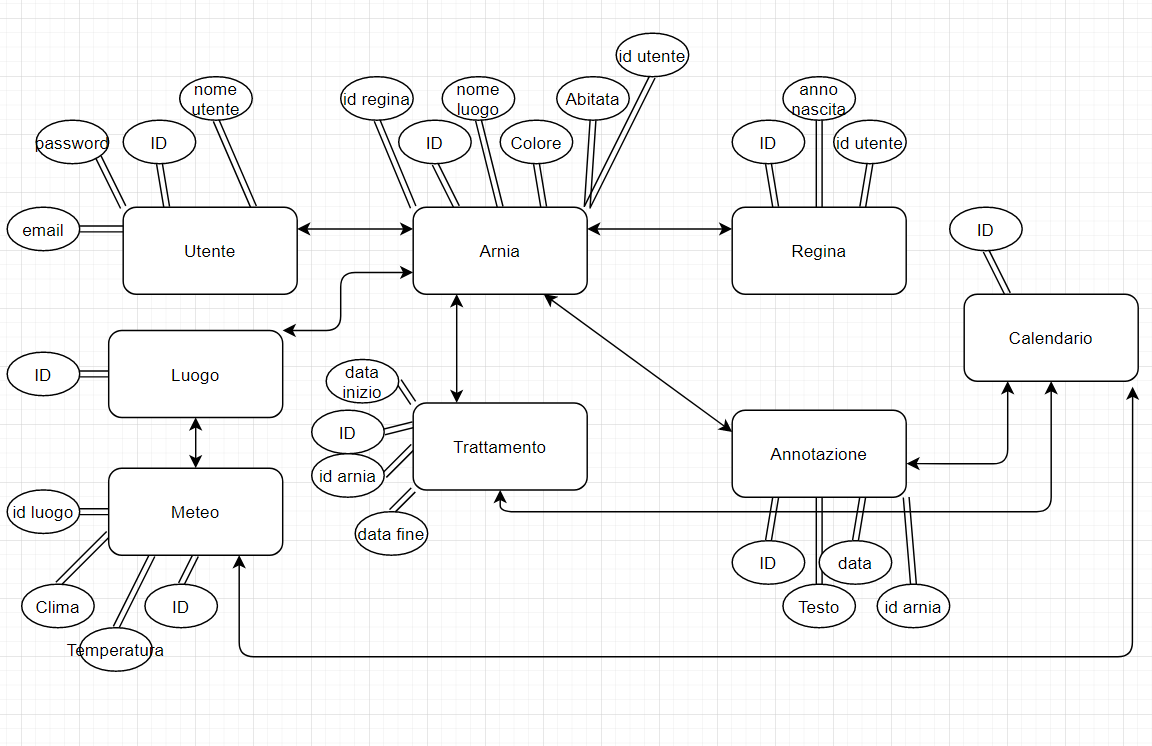
Gli utenti e i tutti i dati relativi ad uno di essi verranno salvati in un database.

Figura - Schema ER Gestione Apiario

Questo è lo schema ER utilizzato dalla banca dati, ha otto tabelle, utente avente i dati di ogni utente registrato nel sito, arnia avente i dati delle arnie di ogni utente nel sito, regina avente i dati di ogni regina, calendario avente i dati del calendario, luogo avente i dati dei luoghi dove le arnie sono situate, trattamento avente i trattamenti che sono stati applicati alle arnie, annotazione avente le note aggiuntive delle arnie e meteo avente i dati relativi al meteo di un dato luogo.

## Descrizione delle tabelle del database

|  |  |
| --- | --- |
| **Utente** | |
| Attributo | Descrizione |
| Nome utente | Il nome utente con il quale viene riconosciuto quel dato utente. Non può essere nullo. |
| Password | La password cifrata con md5 (cifratura hash a 128bit) con la quale viene effettuata la verifica di un utente. Non può essere nullo.  Es: fa838cb64417ac5d8eedc7112d54e11c |
| E-mail | L’email con la quale l’utente vuole registrarsi. Non può essere nullo. |

