

*Acchiardi Paolo**5AROB**31/05/2021*

Elaborato “Gestione sicurezza accesso eventi”

Indicazioni per lo svolgimento dell'elaborato

Per fornire la soluzione al quesito proposto, il candidato dovrà sfruttare le competenze acquisite nelle discipline “SISTEMI E RETI” ed “INFORMATICA”, attenendosi alla traccia di svolgimento di seguito indicata e toccando obbligatoriamente tutti i punti in essa indicati.

Il candidato è libero di individuare spunti di collegamento con le altre discipline dell'area informatica-smart robot (compreso PCTO o educazione civica) che potranno essere trattati o implementati anche nel dettaglio. In particolare può indicare le criticità del progetto e monitorarne l'avanzamento con le tecniche del WBS, Gantt e CPM.

Il candidato è tenuto a:

- motivare ognuna delle ipotesi aggiuntive e ognuna delle scelte progettuali che abbia deciso di adottare in ciascuno dei punti della traccia.
- svolgere in lingua inglese uno degli aspetti richiesti al punto 1 della traccia di svolgimento

Traccia di svolgimento:

1. Obbligatoriamente almeno tre tra gli aspetti in elenco:
 - A. Infrastruttura di rete proposta (a scelta tra cloud oppure data center in-house): rappresentazione grafica obbligatoria e descrizione obbligatoria
 - B. Protocolli di rete utilizzati: descrizione obbligatoria
 - C. Sicurezza della infrastruttura di rete proposta: descrizione obbligatoria
 - D. Gestione della privacy dei dati: descrizione obbligatoria
 - E. Utilizzo delle funzioni di hash oppure degli algoritmi crittografici, a scelta: descrizione obbligatoria, esempio di implementazione obbligatori.
2. Progettare la base di dati per la gestione del caso assegnato con indicazione del modello concettuale e del corrispondente modello logico. Verificare che le tabelle ottenute siano normalizzate.
3. Implementare il database in MariaDB/MySQL con relative istruzioni in SQL per la creazione delle tabelle necessarie (curare i check) e inserire dei dati di esempio.
4. Realizzare il sito/web app che permetta la navigazione di tutte le funzioni anche se in "working progress" e permetta l'utilizzo completo di un segmento significativo dell'applicazione Web che consenta l'interazione con la base di dati.
5. Inoltre realizzare le pagine che visualizzano il risultato delle query del punto successivo, utilizzando almeno una table. Utilizzare PHP e PDO obbligatoriamente mentre si curi l'aspetto grafico come si preferisce (con solo HTML e CSS o con bootstrap)
6. Redigere le interrogazioni espresse in linguaggio SQL indicate nel testo, utilizzando anche parametri forniti tramite le pagine web realizzate.

Ulteriori indicazioni per lo svolgimento e la consegna

La prova deve essere svolta singolarmente, dovrà essere redatta interamente in formato elettronico e NON dovrà contenere scansioni di documenti cartacei, manoscritti oppure rappresentazioni grafiche tracciate manualmente.

La prova dovrà essere consegnata in formato .zip completa del pdf, dei file delle pagine web richieste e del database MySQL utilizzato (denominazione file: elaborato_cognome_nome_classe.zip, es. elaborato_Rossi_Mario_5AROB.zip), all'indirizzo di posta elettronica istituzionale dei docenti di indirizzo e anche all'indirizzo della casella di posta elettronica istituzionale che sarà indicata in apposita circolare, tassativamente entro le ore 12:00 di lunedì 31 maggio, dal proprio account personale con estensione .itiscuneo.eu:

- simone.conradi@itiscuneo.eu
- roberta.molinari@itiscuneo.eu
- claudio.borgogno@itiscuneo.eu

Acchiardi Paolo

Gestione sicurezza accessi eventi

L'Agenzia di Organizzazione Eventi "Granda Evento snc" vuole gestire internamente la sicurezza di eventi come concerti, serate in discoteca, ecc. che organizza, anziché subappaltare esternamente come fatto finora.

1. Il sistema di sicurezza di un evento richiede le seguenti componenti:
 - telecamere IP ad alta risoluzione, collegate a sistema di videoregistrazione, monitorate da sala di controllo (vedi più avanti)
 - altoparlanti VoIP per invio messaggi in tutta l'area coperta dall'evento
 - sensori di fumo
 - metal detector nei pressi degli accessi
 - telecamera per face-detection dotata di scanner di temperatura corporea nei pressi degli accessi
 - sala di controllo, mobile su furgone, con postazione multimonitor per osservazione, coordinamento personale sicurezza e chiamata soccorsi.
 - infrastruttura di rete utile a realizzare la connettività necessaria
2. affissione cartelli presenza telecamere (obbligo legale, ma utile anche a scopo dissuasivo);
3. presenza personale in ruolo di "buttafuori" e di personale in ruolo di "primo soccorso" (prevedendo che il medesimo individuo, se addestrato, possa svolgere ambo i ruoli), con ogni soggetto collegato via cuffiette e microfono alla sala di controllo;

Si rende pertanto necessaria una base di dati MYSQL (installata nella sede della "Granda Evento snc") per:

- elenco personale con dati anagrafici e tipi di mansioni che può ricoprire;
- elenco mansioni necessarie per un certo evento (ad esempio: buttafuori, addetto primo soccorso, attacchino cartelli, operatore monitor, autista furgone del centro di controllo mobile, tecnico riparatore apparati elettronici, capo responsabile sicurezza evento). Il personale può assumere i vari ruoli nei vari eventi e venire retribuito in base alle ore lavorate secondo una paga oraria stabilita per ogni mansione;
- elenco paga oraria per mansione;
- elenco eventi, durata in giorni ed in ore medie per giorno, budget totale previsto per ciascuno;
- elenco esigenze singolo evento in termini di mansioni necessarie e numero elementi per singola mansione, e relativa previsione di spesa.
- conteggio accessi dei partecipanti all'evento in tempo reale tramite telecamera che registra anche il luogo in cui avviene l'accesso

Il sistema deve essere in grado di permettere:

- gestione anagrafiche eventi, mansioni e personale, con funzionalità CRUD;
- su singolo evento imputare numero e tipo di mansioni che è necessario coprire, calcolando l'importo necessario e la percentuale di questo sul budget dell'evento e consentendo modifiche anche a seguito di variazioni delle anagrafiche.
- produzione statistiche storiche di affluenza negli eventi.

Queste le interrogazioni da fornire in linguaggio SQL e da implementare in PHP oltre a quelle richieste per la realizzazione del sistema:

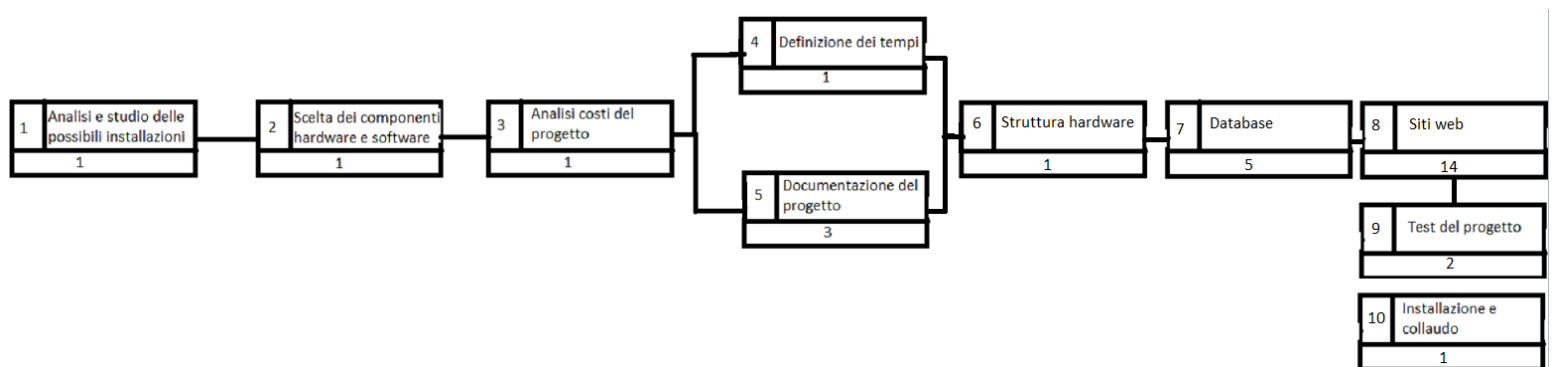
1. Elenco degli eventi del prossimo mese e numero di personale attualmente ingaggiato e quello mancante
2. L'evento che ha registrato il minor numero di ingressi
3. Il personale che è abile per le stesse mansioni dell'addetto X che va sostituito nell'evento Y

GPOI

Per monitorare l'avanzamento del progetto ho utilizzato le tecniche del WBS, del PDM e del Gantt:

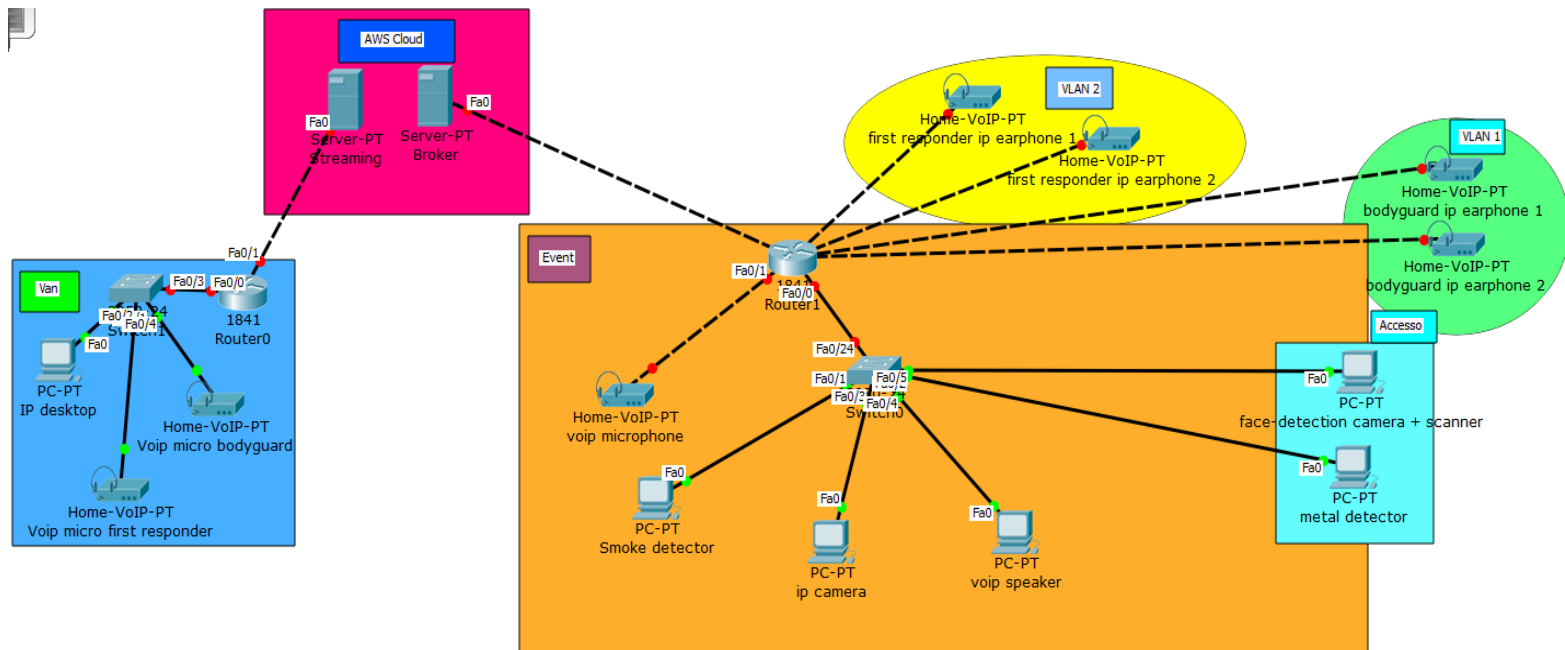
| Work Breakdown Structure (WBS) | | | |
|--------------------------------|--|-----------------|-----------------|
| Progetto | Sicurezza eventi "Granda Evento snc" | Codice progetto | GE00 |
| Data | 1 maggio 2021 | Revisione | 0 |
| Cliente | Granda Evento snc | Sponsor | Acchiardi Paolo |
| WBS | Descrizione | Durata (gg) | |
| 1.0 Progetto | | | |
| 1.1 Avvio | | | |
| 1.01.01 | Analisi e studio delle possibili installazioni | 1 | |
| 1.01.02 | Scelta dei componenti hardware e software | 1 | |
| 1.01.03 | Analisi costi del progetto | 1 | |
| 1.2 Pianificazione | | | |
| 1.02.01 | Definizione dei tempi | 1 | |
| 1.02.02 | Documentazione del progetto | 3 | |
| 1.3 Realizzazione | | | |
| 1.03.01 | Struttura hardware | 1 | |
| 1.03.02 | Database (Mysql) | 5 | |
| 1.03.03 | Siti web (PHP) | 14 | |
| 1.03.04 | Test del progetto | 2 | |
| 1.4 Chiusura | | | |
| 1.04.01 | Installazione e collaudo | 1 | |

PDM (Product Data Management)



Gantt diagram

| | da | a | gg | 1/5 | 2/5 | 3/5 | 4/5 | 5/5 | 6/5 | 7/5 | 8/5 | 9/5 | 10/5 | 11/5 | 12/5 | 13/5 | 14/5 | 15/5 | 16/5 | 17/5 | 18/5 | 19/5 | 20/5 | 21/5 | 22/5 | 23/5 | 24/5 | 25/5 | 26/5 | 27/5 | 28/5 | 29/5 | 30/5 |
|--|------------|------------|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 1 Analisi e studio delle possibili installazioni | 01/05/2021 | 01/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 Scelta dei componenti hardware e software | 02/05/2021 | 02/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 Analisi costi del progetto | 03/05/2021 | 03/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 Definizione dei tempi | 04/05/2021 | 04/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 Documentazione del progetto | 04/05/2021 | 06/05/2021 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 Struttura hardware | 07/05/2021 | 07/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 Database | 08/05/2021 | 12/05/2021 | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 Siti web | 13/05/2021 | 26/05/2021 | 14 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 Test del progetto | 27/05/2021 | 28/05/2021 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 Installazione e collaudo | 29/05/2021 | 29/05/2021 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

SISTEMI E RETI - INGLESE**Network infrastructure:**

In this project i preferred to use cloud instead of data center in-house because cloud has a lot of advantages such as:

- *lower costs;*
- *scalability;*
- *flexibility;*
- *etc...*

Each sensor in the event area that is physically stationary (smoke detectors, ip cameras, voip speakers, face-detection cameras, metal detectors) is connected to a switch by cable which in turn is connected to the router. Other sensors (voip microphones, ip earphones) are connected to the router.

These sensors send data to a Broker that is located on AWS Cloud.

In the van there are 3 devices:

- *pc with which you can view the footage of the ip cameras in real time (there is a server written in python and Flask on AWS Cloud);*
- *voip microphones connected to the bodyguards ip earphones;*
- *voip microphones connected to the first responders ip earphones.*

Voip microphones broadcast data on the corresponding VLAN.

In the event area there is a voip microphone controlled by an assistant that send data to a voip speaker.

Gestione della privacy dei dati:

Una volta che i sensori inviano i dati al broker, i dati vengono memorizzati su un database MySQL.

Di seguito elenco i sensori che inviano dati da salvare sul database e i tipi di dati che mandano:

- *sensori di fumo, inviano un dato al secondo e quando esso è superiore a 600 bisogna far scattare l'allarme antincendio;*
- *metal detectors*
- *telecamera con il riconoscimento facciale + scanner*

Per la gestione dei dati senza incorrere in sanzioni è sufficiente affiggere dei cartelli presenza telecamere vicino alle telecamere (agli accessi dell'area dell'evento e all'interno).

Per il rispetto del GDPR sarà necessario evitare la vendita dei dati e delle informazioni presenti sui database a terzi e scrivere un documento in cui sono contenute determinate informazioni quali: chi sono coloro che hanno accesso ai dati, i tipi di informazioni presenti sui database, per quanto tempo i dati saranno memorizzati.

Utilizzo delle funzioni di hash:

Quando si accede al sito web che permette di modificare l'organizzazione dell'evento e di visualizzare determinate statistiche viene richiesto il nome utente e la password. I due parametri, che sono salvati su un'apposita tabella del database MySQL, vengono richiesti per impedire ad una persona qualunque che visualizza il sito di poter modificare l'organizzazione dell'evento.

Ovviamente per evitare che in caso di furti di dati vengano rubate le credenziali la password, prima di essere memorizzata sul database, viene fatta passare attraverso l'algoritmo di hashing MD5, il quale in uscita produrrà una stringa di lunghezza fissa di 128 caratteri.

Sicurezza dell'infrastruttura di rete proposta:

Per garantire una buona sicurezza dell'infrastruttura di rete è necessario inserire due firewall sui due router (quello dentro al van e quello situato nell'area dell'evento). Il firewall all'interno dell'area dell'evento (router A) garantirà l'accesso solo ai sensori dell'infrastruttura di rete impedendo così alle persone non autorizzate di accedere e di inondare la rete di dati, se proprio si vorrà offrire un servizio wifi alla clientela si installerà un ulteriore router apposito.

La tabella del firewall del router A sarà strutturata nel seguente modo:

| <i>id</i> | <i>protocol</i> | <i>source ip</i> | <i>destination ip</i> | <i>destination port</i> | <i>action</i> |
|-----------|-----------------|--|-----------------------|-------------------------|---------------|
| 1 | TCP | broker ip / server streaming ip | sensor ip | 8080 | accept |
| 2 | Any | Any | Any | Any | deny |