|  |  |
| --- | --- |
| **Arnia** | |
| Colore | Il colore con il quale l’apicoltore riconosce l’arnia. |
| Abitata (?) | Valore booleano che l’apicoltore può settare per sapere se l’arnia ha all’interno un alveare oppure no. Non può essere nullo. |
| Id\_utente | Id dell’utente (apicoltore) che possiede quest’arnia. Non può essere nullo. |
| Id\_regina | Id della regina risiedente nell’arnia se abitata. |
| Id\_luogo | Nome del luogo nella quale risiede l’arnia. Non può essere nullo. |

|  |  |
| --- | --- |
| **Regina** |  |
| Anno\_nascita | Giorno, mese ed anno di nascita della regina. Non può essere nullo. |
| Id\_arnia | Id dell’a |

|  |  |
| --- | --- |
| **Annotazione** | |
| Data | Data del giorno in cui è stata fatta quell’annotazione. Non può essere nullo. |
| Testo | Testo, contenuto dell’annotazione. Non può essere nullo. |
| Id\_arnia | Arnia alla quale è stata aggiunta quest’annotazione. Non può essere nullo. |

|  |  |
| --- | --- |
| **Trattamento** | |
| Data\_inizio | Data del giorno iniziale del trattamento. Non può essere nullo. |
| Data\_fine | Data del giorno finale del trattamento. Non può essere nullo. |
| Id\_arnia | Arnia alla quale è stata aggiunta quest’annotazione. Non può essere nullo. |

|  |  |
| --- | --- |
| **Meteo** | |
| Clima | Tipo di clima in un dato luogo ad un dato giorno (es. pioggia, neve, soleggiato, ecc.). Non può essere nullo. |
| Temperatura | La temperatura in quel giorno di un dato luogo. Non può essere nullo. |
| Id\_luogo | Luogo di dove vengono salvate le informazioni. Non può essere nullo. |

## Schema logico

Utente(id, nome\_utente, password, email)

Arnia(id, colore, abitata, nome\_luogo(FK), id\_utente(FK), id\_regina(FK))

Regina(id, anno\_nascita, id\_utente(FK))

Luogo(nome)

Meteo(id, clima, temperatura, nome\_luogo(FK))

Calendario(id)

Annotazione(id, testo, data, id\_arnia(FK))

Trattamento(id, data\_inizio, data\_fine, id\_arnia(FK))

## Design delle interfacce

Per il design delle interfacce ho utilizzato il sito [www.mockflow.com](http://www.mockflow.com)



Figura - Pagina di benvenuto

Questa è la pagina di benvenuto, ogni utente che accede al sito, come prima pagina vedrà questa.

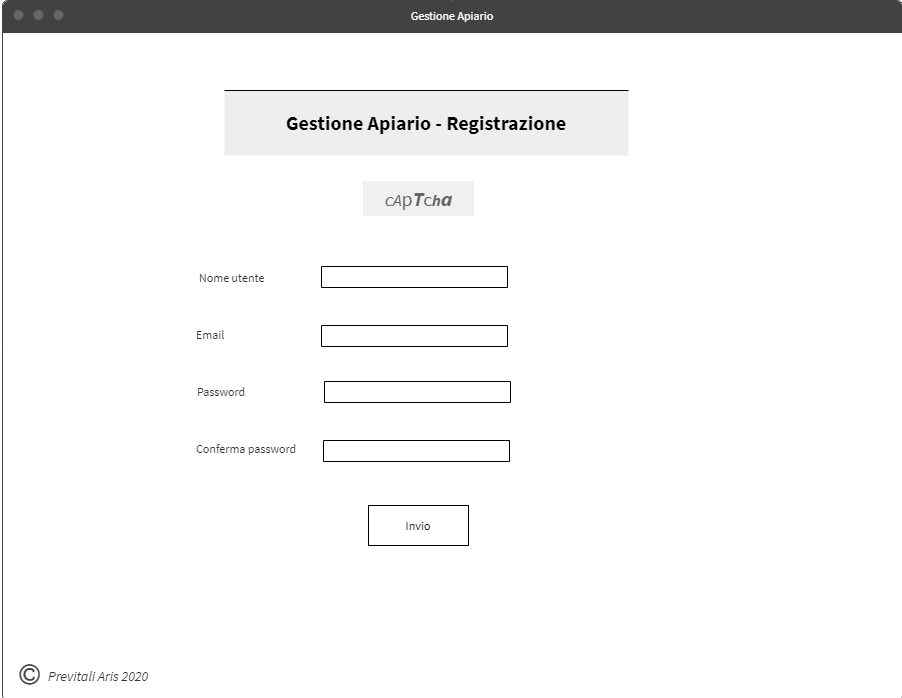


Figura - Pagina di registrazione

Questa è la pagina di registrazione dove l’utente può creare il proprio account.