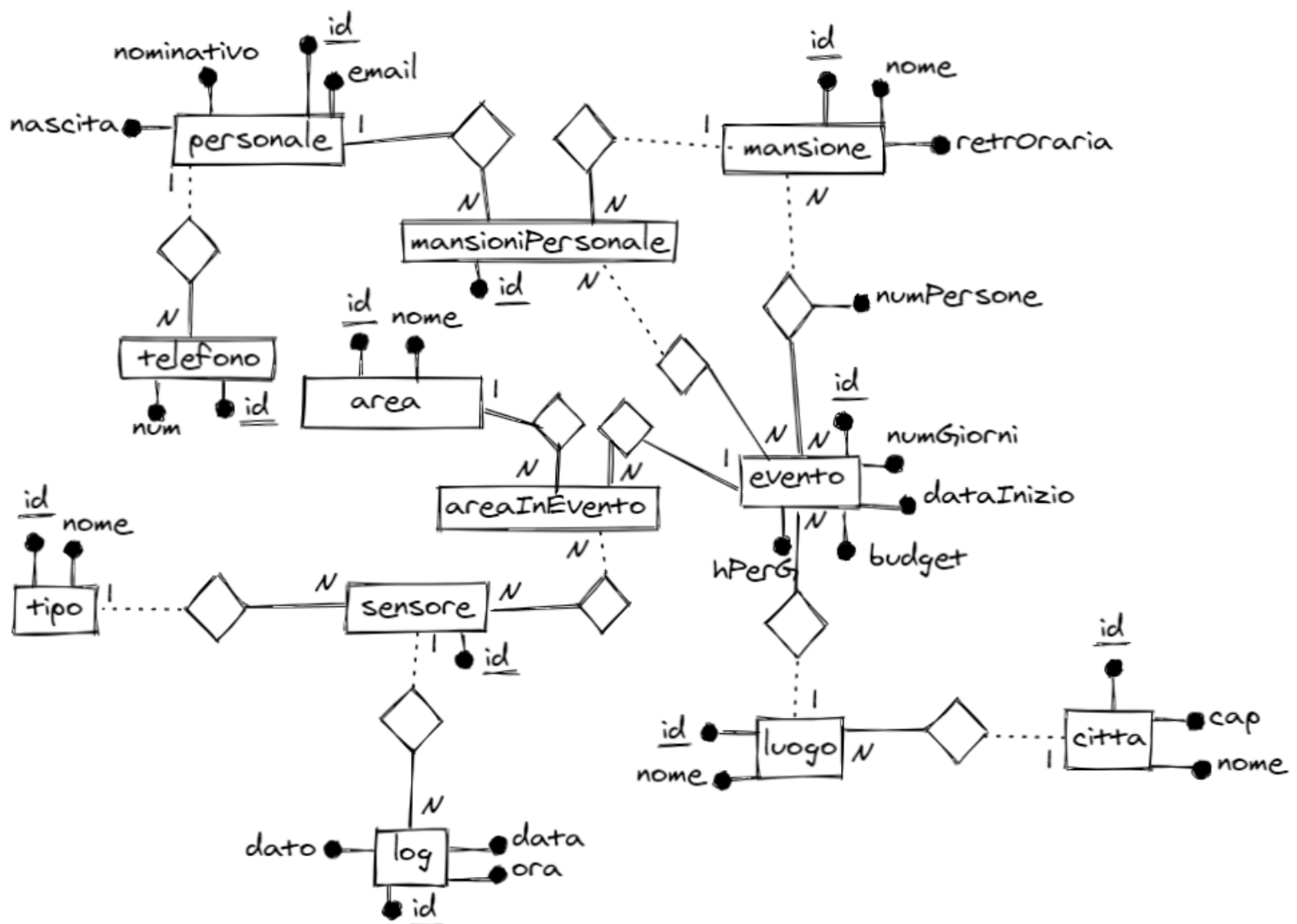
Il firewall sul router B consentirà la connessione solo ai desktop utili per il collegamento con le videocamere ip e ai microfoni utili per comunicare con gli addetti al primo soccorso e i bodyguard.

Dunque la tabella del firewall del router B risulterà essere strutturata nel seguente modo:

| <i>id</i> | <i>protocol</i> | <i>source ip</i> | <i>destination ip</i> | <i>destination port</i> | <i>action</i> |
|-----------|-----------------|---------------------|-----------------------------|-------------------------|---------------|
| 1 | TCP | server streaming ip | desktop ip, microphone s ip | 8080 | accept |
| 2 | Any | Any | Any | Any | deny |

INFORMATICA

SCHEMA E-R:



SUPPOSIZIONI / APPUNTI:

- Come primary key ho inserito i campi "id" per facilitare l'inserimento dei dati nelle altre tabelle (ad es. in "Telefoni" inserisco solo l'id della persona e non il codice fiscale);
- ho supposto che ogni persona deve inserire una email per essere contattato ma può scegliere se inserire 0 o più numeri di telefono;
- in "Eventi" ho deciso di non inserire la colonna "dataFine" perchè si calcola facilmente sommando "dataInizio" e "numGiorni";
- "Luoghi" contiene i nomi dei luoghi in cui si svolgono i vari eventi (es. stadio San Siro di Milano);
- ho creato la tabella "Citta" per evitare di scrivere in modo differente i nomi delle città e generare problemi nel recuperare dei dati (es. "Borgo San Dalmazzo" potrei scriverlo "Borgo S. Dalmazzo" ma grazie alla tabella "Citta" si evita questo problema);
- "Aree" indica i vari spazi di un evento (es. accessoPrincipale, scalaUscitaEmergenza ecc.);
- Non si possono inserire eventi che non richiedono personale;
- In fase di creazione di un evento non si possono installare dei sensori impegnati in un altro evento, solamente quando l'altro evento sarà concluso si potranno utilizzare quei sensori per nuovi eventi;
- le associazioni tra "Aree", "Sensori", "Eventi" sono N:N perchè su un database un sensore può essere registrato in più aree diverse, per esempio un sensore può trovarsi in un'area in una certa data ma se si crea un nuovo evento il prossimo mese si può assegnare lo stesso sensore a quel futuro evento. Dunque si crea un'entità "areaInEvento";
- La tabella "PersonaleInEventi" contiene l'elenco delle persone nei vari eventi e con i vari ruoli a loro assegnati, non esistono righe contenenti la stessa persona nello stesso evento che ricopre lo stesso ruolo;
- Nel db ho creato una tabella "Login" per verificare che le persone che accedono al sito di gestione degli eventi siano autorizzate.

MAPPING:

Telefoni(id, num, idPersona*)

Personale(id, email, nome, cognome, nascita)

Mansioni(id, nome, retrOraria)

Eventi(id, nome, numGiorni, dataInizio, budget, hPerG, idLuogo*)

Luoghi(id, nome, idCitta*)

Citta(id, cap, nome)

Aree(id, nome)

Sensori(id, idTipo*)

Log(id,data,ora,datao, idSensore)*
Tipi(id,nome)
MansioniPersonale(id,idMansione,idPersona*)*
MansioniInEventi(id, idEvento, idMansione*, numPersone)*
PersonaleInEventi(id,idMansionePersona, idEvento*)*
AreeInEventi(id,idArea,idEvento*)*
SensoriInEventi(id,idSensore,idAreaInEvento*)*
Login(id,user,pw)

CREAZIONE DELLE TABELLE NEL DATABASE:

Per iniziare la programmazione del database eseguo le varie query per creare le tabelle:

```
CREATE TABLE `login` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `user` varchar(255) NOT NULL UNIQUE,  
  `pw` varchar(255) NOT NULL,  
  PRIMARY KEY (`id`)  
)
```

```
CREATE TABLE `aree` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `nome` varchar(255) NOT NULL UNIQUE,  
  PRIMARY KEY (`id`)  
)
```

```
CREATE TABLE `citta` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `cap` int(11) NOT NULL UNIQUE CHECK (`cap` > 0),  
  `nome` varchar(255) NOT NULL,  
  PRIMARY KEY (`id`)  
)
```

```
CREATE TABLE `luoghi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `nome` varchar(255) NOT NULL,  
  `idCitta` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  FOREIGN KEY (`idCitta`) REFERENCES `citta`(`id`),  
  UNIQUE KEY (`nome`,`idCitta`)  
)
```

```
CREATE TABLE `eventi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `nome` varchar(255) NOT NULL,  
  `numGiorni` int(11) NOT NULL CHECK (`numGiorni` > 0),  
  `dataInizio` date NOT NULL,
```

```
`budget` double NOT NULL CHECK (`budget` > 0),
`hPerG` int(11) NOT NULL CHECK (`hPerG` > 0),
`idLuogo` int(11) NOT NULL,
PRIMARY KEY (`id`),
UNIQUE KEY (`nome`),
UNIQUE KEY (`dataInizio`, `idLuogo`),
FOREIGN KEY (`idLuogo`) REFERENCES `luoghi` (`id`)
)
```

```
CREATE TABLE `areeineventi` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `idArea` int(11) NOT NULL,
  `idEvento` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`idArea`, `idEvento`),
  FOREIGN KEY (`idArea`) REFERENCES `aree` (`id`),
  FOREIGN KEY (`idEvento`) REFERENCES `eventi` (`id`)
)
```

```
CREATE TABLE `mansioni` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nome` varchar(255) NOT NULL UNIQUE,
  `retrOraria` double NOT NULL CHECK (`retrOraria` > 0),
  PRIMARY KEY (`id`)
)
```

```
CREATE TABLE `mansioniineventi` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `idEvento` int(11) NOT NULL,
  `idMansione` int(11) NOT NULL,
  `numPersone` int(11) NOT NULL DEFAULT 0 CHECK (`numPersone` > -1),
  PRIMARY KEY (`id`),
  UNIQUE KEY (`idEvento`, `idMansione`),
  FOREIGN KEY (`idEvento`) REFERENCES `eventi` (`id`),
  FOREIGN KEY (`idMansione`) REFERENCES `mansioni` (`id`)
)
```

```
CREATE TABLE `personale` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `email` text NOT NULL CHECK (`email` like '%@%.%') UNIQUE,
  `nome` text NOT NULL,
  `cognome` text NOT NULL,
  `nascita` date NOT NULL CHECK (`nascita` < CURDATE()),
  `cf` varchar(255) DEFAULT NULL CHECK (`cf` like '_____') UNIQUE,
  PRIMARY KEY (`id`)
)
```

```
CREATE TABLE `mansionipersonale` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `idMansione` int(11) NOT NULL,
  `idPersona` int(11) NOT NULL,
  PRIMARY KEY (`id`),
```

```
UNIQUE KEY (`idMansione`, `idPersona`),  
FOREIGN KEY (`idMansione`) REFERENCES `mansioni` (`id`),  
FOREIGN KEY (`idPersona`) REFERENCES `personale` (`id`)  
)
```

```
CREATE TABLE `personaleineventi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `idMansionePersona` int(11) NOT NULL,  
  `idEvento` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE KEY (`idMansionePersona`, `idEvento`),  
  FOREIGN KEY (`idMansionePersona`) REFERENCES `mansionipersonale` (`id`),  
  FOREIGN KEY (`idEvento`) REFERENCES `eventi` (`id`)  
)
```

```
CREATE TABLE `tipi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `nome` varchar(255) NOT NULL UNIQUE,  
  PRIMARY KEY (`id`)  
)
```

```
CREATE TABLE `sensori` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `idTipo` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  FOREIGN KEY (`idTipo`) REFERENCES `tipi` (`id`)  
)
```

```
CREATE TABLE `log` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `idSensore` int(11) NOT NULL,  
  `data` date NOT NULL,  
  `ora` time NOT NULL,  
  `dato` varchar(255) NOT NULL,  
  PRIMARY KEY (`id`),  
  FOREIGN KEY (`idSensore`) REFERENCES `sensori` (`id`)  
)
```

```
CREATE TABLE `sensoriineventi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `idSensore` int(11) NOT NULL,  
  `idAreaInEvento` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE (`idSensore`, `idAreaInEvento`),  
  FOREIGN KEY (`idSensore`) REFERENCES `sensori` (`id`),  
  FOREIGN KEY (`idAreaInEvento`) REFERENCES `areeineventi` (`id`)  
)
```

```
CREATE TABLE `telefoni` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `num` varchar(255) NOT NULL CHECK (`num` like '____') UNIQUE,  
  `idPersona` int(11) NOT NULL,
```

```
PRIMARY KEY (`id`),  
FOREIGN KEY (`idPersona`) REFERENCES `personale` (`id`)  
)
```