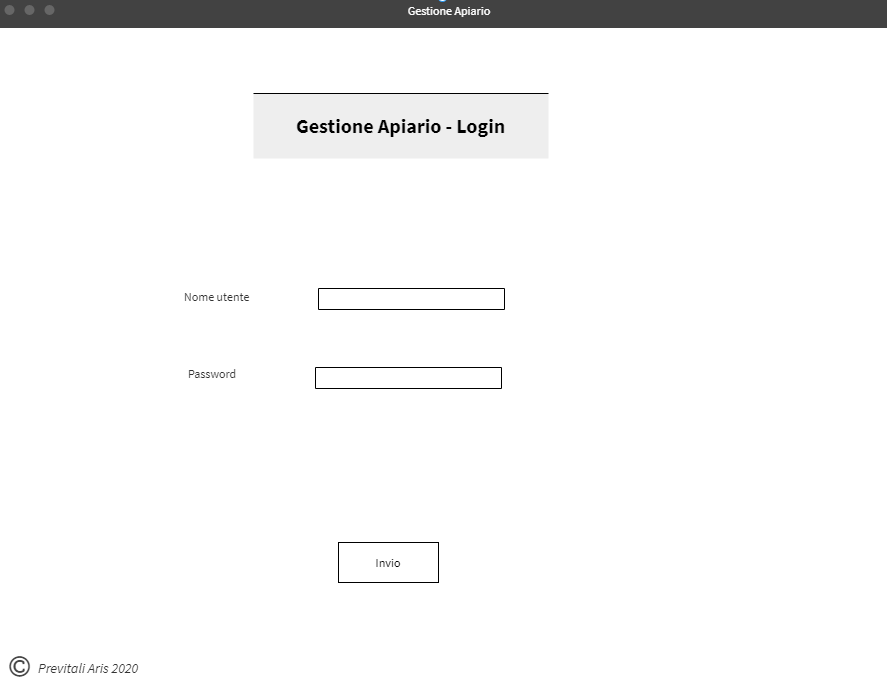


Figura - Pagina di login

Questa è la pagina di login, se l’utente ha già un account può loggarsi ed entrare nella pagina home.

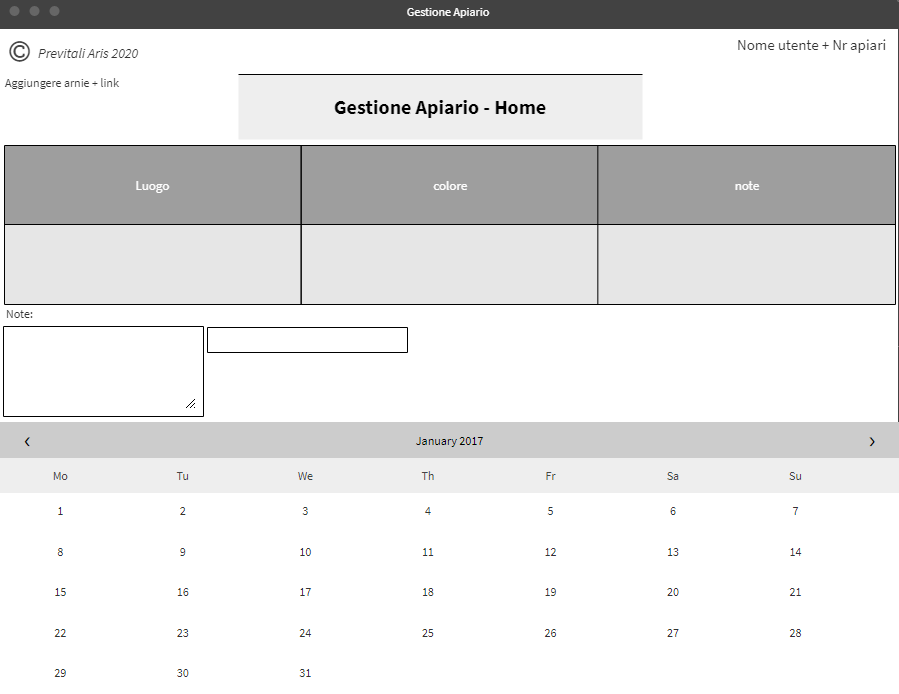


Figura - Pagina home

Questa è la pagina home, dove l’utente svolge le principali azioni possibili. Il design attuale è un po’ differente per alcuni dettagli, come l’aggiunta di alcuni bottoni come il logout, aggiunta arnia e la modifica e l’assenza del calendario.