La tabella "login" serve per controllare che le credenziali siano valide in fase di accesso al sito web, la password e il nome utente è lo stesso per tutto il personale che si occupa della gestione della sicurezza dei vari eventi dato che le funzionalità del sito web non variano in base alla persona che accede al sito.

La tabella "tipi" contiene i vari tipi di sensori che necessitano la gestione dei loro dati, in questo caso i tipi sono 3: "ip camera", "metal detector", "smoke detector".

La tabella "log" contiene tutti i dati che i sensori inviano durante i vari eventi.

STRUTTURA DEL SITO WEB

Il sito è strutturato ad albero e tramite bootstrap creo in ogni sottopagina il percorso fatto per arrivarci, per poter accedere ad una pagina di un ramo superiore basterà cliccare sulla voce corrispondente.

[Home page](#) > [QUERY](#) > prima query

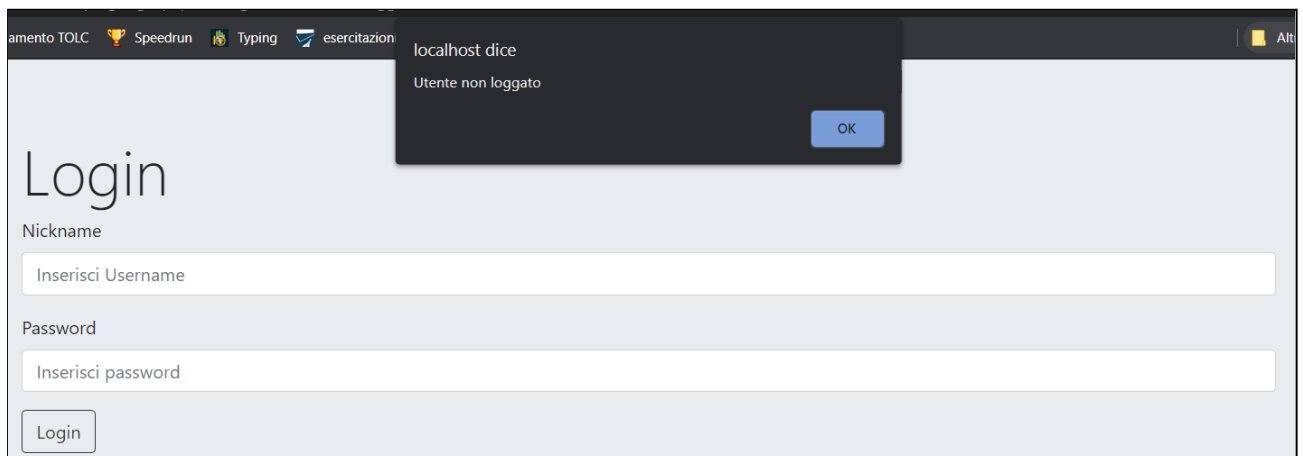
```
<nav style="--bs-breadcrumb-divider: url(&#34;data:image/svg+xml,%3Csvg xmlns='h  
<ol class="breadcrumb">  
  <li class="breadcrumb-item"><a href=" ../control.php">Home page</a></li>  
  <li class="breadcrumb-item"><a href="query_list.php">QUERY</a></li>  
  <li class="breadcrumb-item active" aria-current="page">prima query</li>  
</ol>  
</nav>
```

Per evitare che un utente acceda ad una sottopagina senza essere prima passato attraverso la pagina di login ho effettuato dei controlli php in ogni sottopagina, in fase di login creo una sessione e creo la variabile di sessione 'user' e 'pw' (solo se le credenziali inserite sono valide).

```
session_start();
$_SESSION["user"] = $user;
$_SESSION["pw"] = $pw;
header("location: control.php");
```

I controlli eseguiti in ogni pagina verificano il passaggio dell'utente tramite la pagina di login, se ciò non è avvenuto l'utente viene reindirizzato alla pagina di login e un alert segnala l'errore.

```
<?php
require_once $_SERVER['DOCUMENT_ROOT'].'/elaborato_prog/config.php';
session_start();
if (!($_SESSION && key_exists("user",$_SESSION))){
    header("location: login.php?message=Utente+non+loggato");
}
?>
```



The screenshot shows a web browser window with a dark theme. The address bar shows 'localhost dice'. The page title is 'Login'. Below the title, there are two input fields: 'Nickname' with the placeholder 'Inserisci Username' and 'Password' with the placeholder 'Inserisci password'. A 'Login' button is at the bottom left. An alert dialog box is open in the center, titled 'localhost dice', with the message 'Utente non loggato' and an 'OK' button.

Gli alert sono stati programmati tramite uno script javascript presente nelle varie pagine in cui voglio generarli, nel nostro caso in "login.php" è presente uno script javascript che verifica l'esistenza di una variabile "message" all'interno dell'url, se essa esiste viene stampato il contenuto tramite un alert.

```
<script type='text/javascript'>
  const url = new URL(window.location.href);
  $(document).ready(()=>{
    if(url.searchParams.get("message")){
      const message = url.searchParams.get("message");
      alert(message);
    }
  })
</script>
```

Oltre al percorso fatto dall'utente per raggiungere una sottopagina in ogni pagina è presente un tasto di "logout" in alto a destra che elimina la sessione e reindirizza l'utente alla pagina di login.



```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <div class="container px-4 px-lg-5">
    <a class="navbar-brand" href="#">Area riservata</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse">
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0 ms-lg-4"></ul>
      <form class="d-flex" action = "login.php">
        <button class="btn btn-outline-dark" type="submit">
          logout
        </button>
      </form>
    </div>
  </div>
</nav>
```

INTERROGAZIONI AL DATABASE (QUERIES)

All'interno della traccia d'esame venivano richieste 3 interrogazioni principali, per visualizzare i risultati di esse ho inserito dei pulsanti bootstrap che reindirizzano alle varie queries.

Home page

In questa pagina puoi:

Svolgere le query
apri la finestra per svolgere le query sul database

QUERY

Tabella dei sensori
apri la finestra per vedere la tabella dei sensori in tempo reale

SENSORI

Accessi eventi
apri la finestra per vedere gli accessi nei vari eventi

ACCESSI

Gestione aziendale
apri la finestra per vedere le opzioni per la gestione dell'azienda

GESTIONE

[Home page](#) > QUERY

In questa pagina puoi:

Svolgere la query 1
Eventi del prossimo mese

QUERY 1

Svolgere la query 2
Evento con il minor numero di ingressi

QUERY 2

Svolgere la query 3
Personale che potrebbe sostituire un addetto

QUERY 3

Copyright © Granda eventi snc 2021

Di seguito trovate il codice di ogni query da me eseguita per ottenere i risultati richiesti:

PRIMA QUERY ->

```
SELECT E.nome AS 'nome', E.dataInizio AS 'inizio', E.numGiorni AS
'durata', E.budget AS 'budget',
        C.nome AS 'citta', L.nome AS 'luogo', count(*) AS
'personale ingaggiato', ((SELECT sum(ME.numPersone)
        FROM mansioniineventi ME WHERE ME.idEvento =
E.id)-count(*)) AS 'personale mancante'
FROM eventi E, citta C, luoghi L, personaleineventi PE
WHERE E.idLuogo = L.id AND L.idCitta = C.id
        AND PE.idEvento = E.id AND ((MONTH(E.dataInizio)-1 =
MONTH(CURRENT_DATE())) AND
        (YEAR(CURRENT_DATE()) = YEAR(E.dataInizio))) OR
        ((YEAR(CURRENT_DATE()) = (YEAR(E.dataInizio)-1)) AND
```

```
((MONTH(CURRENT_DATE()) = 12) AND (MONTH(E.dataInizio) = 1)))) GROUP BY E.id
```

[Home page](#) > [QUERY](#) > prima query

Eventi del prossimo mese con numero di personale attualmente ingaggiato e numero di personale mancante

| Evento | Data inizio | Durata (giorni) | Budget | Città | Luogo | Personale ingaggiato | Personale mancante |
|------------|-------------|-----------------|--------|--------|-----------------|----------------------|--------------------|
| Milan-Juve | 2021-06-02 | 1 | 500 | Milano | Stadio San Siro | 2 | 1 |