# Implementazione

## Virtual Machine, preparazione Web Server

Prima di cominciare con la vera e propria implementazione del progetto ho creato una virtual machine alla quale ho aggiunto il file iso di Windows 10, XAMPP e tutto il resto necessario per il progetto (es: VS code, Mozilla FireFox, ecc.).  
Ho creato una macchina virtuale per la comodità di avere su un solo computer tutto il necessario per il progetto, web server, database, client.

Per il web server ho dovuto solo installare XAMPP e configurarlo.

## Database

Il database è una parte importante di tutto il progetto, è infatti il database ha tenere al suo interno tutti i dati necessari per permettere al sito di essere utile ed efficace.

Il database creato si basa sullo schema ER che ho descritto precedentemente nella sezione 3.2 design dei dati e del database, è importante che un database funzioni correttamente.

Ho dovuto cambiare più volte la struttura stessa del database siccome in all’evenienza di alcuni problemi sul sito mi sono accorto di piccoli errori di progettazione del database, fortunatamente ho solo dovuto cambiare il file SQL della creazione e non rifare tutto da capo ogni volta.

## Programmazione

Per l’implementazione di questo progetto ho utilizzato PHP principalmente, HTML, CSS e MySQL per interrogare e creare il database sulla quale si basa il progetto.

Non ho utilizzato un template MVC per un motivo principale: siccome avevo iniziato il progetto senza MVC non conoscendolo e una volta venuto a conoscenza di esso, non lo padroneggiavo molto bene e avendo già scritto un po’ di codice, ho pensato che trasferirlo nel modello MVC ci sarebbe voluto molto più tempo che continuare nel modo precedente e il tempo speso in questa operazione sommato a quello speso per capire come funzionasse un MVC fosse troppo.

Tuttavia, se dovessi rifare il progetto con le conoscenze acquisite durante esso e nei mesi nel quale era in corso ora molto probabilmente mi baserei su un template MVC siccome più comodo e molto più ordinato che singoli file da gestire manualmente.

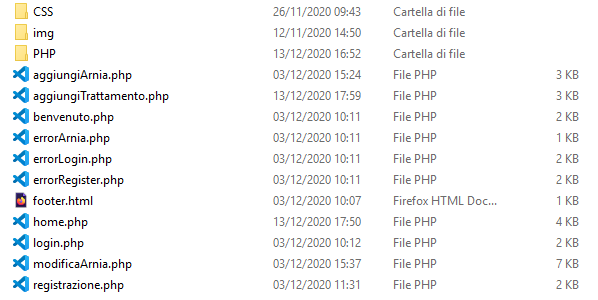
Nonostante non ho utilizzato un template MVC ho cercato di mantenere un certo ordine all’interno dei file sorgente:

Figura - Struttura file

I file nella cartella CSS servono a stabilire lo stile delle pagine del sito.  
I file nella cartella PHP servono per stabilire una connessione con quello che fa l’utente e quello che il sito, in base all’azione dell’utente, deve fare.

I file fuori dalle cartelle sono le cosiddette views, sono le pagine finale in HTML che l’utente vedrà visualizzando il sito, nonostante ciò all’interno hanno un po’ di logica di PHP.

## Implementazione sito

Per l’implementazione del sito mi sono trovato spesso a risolvere problemi di vario genere, dai più semplici da risolvere a quelli più complessi. Sono quasi sempre riuscito a risolverli, però impiegando più tempo del previsto il quale mi sarebbe servito per altre funzionalità tutt’ora non presenti nel progetto.

Non conoscendo quasi per nulla la programmazione con PHP all’inizio di questo progetto ho dovuto ricercare approfonditamente l’argomento, ma soprattutto in caso di eventuali errori il tempo impiegato per la risoluzione di esso è stato molto più ampio che se avessi dovuto fare il progetto con un linguaggio già conosciuto.

Dato questo problema nel sito mancano alcune funzionalità che dai requisiti e dalle specifiche mi sono state date, alcune funzionalità importanti che ho tenuto per la fine siccome ritenute più complesse non sono state appunto create, il calendario nella pagina home dell’utente e la funzionalità della meteo per il luogo di un’arnia, inoltre, le note per le arnie sono aggiungibili solo durante la creazione di esse e non per un dato giorno nel calendario.

Per la gestione di funzionalità che comprendono più pagine differenti e che quindi sono scollegate ho utilizzato le sessioni di PHP, le quali permettono di aprire una sessione unica per macchina che si collega al sito in modo da avere la possibilità di ogni file collegato con gli altri e la certezza che l’utente connesso sia un utente registrato.

Per la realizzazione delle relazioni dei file ho spesso utilizzato l’include di PHP per appunto avere i file inerenti ad una singola attività che ho avuto il bisogno di utilizzare spesso in un altro file utilizzato per più attività connesse fra loro, ad esempio la connessione al database.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-001  REQ-009 | Nome: | Gestione multiutente con login e registrazione |
| Descrizione: | Testare le funzionalità di registrazione e di login del sito web | | |
| Prerequisiti: |  | | |
| Procedura: | 1. Andare nella pagina di benvenuto sul sito 2. Andare nella pagina di registrazione tramite l’apposito bottone 3. Immettere le volute credenziali 4. Interrogare il database per scoprire se sono stati aggiunti i dati corretti 5. Ripetere l’operazione con il login, controllare se viene fatto il re indirizzamento nella pagina home | | |
| Risultati attesi: | Dati corretti nel database nella registrazione, re indirizzamento nella pagina home nel login. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-002  REQ-002  REQ-003  REQ-008 | Nome: | Aggiunta di un’arnia |
| Descrizione: | Testare la funzionalità di aggiunta di un’arnia | | |
| Prerequisiti: |  | | |
| Procedura: | 1. Andare nella pagina home del sito 2. Andare nella pagina di aggiunta arnia tramite l’apposito bottone 3. Immettere i dati voluti dell’arnia 4. Interrogare il database per scoprire se sono stati aggiunti i dati corretti | | |
| Risultati attesi: | Dati corretti nel database. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-003  REQ-002  REQ-003  REQ-008 | Nome: | Modifica o eliminazione di un’arnia |
| Descrizione: | Testare la funzionalità di modifica ed eliminazione di un’arnia | | |
| Prerequisiti: |  | | |
| Procedura: | 1. Andare nella pagina home del sito 2. Andare nella pagina di modifica / eliminazione arnia tramite l’apposito bottone 3. Immettere i dati voluti dell’arnia 4. Interrogare il database per scoprire se i dati sono stati modificati correttamente 5. Eliminare un’arnia 6. Interrogare il database per scoprire se l’arnia è stata eliminata correttamente | | |
| Risultati attesi: | Dati corretti nel database. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-004  REQ-005 | Nome: | Aggiunta trattamento ad un’arnia |
| Descrizione: | Testare la funzionalità di aggiunta di un trattamento per un’arnia | | |
| Prerequisiti: |  | | |
| Procedura: | 1. Andare nella pagina home del sito 2. Cliccare su *aggiungi trattamento* nell’arnia scelta 3. Aggiungere i dati 4. Interrogare il database per scoprire se i dati sono stati aggiunti correttamente 5. Controllare dopo la data di fine se il sistema notifica l’utente | | |
| Risultati attesi: | Dati corretti nel database. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-005  REQ-006 | Nome: | Aggiunta annotazione ad un’arnia |
| Descrizione: | Testare la funzionalità di aggiunta di un’annotazione per un’arnia | | |
| Prerequisiti: |  | | |
| Procedura: | 1. Andare nella pagina home del sito 2. Cliccare su *aggiungi nota* nell’arnia scelta 3. Aggiungere i dati 4. Interrogare il database per scoprire se i dati sono stati aggiunti correttamente | | |
| Risultati attesi: | Dati corretti nel database. | | |

## Risultati test

|  |  |  |  |
| --- | --- | --- | --- |
| **ID Test-case** | **Nome Test-case** | **Risultato** | **Note** |
| TC-001 | Gestione multiutente con login e registrazione |  |  |
| TC-002 | Aggiunta di un’arnia |  |  |
| TC-003 | Modifica o eliminazione di un’arnia |  |  |
| TC-004 | Aggiunta trattamento ad un’arnia |  | Punto 5 non implementato nel codice sorgente |
| TC-005 | Aggiunta annotazione ad un’arnia |  |  |