SECONDA QUERY ->

```
SELECT E.nome AS 'nomeEvento', count(*) AS 'numAccessi'
      FROM log L, sensori S, tipi T, sensoriineventi SE,
areeineventi AE, eventi E
      WHERE L.idSensore = S.id AND S.idTipo = T.id AND T.nome =
'ip camera' AND SE.idSensore = L.idSensore
      AND SE.idAreaInEvento = AE.id AND AE.idEvento = E.id AND
(E.dataInizio+E.numGiorni-1) < CURDATE()
      AND L.data BETWEEN E.dataInizio AND
(E.dataInizio+E.numGiorni-1) GROUP BY E.nome HAVING numAccessi =
      (SELECT MIN(num) FROM (SELECT count(*) AS 'num' FROM log L,
sensori S, tipi T, sensoriineventi SE, areeineventi AE,
      eventi E WHERE L.idSensore = S.id AND S.idTipo = T.id AND
T.nome = 'ip camera' AND SE.idSensore = L.idSensore AND
      SE.idAreaInEvento = AE.id AND AE.idEvento = E.id AND
(E.dataInizio+E.numGiorni-1) < CURDATE() AND L.data BETWEEN
      E.dataInizio AND (E.dataInizio+E.numGiorni-1) GROUP BY
E.nome) x)
```

[Home page](#) > [QUERY](#) > seconda query

Evento che ha registrato il minor numero di ingressi

| Evento | Numero di accessi |
|--------------------------------|-------------------|
| Festival del cinema di Venezia | 117 |

La terza query è stata eseguita con più interrogazioni per verificare la correttezza degli input inseriti dall'utente e perchè la lunghezza di essa è variabile, dipende dal numero di mansioni che un addetto svolge all'interno dell'evento nel quale dev'essere sostituito.

TERZA QUERY ->

```
SELECT id FROM personale P WHERE P.cf = :addetto
```

```
SELECT id FROM eventi E WHERE E.nome = :evento
```

```
SELECT MP.idMansione FROM personale P, eventi E, personaleineventi PE,
      mansionipersonale MP WHERE E.nome = :evento AND
P.cf = :addetto AND E.id = PE.idEvento
      AND PE.idMansionePersona = MP.id AND MP.idPersona
= P.id
```

```
$query = "SELECT DISTINCT P.nome, P.cognome, P.nascita, P.email, P.cf
FROM personale P, mansionipersonale MP
WHERE MP.idPersona = P.id AND P.cf <> :addetto";
foreach($results as $res){
    $query.=" AND MP.idPersona IN
    (SELECT MP.idPersona
    FROM mansionipersonale MP
    WHERE MP.idMansione = ".$res["idMansione"].")";
}
```

[Home page](#) > [QUERY](#) > terza query

Elenco personale abile per le stesse mansioni di un addetto che dev'essere sostituito in un altro evento

Codice fiscale Addetto:

Nome evento:

[Home page](#) > [QUERY](#) > terza query

| Nome | Cognome | Nascita | Email | Codice fiscale |
|----------|---------|------------|-----------------------|------------------|
| Gianluca | Brodo | 1996-03-16 | gianluca.392@yahoo.it | BRDGLC96C16L219S |

Se l'addetto non ricopre mansioni in quell'evento oppure l'addetto/l'evento non esiste viene segnalato il tutto.

localhost dice

Questo addetto non esiste

OK

localhost dice

Questo evento non esiste

OK

localhost dice

Questo addetto non ricopre mansioni in questo evento

OK

All'interno del sito se si clicca sul bottone "Gestione aziendale" si viene reindirizzati su una pagina che garantisce la gestione (funzionalità CRUD) delle varie tabelle del database.

[Home page](#) > GESTIONE

In questa pagina puoi:

| | | |
|--|---|---|
| Gestire il personale apri la finestra per gestire il personale PERSONALE | Gestire le città apri la finestra per gestire le città CITTÀ | Gestire i luoghi apri la finestra per gestire i luoghi nelle città LUOGHI |
| Gestire le aree apri la finestra per gestire le aree dei vari eventi AREE | Gestire i budget apri la finestra per gestire i budget degli eventi BUDGET | Gestire gli eventi apri la finestra per gestire gli eventi EVENTI |
| Gestire le mansioni apri la finestra per gestire le mansioni MANSIONI | Gestire le mansioni negli eventi apri la finestra per gestire le mansioni negli eventi MANSIONI IN EVENTI | Gestire le mansioni del personale apri la finestra per gestire le mansioni del personale MANSIONI DEL PERSONALE |
| Gestire il personale negli eventi apri la finestra per gestire il personale negli eventi PERSONALE IN EVENTI | Gestire i sensori apri la finestra per gestire i sensori SENSORI | Gestire le tipologie di sensori apri la finestra per gestire le tipologie di sensori TIPOLOGIE DI SENSORI |
| Gestire i sensori negli eventi apri la finestra per gestire i sensori negli eventi SENSORI IN EVENTI | | |

I pulsanti gialli indicano che la pagina di gestione corrispondente non è ancora accessibile in quanto è ancora da sviluppare, se si preme su uno dei seguenti bottoni si viene reindirizzati ad una pagina “workInProgress.php”.

Area riservata logout

Benvenuto nell'area riservata!

da qui potrai effettuare tutti i controlli

[Home page](#) > [GESTIONE](#) > Lavoro in corso

LAVORO IN CORSO

Copyright © Granda eventi snc 2021

Gli altri pulsanti garantiscono l'accesso alla gestione della tabella selezionata e permettono all'utente di eseguire varie azioni.

[Home page](#) > [GESTIONE](#) > Gestione dei luoghi

In questa pagina puoi:

Inserire un luogo

apri la finestra per inserire un luogo di una città

[Inserisci un luogo](#)

Eliminare un luogo

apri la finestra per eliminare un luogo

[Elimina un luogo](#)

Visualizzare tutti i luoghi

apri la finestra per visualizzare tutti i luoghi

[Visualizza tutti i luoghi](#)

Visualizzare i luoghi di una città

apri la finestra per visualizzare i luoghi di una città

[Visualizza i luoghi di una città](#)

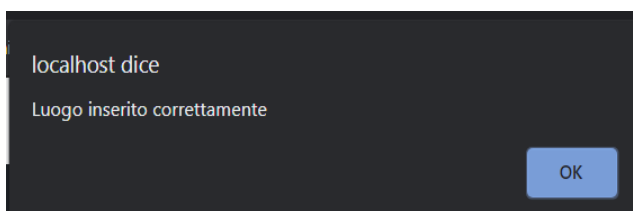
[Home page](#) > [GESTIONE](#) > [Gestione luoghi](#) > Inserisci luogo

Inserisci luogo

Nome del luogo:

Nome della città:

CAP della città:

Se invece si inserisce dei dati errati vengono effettuati tutti i relativi controlli e se vengono rilevati degli errori vengono segnalati, sempre tramite degli alert programmati in Javascript.

[Home page](#) > [GESTIONE](#) > [Gestione luoghi](#) > Inserisci luogo

Inserisci luogo

Nome del luogo:

Nome della città:

CAP della città:

Ogni query controlla gli input inseriti dall'utente e segnala eventuali errori (i controlli vengono eseguiti con php e gli errori vengono segnalati con gli alert di javascript). Per rendere sicure le interrogazioni ho utilizzato PDO e le query sono state prima preparate e dopo eseguite (le variabili vengono inserite con il metodo bindParam()).

```
try{
    $con = new PDO("mysql:host=localhost;dbname=".$dbname,"root");
    $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $nome = $_POST['nome'];
    $cap = $_POST['cap'];
    $command = $con -> prepare("INSERT INTO citta (cap, nome) VALUES (:cap,:nome)");
    $command -> bindParam(":cap", $cap, PDO::PARAM_STR);
    $command -> bindParam(":nome", $nome, PDO::PARAM_STR);
    $command -> execute();
    header("location: aggiungiCitta.php?message=Citta+inserita+correttamente");
}catch(PDOException $e){
    die("<form method = 'POST' action = '../..../control.php'>
        <button type = 'submit' class = 'btn btn-outline-dark'>Pagina di controllo</button>
    </form> <br> Errore: " . $e -> getMessage());
}
```

Se viene rilevato un errore di Mysql stampo un pulsante in grado di reindirizzare l'utente sulla pagina di controllo "control.php" per evitare che egli rimanga bloccato.

API E ACQUISIZIONE DI ESSE

Per facilitare l'inserimento degli input all'utente ho generato dei dropdown tramite javascript, gli elementi presi da javascript si trovano su delle pagine apposite da me programmate in php che effettuano delle interrogazioni al database e ritornano dei valori in formato json. Di seguito è presente il codice per generare dei dropdown contenenti tutti i codici fiscali degli addetti presenti nel database.

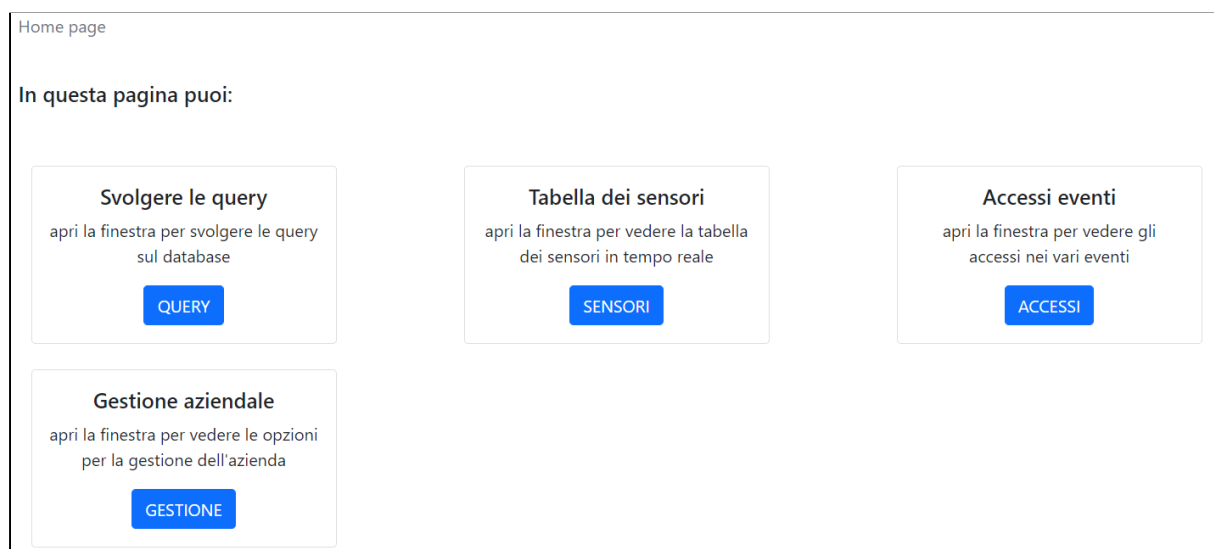
Api php che permette la visualizzazione dei vari codici fiscali:

```
<?php
require_once $_SERVER['DOCUMENT_ROOT'].'/elaborato_prog/config.php';
session_start();
if (!($SESSION && key_exists("user",$SESSION))){
    header("location: ../../login.php?message=Utente+non+loggato");
}else{
    try{
        $con = new PDO("mysql:host=localhost;dbname=".$dbname,"root");
        $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $command = $con -> prepare("SELECT P.cf FROM personale P");
        $command -> execute();
        $results = $command -> fetchall(PDO::FETCH_ASSOC);
        echo json_encode($results);
    }catch(PDOException $e){
        die("Errore: " . $e -> getMessage());
    }
}
```

Codice javascript che aggiunge i codici fiscali ricevuti ad un dropdown:

```
<script type='text/javascript'>
  const url = new URL(window.location.href);
  $(document).ready(async()=>{
    if(url.searchParams.get("message")){
      const message = url.searchParams.get("message");
      alert(message);
    }
    const RESCODICIFISCALI = await fetch(`${url.origin}/elaborato_prog/api/personale/getAllCF.php`);
    const cf = await RESCODICIFISCALI.json();
    cf.forEach((el)=>{
      const codiceFiscale = el.cf;
      $("#addetti").append(`<option value="${codiceFiscale}">`);
    })
  })
</script>
```

GRAFICO ACCESSI (CHARTJS)



Per visualizzare il numero di accessi nei vari eventi ho utilizzato php per ricevere i risultati. Per contare gli accessi ad un evento controllo i dati inviati dalle ip camere presenti all'area di accesso all'evento (1 dato = 1 persona). Siccome una ip camera può essere usata in più eventi ma non contemporaneamente controllo la data in cui è stato inviato il segnale e le date di inizio e fine degli eventi, il sensore deve anche essere presente nella tabella "sensoriineventi".

```

<?php
require_once $ SERVER['DOCUMENT_ROOT'].'/elaborato_prog/config.php';
session_start();
if (!($_SESSION && key_exists("user",$_SESSION))){
    header("location: ../login.php?message=Utente+non+loggato");
}
try{
    $con = new PDO("mysql:host=localhost;dbname=".$dbname,"root");
    $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $command = $con -> prepare("SELECT E.nome AS 'nomeEvento', count(*) AS 'numAccessi'
    FROM log L, sensori S, tipi T, sensoriineventi SE, areeineventi AE, eventi E
    WHERE L.idSensore = S.id AND S.idTipo = T.id AND T.nome = 'ip camera' AND SE.idSensore = L.idSensore
    AND SE.idAreaInEvento = AE.id AND AE.idEvento = E.id AND (E.dataInizio+E.numGiorni-1) < CURDATE()
    AND L.data BETWEEN E.dataInizio AND (E.dataInizio+E.numGiorni-1) GROUP BY E.nome");
    $command -> execute();
    $results = $command -> fetchall(PDO::FETCH_ASSOC);
    $data = array();
    foreach($results as $res){
        $data[] = $res;
    }
    print json_encode($data);
}catch(PDOException $e){
    die("<form method = 'POST' action = '../control.php'>
    <button type = 'submit' class = 'btn btn-outline-dark'>Pagina di controllo</button>
    </form><br>Errore: " . $e -> getMessage());
}

```

I risultati di questa API vengono di seguito presi da javascript e ajax per creare l'istogramma corrispondente.

```

$(document).ready(function(){
    $.ajax({
        url: "http://localhost/elaborato_prog/accessi/accessiEventi.php",
        method: "GET",
        success: function(response){
            console.log(response);
            data = JSON.parse(response);
            var nomiEventi = [];
            var numAccessi = [];
            for(var i=0; i<data.length;i++) {
                nomiEventi.push(data[i].nomeEvento);
                numAccessi.push(data[i].numAccessi);
            }
            var chartdata = {
                labels: nomiEventi,
                datasets : [
                    {
                        label: 'Accessi evento',
                        background: 'rgba(200, 200, 200, 0.75)',
                        borderColor: 'rgba(200, 200, 200, 0.75)',
                        hoverBackgroundColor: 'rgba(200, 200, 200, 1)',
                        hoverBorderColor: 'rgba(200, 200, 200, 1)',
                        data: numAccessi
                    }
                ]
            };

            var ctx = $("#mycanvas");
            var barGraph = new Chart(ctx, {
                type: 'bar',
                data: chartdata
            });
        },
        error: function(data){
            console.log(data);
        }
    });
});

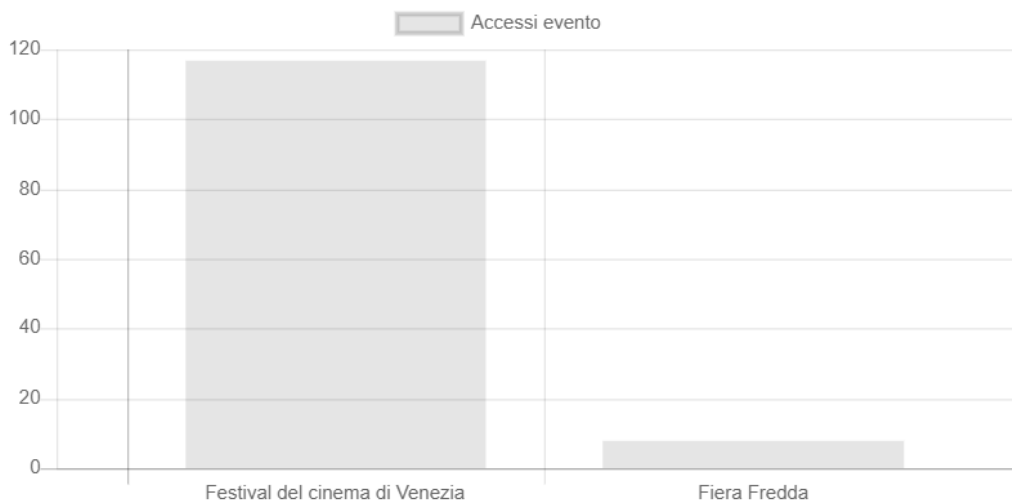
```


La variabile "ctx" prende un tag presente nella pagina di visualizzazione del grafico e lo utilizza per inserire l'istogramma corrispondente.

```
<h1> Accessi negli eventi </h1>
<div id="chart-container">
  <canvas id="mycanvas">
  </canvas>
</div>
<script type="text/javascript" src="js/jquery-3.6.0.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/chart.js/2.4.0/Chart.min.js"></script>
<script type="text/javascript" src="js/app.js"></script>
```

[Home page](#) > ACCESSO EVENTI

Accessi negli eventi



Copyright © Granda eventi snc 2021

Nella pagina di visualizzazione sono presenti solo gli eventi già terminati siccome se un evento è in corso gli accessi possono ancora aumentare dunque non sarebbe utile visualizzare gli accessi e valutare il successo ottenuto da quell'evento.

SIMULATORE SENSORI

Per simulare l'invio dei dati dei vari sensori ho utilizzato un programma Python includendo la libreria 'numpy' per la generazione di valori random la libreria 'time' per stoppare il programma per tot. secondi / centesimi di secondo e 'requests' per

collegarmi ad alcune API in php da me create. Il programma dev'essere eseguito dal prompt dei comandi di anaconda3.