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

All’interno del progetto ci sono due grossi elementi mancanti, il primo è l’assenza del calendario nella pagina home del sito, il secondo sono le informazioni meteo del luogo dove risiedono le arnie che si sarebbero dovute vedere all’interno del calendario, Vi sono altri due elementi non implementati al 100%, uno è la notifica di un trattamento al giorno della sua fine, il secondo è la visualizzazione delle note sull’arnia all’interno del calendario siccome esso non è implementato.

Ci sono queste mancanze perché’ non avendo calcolato bene i tempi degli altri elementi, e avendo lasciato il calendario e la meteo per la fine, siccome ritenuti più complicati, non sono riuscito ad avere il tempo necessario per implementarli nel sito.

# Gantt Consuntivo

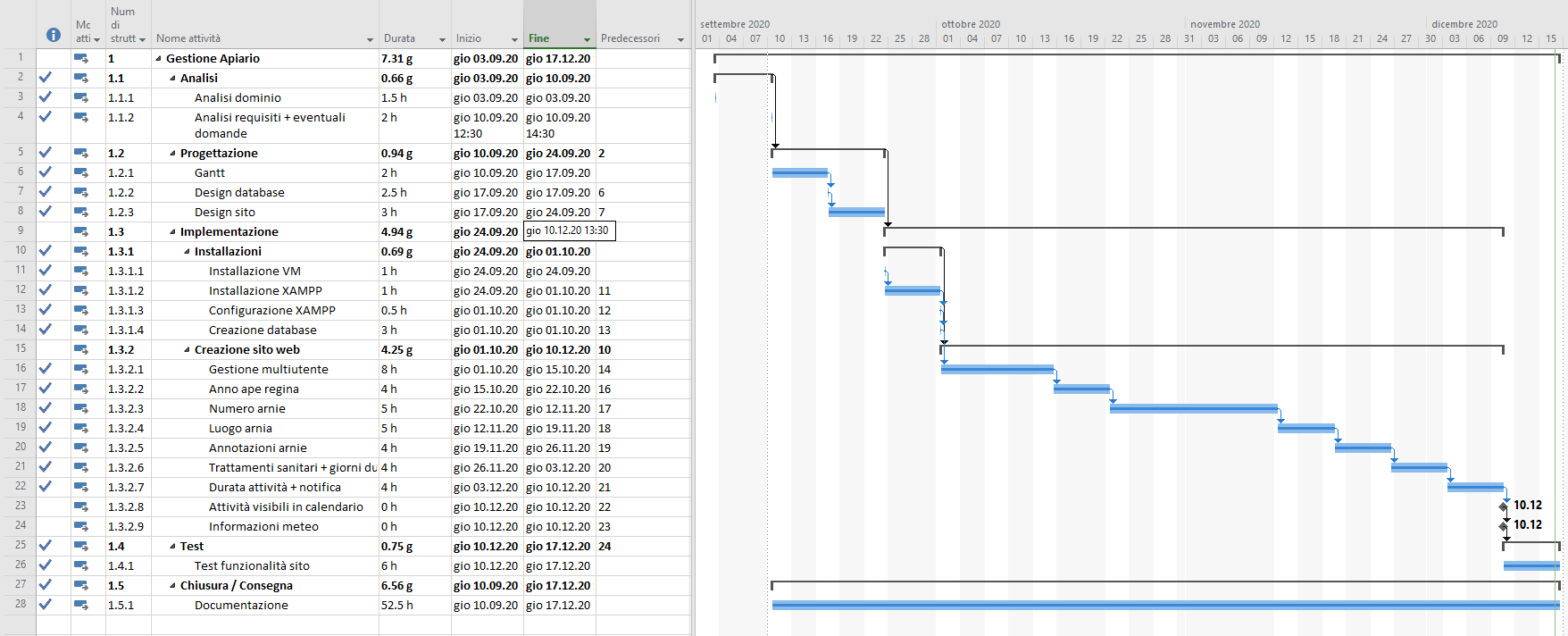


Figura - Gantt Consuntivo

## Differenze con il gantt preventivo

Nel gantt consuntivo ci sono un po’ di differenze con quello preventivo esplicato nel capitolo *2.4 Pianificazione,* infatti come si può vedere due punti (1.3.2.8; 1.3.2.9) della creazione del sito web non sono stati completati, sono infatti il calendario e le informazioni meteo.  
Come spiegato nel punto *5.3 Mancanze/limitazioni* conosciute non sono implementate per mancanza di tempo.