L'ip camera ha uno scanner di temperatura e invia dei numeri reali che vanno da 36 a 38 (sopra i 37.5 il dato è allarmante). I metal detector inviano dei dati che possono essere 0 o 1 (0 -> metallo non rilevato, 1 -> metallo rilevato, allarme). Gli smoke detector inviano dei dati che vanno da 0 a 650 (sopra i 600 il dato è allarmante). Di seguito è presente un segmento di codice.

```
import requests as rq
import numpy as np
from random import randrange
import time
count febbre = 0
count metallo = 0
count fumo = 0
inviaDati = "http://localhost/elaborato_prog/dati_sensori/riceviDatiSensori.php"
while(1):
    sensori = rq.get("http://localhost/elaborato_prog/api/sensori/getAllSensorsInEvents.php").json()
    areaMax = int(rq.get("http://localhost/elaborato_prog/api/eventi/getMaxAreaId.php").json()[0]['idMaxArea'])
    count_area = randrange(areaMax+1)
    print(f"Area: {count_area}")
    stop = float(np.random.uniform(0,2))
    time.sleep(stop)
    print(f"Stop: {stop}")
    for sensore in sensori:
        if(count_area == int(sensore['area'])):
            if sensore['nome'] == 'ip camera':
                temperatura = round(float(np.random.uniform(36,38)),2)
                if(temperatura > 37.5):
                    count febbre += 1
                    if(count febbre == 4):
                        #allarme
                        print(f"{sensore['nome']},{sensore['id']},{temperatura}")
                        esito = rq.get(inviaDati, params = {"idSensore":sensore["id"],"dato":temperatura})
                        esito = None
                        count febbre = 0
                    else:
                        temperatura = round(float(np.random.uniform(36,37)),2)
                        print(f"{sensore['nome']},{sensore['id']},{temperatura}")
                        esito = rq.get(inviaDati, params = {"idSensore":sensore["id"],"dato":temperatura})
                        esito = None
                else:
                    print(f"{sensore['nome']},{sensore['id']},{temperatura}")
                    esito = rq.get(inviaDati, params = {"idSensore":sensore["id"],"dato":temperatura})
                    esito = None
```

La variabile "sensori" contiene tutti i sensori presenti nei vari eventi in corso che inviano dati, l'area viene generata randomicamente poichè quando un sensore di un'area rileva dei dati tutti gli altri sensori presenti nella stessa area rilevano anche loro dei dati dunque devo gestire l'invio dei dati in contemporanea (es. metal detector e ip camera presente ad un'entrata di un evento).

I vari dati vengono spediti alla pagina "riceviDatiSensori.php" che invia i dati al database e li salva nella tabella log.

```

<?php //idSensore,dato
require_once $_SERVER['DOCUMENT_ROOT'].'/elaborato_prog/config.php';
if($_SERVER["REQUEST_METHOD"]=="GET"){
    if(key_exists("idSensore",$GET) && key_exists("dato",$GET)){
        $idSensore = $_GET["idSensore"];
        $dato = $_GET["dato"];
        try{
            date_default_timezone_set('Europe/Rome');
            $hour = explode(" ",date('Y-m-d H:i:s'))[1];
            $con = new PDO("mysql:host=localhost;dbname=". $dbname,"root");
            $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            $command = $con -> prepare("INSERT INTO log (idSensore, data, ora, dato) VALUES (:idSensore,CURDATE(),:hour,:dato)");
            $command -> bindParam(":idSensore", $idSensore, PDO::PARAM_STR);
            $command -> bindParam(":hour", $hour, PDO::PARAM_STR);
            $command -> bindParam(":dato", $dato, PDO::PARAM_STR);
            $command -> execute();
        }catch(PDOException $e){
            die("<form method = 'POST' action = '../control.php'>
                <button type = 'submit' class = 'btn btn-outline-dark'>Pagina di controllo</button>
                </form> <br> Errore: " . $e -> getMessage());
        }
    }
}
}

```

La pagina “RealTimeLogSensori.php” stampa la tabella log corrispondente e segna in rosso le righe contenenti dati allarmanti.

```

<?php
try{
    $con = new PDO("mysql:host=localhost;dbname=". $dbname,"root");
    $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $command = $con -> prepare("SELECT DISTINCT L.id AS 'IdLog', L.idSensore, T.nome AS 'Tipo', A.nome AS 'Nome area',
        E.nome AS 'Nome evento', L.data, L.ora, L.dato FROM log L, sensori S, tipi T, sensoriineventi SE, areeineventi AE,
        aree A, eventi E WHERE L.idSensore = S.id AND S.idTipo = T.id AND S.id = SE.idSensore AND SE.idAreaInEvento =
        AE.id AND AE.idArea = A.id AND AE.idEvento = E.id AND L.data >= E.dataInizio AND L.data <=
        (E.dataInizio+E.numGiorni-1) ORDER BY L.id DESC");
    $command -> execute();
    $results = $command -> fetchAll(PDO::FETCH_ASSOC);
    echo "<table><tr><th>IdLog</th><th>idSensore</th><th>Tipo</th><th>Nome area</th>
        <th>Nome evento</th><th>Data</th><th>Ora</th><th>Dato</th></tr>";
    foreach($results as $res){
        if(($res["Tipo"] == "metal detector" && $res["dato"] == 1) || ($res["Tipo"] == "smoke detector" && $res["dato"] >= 600)
            || ($res["Tipo"] == "ip camera" && $res["dato"] > 37.5)){
            //rosso, dato allarmante
            $text = "<span style='color:#FF0000'><strong>";
            $text2 = "</strong>";
        }else{
            //nero, dato normale
            $text = "<span style='color:#000000'>";
            $text2 = "";
        }
        echo "<tr><td>".$text.$res["IdLog"].$text2."</td><td>".
            $text.$res["idSensore"].$text2."</td><td>".$text.$res["Tipo"].$text2."</td>
            <td>".$text.$res["Nome area"].$text2."</td><td>".$text
            .$res["Nome evento"].$text2."</td><td>".$text.$res["data"].$text2."</td>
            <td>".$text.$res["ora"].$text2."</td><td>".$text.$res["dato"].$text2."</td></tr>";
    }
    echo "</table>";
}

```

Infine uno script di codice contenuto in “logSensori.php” permette la visualizzazione della tabella log e l’aggiornamento dei dati inviati dai sensori in tempo reale (si visualizzano i dati inviati dai sensori in tempo reale sulla pagina web corrispondente senza essere obbligati a ricaricare la pagina, è tutto fatto automaticamente da questo script di codice).

```
<h5>Tabella log sensori in eventi: </h5>
<br><br>
<div id="link_wrapper"></div>
</div>
</section>
<!-- Footer --><br>
<footer class="py-5 bg-dark">
  <div class="container"><p class="m-0 text-center text-white">Copyright &copy; Granda eventi snc 2021</p></div>
</footer>
<!-- Bootstrap core JS
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js"></script>
<!-- Core theme JS-->
<script src="js/scripts.js"></script>
</body>
</html>
<script>
function loadXMLDoc() {
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    document.getElementById("link_wrapper").innerHTML =
      this.responseText;
  }
};
xhttp.open("GET", "RealTimeLogSensori.php", true);
xhttp.send();
}

setInterval(function(){
  loadXMLDoc();
  // 1 sec
}, 1000);

window.onload = loadXMLDoc;
</script>
```

*La visualizzazione finale si può ottenere premendo sul bottone “**SENSORI**” nell’home page.*

Home page

In questa pagina puoi:

Svolgere le query

apri la finestra per svolgere le query sul database

QUERY

Tabella dei sensori

apri la finestra per vedere la tabella dei sensori in tempo reale

SENSORI

Accessi eventi

apri la finestra per vedere gli accessi nei vari eventi

ACCESSI

Gestione aziendale

apri la finestra per vedere le opzioni per la gestione dell'azienda

GESTIONE

| | | | | | | | |
|------------|----------|-----------------------|-----------------|---------------------------------------|-------------------|-----------------|-------------|
| 221 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:33 | 0 |
| 220 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:33 | 36.41 |
| 219 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:24 | 0 |
| 218 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:24 | 36.93 |
| 217 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:14 | 0 |
| 216 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:15:14 | 36.19 |
| 215 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:59 | 0 |
| 214 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:59 | 36.46 |
| 213 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:54 | 0 |
| 212 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:54 | 37.7 |
| 211 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:44 | 0 |
| 210 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:44 | 36.65 |
| 209 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:42 | 0 |
| 208 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:42 | 36.01 |
| 207 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:39 | 0 |
| 206 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:39 | 37.44 |
| 205 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:39 | 0 |
| 204 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:39 | 36.49 |
| 203 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:14 | 0 |
| 202 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:14 | 36.89 |
| 201 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:13 | 1 |
| 200 | 3 | ip camera | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:13 | 36.72 |
| 199 | 4 | metal detector | accesso1 | Festival del cinema di Venezia | 2021-05-23 | 11:14:12 | 0 |

Le righe 201 e 212 sono scritte in grassetto e in rosso siccome contengono dei dati allarmanti (metal detector -> 1, ip camera -> >37.5).

GESTIONE BUDGET EVENTI

Per la gestione dei budget dei vari eventi ho creato un pulsante apposito nella scheda "gestione".

[Home page](#) > [GESTIONE](#)

In questa pagina puoi:

| | | |
|---|---|---|
| Gestire il personale apri la finestra per gestire il personale PERSONALE | Gestire le città apri la finestra per gestire le città CITTÀ | Gestire i luoghi apri la finestra per gestire i luoghi nelle città LUOGHI |
| Gestire le aree apri la finestra per gestire le aree dei vari eventi AREE | Gestire i budget apri la finestra per gestire i budget degli eventi BUDGET | Gestire gli eventi apri la finestra per gestire gli eventi EVENTI |
| Gestire le mansioni apri la finestra per gestire le mansioni MANSIONI | Gestire le mansioni negli eventi apri la finestra per gestire le mansioni negli eventi MANSIONI IN EVENTI | Gestire le mansioni del personale apri la finestra per gestire le mansioni del personale MANSIONI DEL PERSONALE |

[Home page](#) > [GESTIONE](#) > Gestione budget in eventi

Nome dell'evento:

Successivamente vengono stampate 2 tabelle, una contenente il budget disponibile per l'evento selezionato e un'altra contenente tutte le mansioni disponibili con la retribuzione oraria corrispondente, un campo della tabella permette di inserire il numero di personale che si vuole per ogni mansione.

[Home page](#) > [GESTIONE](#) > Gestione budget in eventi

Calcolo budget evento 'Festival del cinema di Venezia'

| Mansione | Retribuzione oraria | Numero personale |
|-------------------|---------------------|--------------------------------|
| buttafuori | 15.93 | <input type="text" value="0"/> |
| primo soccorso | 10 | <input type="text" value="0"/> |
| operatore monitor | 10.45 | <input type="text" value="0"/> |

| Nome Evento | Budget Evento |
|--------------------------------|---------------|
| Festival del cinema di Venezia | 4500 |

Premendo sul tasto “calcola budget” viene calcolato il costo totale che si avrebbe con il numero di personale inserito nella colonna “Numero personale”. Ipotizziamo di aver inserito 1 buttafuori, 2 persone del primo soccorso, 1 operatore monitor.

[Home page](#) > [GESTIONE](#) > [Gestione budget in eventi](#) > Calcolo budget

Statistiche budget

Durata evento -> 5 giorni, 8 ore per giorno




Budget disponibile -> 4500

Budget richiesto -> 1855.2

Percentuale budget richiesto su budget disponibile -> 41.23%

Successivamente è possibile salvare le mansioni richieste sul database (tabella -> mansioniineventi).

Questa è la tabella prima (nessuna mansione segnata per l'evento 'Festival del cinema di Venezia' id -> 4).






























| <div><div><div>↔</div><div>T</div><div>→</div></div><div>▼</div></div> | | | | | | id | idEvento | idMansione | numPersone | |
|--|---|----------|---|-------|---|---------|----------|------------|------------|---|
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 1 | 1 | 1 | 2 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 2 | 3 | 1 | 2 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 3 | 3 | 2 | 1 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 4 | 1 | 2 | 1 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 5 | 2 | 4 | 1 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 12 | 2 | 1 | 4 |
| <input type="checkbox"/> |  | Modifica |  | Copia |  | Elimina | 13 | 2 | 2 | 1 |

Dopo aver premuto sul pulsante:

[Home_page](#) > [GESTIONE](#) > [Gestione budget in eventi](#) > Salvo mansioni

Mansioni inserite nel database

Ritorna alla pagina di controllo

| <div><div>← T →</div><div>▼</div></div> | | | | id | idEvento | idMansione | numPersone | |
|---|---|----------|---|---|----------|------------|------------|---|
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 1 | 1 | 1 | 2 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 2 | 3 | 1 | 2 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 3 | 3 | 2 | 1 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 4 | 1 | 2 | 1 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 5 | 2 | 4 | 1 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 12 | 2 | 1 | 4 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 13 | 2 | 2 | 1 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 21 | 4 | 1 | 1 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 22 | 4 | 2 | 2 |
| <input type="checkbox"/> |  | Modifica |  Copia |  Elimina | 23 | 4 | 4 | 1 |

In questa pagina ho implementato numerosi controlli, per esempio se inseriamo un numero di mansioni pari a 0 in un determinato evento il quale ha precedentemente richiesto delle mansioni viene segnalato il tutto.

[Home page](#) > [GESTIONE](#) > [Gestione budget in eventi](#) > Calcolo budget

Statistiche budget

Durata evento -> 5 giorni, 8 ore per giorno

Budget disponibile -> 4500

Budget richiesto -> 0

Percentuale budget richiesto su budget disponibile -> 0%

Salva le mansioni richieste sul database

[Home page](#) > [GESTIONE](#) > [Gestione budget in eventi](#) > Salvo mansioni

Hai richiesto 0 persone per la mansione 'buttafuori' ma sul database sono salvate 1 persone per questa mansione, rivedere il personale.

Hai richiesto 0 persone per la mansione 'primo soccorso' ma sul database sono salvate 2 persone per questa mansione, rivedere il personale.

Hai richiesto 0 persone per la mansione 'operatore monitor' ma sul database sono salvate 1 persone per questa mansione, rivedere il personale.

Questa è una parte del codice php scritto per la pagina che calcola il budget e le statistiche dell'evento. Avendo il numero di persone per ogni mansione in "mansioniineventi", la retribuzione oraria di ogni mansione in "mansioni" e la durata dell'evento in "eventi" si può calcolare il budget richiesto e paragonarlo al budget disponibile.

```
echo "<h1>Statistiche budget</h1>";
try{
    $con = new PDO("mysql:host=localhost;dbname=".$dbname,"root");
    $con -> setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $command = $con -> prepare("SELECT numGiorni, hPerG FROM eventi WHERE nome = :nomeEvento ");
    $command -> bindParam(":nomeEvento", $_SESSION["nomeEvento"], PDO::PARAM_STR);
    $command -> execute();
    $res = $command -> fetch(PDO::FETCH_ASSOC);
    $numGiorni = $res['numGiorni'];
    $orePerGiorno = $res['hPerG'];
    $command = $con -> prepare("SELECT id, nome, retrOraria AS 'retribuzione oraria' FROM mansioni ORDER BY id");
    $command -> execute();
    $res = $command -> fetchall(PDO::FETCH_ASSOC);
    $count = 1;
    $budgetCalcolato = 0;
    foreach($res as $row){
        $budgetCalcolato = $budgetCalcolato + ($row['retribuzione oraria'] * $_POST[$count]);
        $count = $count + 1;
    }
    $budgetCalcolato = $budgetCalcolato * $numGiorni * $orePerGiorno;
    echo "Durata evento -> ". $numGiorni. " giorni, ". $orePerGiorno. " ore per giorno <br>";
    echo "Budget disponibile -> ". round($_SESSION["budget"],2). "<br>";
    echo "Budget richiesto -> ". round($budgetCalcolato,2). "<br>";
    echo "Percentuale budget richiesto su budget disponibile -> ". round(($budgetCalcolato / $_SESSION["budget"]) * 100,2). "% <br><br>";
    if((($budgetCalcolato / $_SESSION["budget"]) * 100) <= 100){
        $count = 1;
        foreach($res as $row){
            $_SESSION['numPersone'][$count] = $_POST[$count]; //numPersonePerMansione
            $count = $count + 1;
        }
        //si possono salvare le mansioni richieste sul db
        echo "<form method = 'POST' action = 'salvoMansioni.php'>
        <button type = 'submit' class = 'btn btn-outline-dark'>Salva le mansioni richieste sul database</button>
        </form>";
    }
    unset($_SESSION["budget"]);
}
```

Per salvare il numero di persone richiesto per ogni mansione ho utilizzato le variabili di sessione in maniera dinamica poichè le variabili da passare da un file all'altro dipendono dal numero di mansioni presenti sul database (possono essere aggiunte o tolte delle mansioni dalla pagina di gestione delle mansioni).

```
$change = false;
if($res == ""){
    //la mansione non è ancora stata inserita nel suddetto evento
    $command = $con -> prepare("INSERT INTO mansioniineventi (idEvento , idMansione , numPersone) VALUES (:idEvento
    :idMansione , :numPersone)");
    $command -> bindParam(":idEvento", $idEvento, PDO::PARAM_STR);
    $command -> bindParam(":idMansione", $row['id'], PDO::PARAM_STR);
    $command -> bindParam(":numPersone", $_SESSION['numPersone'].$count, PDO::PARAM_STR);
    $command -> execute();
    $change = true;
}else if($res['numPersone'] < $_SESSION['numPersone'].$count){
    //si vogliono inserire più persone
    $command = $con -> prepare("UPDATE mansioniineventi SET numPersone = :numPersone WHERE idEvento = :idEvento AND
    $command -> bindParam(":numPersone", $_SESSION['numPersone'].$count, PDO::PARAM_STR);
    $command -> bindParam(":idEvento", $idEvento, PDO::PARAM_STR);
    $command -> bindParam(":idMansione", $row['id'], PDO::PARAM_STR);
    $command -> execute();
    $change = true;
}else if($res['numPersone'] > $_SESSION['numPersone'].$count){
    //si vogliono inserire meno persone -> questo dettaglio viene segnalato
    $command = $con -> prepare("SELECT nome FROM mansioni WHERE id = :idMansione ");
    $command -> bindParam(":idMansione", $row['id'], PDO::PARAM_STR);
    $command -> execute();
    $risultato = $command -> fetch(PDO::FETCH_ASSOC);
    echo "Hai richiesto ". $_SESSION['numPersone'].$count ." persone per la mansione '". $risultato['nome']. "'
    ma sul database sono salvate ". $res['numPersone']. " persone per questa mansione, rivedere
    il personale. <br>";
}
unset($_SESSION['numPersone'].$count);
$count = $count + 1;
```

Un ulteriore controllo viene eseguito sul budget calcolato, se il budget calcolato è maggiore del budget disponibile il sito web non permette di salvare le mansioni sul database. Il pulsante "salva mansioni sul database" appare solo e solo se il budget calcolato è minore o uguale al budget disponibile.

[Home page](#) > [GESTIONE](#) > Gestione budget in eventi

Calcolo budget evento 'Fiera Fredda'

| Mansione | Retribuzione oraria | Numero personale |
|-------------------|---------------------|------------------|
| buttafuori | 15.93 | 121 |
| primo soccorso | 10 | 5 |
| operatore monitor | 10.45 | 5 |

Calcola budget

| Nome Evento | Budget Evento |
|--------------|---------------|
| Fiera Fredda | 600.23 |

[Home page](#) > [GESTIONE](#) > [Gestione budget in eventi](#) > Calcolo budget

Statistiche budget

Durata evento -> 13 giorni, 5 ore per giorno

Budget disponibile -> 600.23

Budget richiesto -> 131935.7

Percentuale budget richiesto su budget disponibile -> 21980.86%

Linguaggi utilizzati -> PHP, MySQL, Bootstrap, Python, Javascript.

Progetto completamente svolto da Acchiardi Paolo, studente della 5A robotica dell'ITIS Mario Delpozzo di Cuneo.