Inoltre ci sono due punti (1.3.2.1; 1.3.2.2) sono stati spostati e fatti come primi elementi del sito, questo solo per una questione di comodità, siccome uno gestisce il multiutente e che quindi averlo fin dall’inizio mi è sembrato logico e più comodo.

Per i punti d’implementazioni le ore di lavoro sono aumentate parecchio, sia per una mia difficoltà nel gestire il tempo all’inizio del progetto sia per la mancata conoscenza approfondita del linguaggio utilizzato. Ciò mi ha fatto allungare ogni attività e quindi non ho avuto tempo per implementare gli ultimi due punti.

# Conclusioni

Secondo me, in questo progetto ho imparato molte cose, sia specifiche al progetto come ad esempio l’uso di PHP e MySQL ma anche cose più generali come ad esempio l’organizzazione, la progettazione di un progetto o, rimanendo nel campo informatico, l’uso di GitHub.

È stato utile anche perché’ grazie a questo progetto ho imparato a come dovrei gestire il tempo.

È stato il mio primo progetto nella quale ho impiegato tutte le mie capacità, essendo da solo, e aver portato a termine qualcosa di funzionante, nonostante alcuni errori o mancanze, sono fiero del lavoro svolto.

## Sviluppi futuri

Sicuramente uno sviluppo per il futuro sarebbe quello di portare a termine l’intero progetto con il calendario e la meteo. Ma anche quello di rifare l’intero progetto con un template MVC in modo da avere tutto più organizzato e meglio presentabile.

Inoltre, si potrebbe aggiungere uno stile alle pagine per rendere il sito più responsive e più bello agli occhi dell’utente.

## Considerazioni personali

Da questo progetto ho imparato, come detto prima, alcuni lati strettamente collegati al progetto come PHP e MySQL, e dei lati più generali, come la gestione del tempo, la progettazione e l’indipendenza nel lavoro.

Dovessi rifare il progetto certamente, ora che ho le conoscenze, sceglierei un template MVC per PHP e HTML in modo da avere il tutto più organizzato, proverei anche a progettare meglio il tempo per l’implementazione siccome non sono riuscito a soddisfare tutte le richieste.

# Bibliografia

## Sitografia

* [www.php.net](http://www.php.net), Consultato durante l’intera durata del progetto
* [www.w3schools.com](http://www.w3schools.com), Consultato durante l’intera durata del progetto
* [stackoverflow.com](https://stackoverflow.com), Consultato durante l’intera durata del progetto

# Allegati

Gli allegati sono nel repository di github, questa è una lista con i percorsi corretti:

* Diari di lavoro: GestioneApiario/diari
* QdC\_Primo Semestre – GestioneApiario: GestioneApiario/documenti
* Codici sorgente: GestioneApiario/src
* Abstract: GestioneApiario/AbstractGestioneApiario

# Indice delle figure

[Figura 1 - Schema use case 6](#_Toc59109405)

[Figura 2 - Diagramma di Gantt 7](#_Toc59109406)

[Figura 3 - Analisi, Gantt 8](#_Toc59109407)

[Figura 4 - Progettazione, Gantt 8](#_Toc59109408)

[Figura 5 - Implementazione, Gantt 9](#_Toc59109409)

[Figura 6 - Test, Gantt 9](#_Toc59109410)

[Figura 7 - Sitemap, Gestione Apiario 11](#_Toc59109411)

[Figura 8 - Schema ER Gestione Apiario 12](#_Toc59109412)

[Figura 9 - Pagina di benvenuto 14](#_Toc59109413)

[Figura 10 - Pagina di registrazione 15](#_Toc59109414)

[Figura 11 - Pagina di login 16](#_Toc59109415)

[Figura 12 - Pagina home 17](#_Toc59109416)

[Figura 13 - Struttura file 19](#_Toc59109417)

[Figura 14 - Gantt Consuntivo 24](#_Toc59109418)